Received: 27 July 2023 | Revised: 26 September 2023 | Accepted: 23 November 2023

DOI: 10.1111/tgis.13126

RESEARCH ARTICLE

Transactions in GIS WILEY

QuadGridSIM: A quadrilateral grid-based method for high-performance and robust trajectory similarity analysis

Juqing Liu[1] | Jun Li[1] | Linwei Qiao[1] | Mingke Li[2] | Emmanuel Stefanakis[2] | Xuesheng Zhao[1] | Qian Huang[3] | Hao Wang[3] | Chengye Zhang[1]

[1]College of Geoscience and Surveying Engineering, China University of Mining and Technology-Beijing, Beijing, China

[2]Department of Geomatics Engineering, Schulich School of Engineering, University of Calgary, Calgary, Canada

[3]Service Lab, Huawei Technology Co. Ltd, Beijing, China

Correspondence
Jun Li and Chengye Zhang, College of Geoscience and Surveying Engineering, China University of Mining and Technology-Beijing, Beijing, China.
Email: junli@cumtb.edu.cn and czhang@cumtb.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Number: 41971355; Yueqi Young Scholar Project of China University of Mining and Technology at Beijing

Abstract

Measuring trajectory similarity is a fundamental algorithm in trajectory data mining, playing a key role in trajectory clustering, pattern mining, and classification, for instance. However, existing trajectory similarity measures based on vector representation have challenges in achieving both fast and accurate similarity measurements. On one hand, most existing methods have a high computational complexity of $O(n \times m)$, resulting in low efficiency. On the other hand, many of them are sensitive to trajectory sampling rates and lack of accuracy. This article proposes QuadGridSIM, a quadrilateral grid-based method for trajectory similarity analysis, which enables high-performance trajectory similarity measure without the cost of low effectiveness. Specifically, we first realize the multiscale coding representation of trajectory data based on quadrilateral discrete grids. Then, a novel trajectory similarity measure is defined to reduce the computational complexity of $O(n)$. Several effectiveness properties of QuadGridSIM are further optimized, including the spatial overlap, directionality, symmetry, and robustness to sampling rate variations. Experimental results based on real-world and simulated taxi trajectory data indicate that QuadGridSIM outperforms most of the other tested algorithms developed previously in terms of effectiveness, particularly

in its robustness regarding trajectory sampling rates. Furthermore, QuadGridSIM exhibits superior performance and is approximately one order of magnitude faster than previous methods in the literature. QuadGridSIM provides a solution to the low-efficiency problem of massive trajectory similarity analysis and can be applied in many application scenarios, such as route recommendation and suspect detection.

1 | INTRODUCTION

With the remarkable development of modern technologies, such as intelligent perception, mobile positioning, and wireless communication, a large quantity of trajectory data containing the location information and behavior status of individuals and groups are being generated and collected (Oueslati et al., 2023; Wang et al., 2020; Zheng, 2015). These trajectories are represented as a series of time sequence points $((P_1, t_1), (P_2, t_2), (P_3, t_3), \ldots, (P_n, t_n))$. Each point contains a spatial location $P(L, B)$, where L and B represent its geographic location (i.e., longitude and latitude), and t is its location timestamp. Trajectory data have been extensively used in various fields because of their rich information in terms of time, space, and semantics, including intelligent transportation (Keler et al., 2017; Li et al., 2019), urban computing (Dodge et al., 2020; Guo et al., 2022), and social sensing (Li et al., 2016; Liu et al., 2015). Scholars have conducted studies on various trajectory data analysis methods, among which the trajectory similarity measure is one of the most fundamental algorithms for trajectory data mining (e.g., trajectory clustering, pattern mining, and trajectory classification) (Wang et al., 2021; Zheng, 2015). Such algorithms have long been a popular research topic (Dodge et al., 2012; Li, Liu, et al., 2022; Sousa et al., 2020) and widely applied in various scenarios (Magdy et al., 2015; Petry et al., 2019; Vlachos et al., 2002), such as commuting mode identification (Buchin et al., 2011), human activity prediction (Chekol & Fufa, 2022), personalized recommendation (Liu & Seah, 2015), anomalous trajectory detection (Wang et al., 2018) and epidemic prevention and control (Kraemer et al., 2020).

Trajectory similarity measure means to quantitatively evaluate the similarity (or distance) between two trajectories. This study focused on the spatial similarity of trajectories, where two trajectories are considered similar if they share close spatial paths (Furtado et al., 2018). Previously, many spatial similarity measures for trajectories have been proposed, such as the Euclidean distance (ED), Hausdorff distance (Hausdorff, 1914), dynamic time warping (DTW) (Berndt & Clifford, 1994), the longest common subsequence (LCSS) (Vlachos et al., 2002), edit distance with real sequence (EDR) (Chen et al., 2005), FastDTW (Salvador & Chan, 2007), symmetrized segment-path distance (SSPD) (Besse et al., 2016), and uncertain movement similarity (UMS) (Furtado et al., 2018). However, those existing methods of trajectory similarity measure lack an optimal balance between effectiveness and efficiency. Specifically, the effectiveness is determined by properties such as directionality, symmetry, and robustness to sampling rate variations, whereas the efficiency refers to the algorithm's computational time complexity. Basically, they can be divided into the following three categories: (1) neither effectiveness nor efficiency. The methods such as DTW, LCSS, Hausdorff, and SSPD have high computational time complexity ($O(n \times m)$) and low efficiency, where n and m are the number of points of the trajectories to be calculated (Dodge et al., 2012; Magdy et al., 2015). In terms of effectiveness, DTW and LCSS are not effective in handling variations in trajectory sampling rates. For instance, as shown in Figure 1a, the similarity between trajectory T_O and trajectories T_A or T_B should be equal to 1 (distance is 0). However, since DTW evaluates the similarity based on the minimum value of the sum of point distances between two trajectories (Toohey & Duckham, 2015), the distance between T_O and

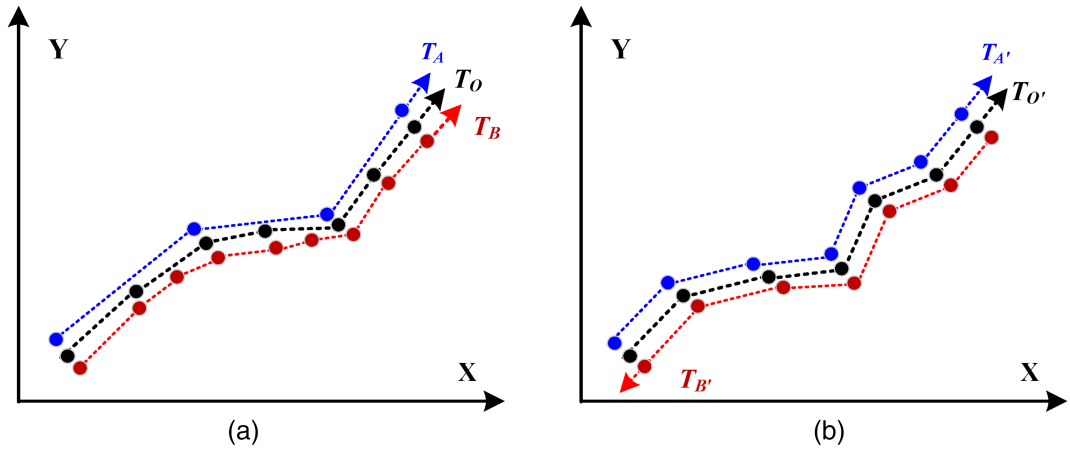

FIGURE 1 Examples of trajectory similarity measure on (a) trajectories with different sampling rates and (b) trajectories with a similar path but opposite directions.

T_A is larger than that between T_O and T_B. Likewise, LCSS calculates the maximum common subsequence of two trajectories (Vlachos et al., 2002), and consequently, the LCSS-based similarity between T_O and T_B is incorrectly greater than that between T_O and T_A. In addition, Hausdorff and SSPD lack directionality and are unable to distinguish between the similarity of two trajectories with the same path but opposite directions. Figure 1b illustrates the issue, where Hausdorff and SSPD calculate the similarity between $T_{O'}$ and $T_{A'}$ or $T_{B'}$ as 1 (distance is 0), while the similarity between $T_{O'}$ and $T_{A'}$ with the same direction is higher than that between $T_{O'}$ and $T_{B'}$. (2) High efficiency but limited effectiveness. Although ED and FastDTW are known for their low algorithm complexity $O(n)$ and high performance, they do not improve the effectiveness of trajectory similarity measures. FastDTW is optimized for performance based on DTW, but it still remains sensitive to the sampling rate. Meanwhile, ED cannot dynamically cope with trajectories of different lengths and accurately compute the similarity between trajectories (Su et al., 2020). (3) Effectiveness but low efficiency. UMS is a cutting-edge algorithm that excels in the effectiveness of trajectory similarity measures. It improves the robustness of the sampling rate by introducing an elliptical representation of trajectories and has properties such as directionality and symmetry. The accuracy of its similarity calculation is significantly better than other algorithms (Furtado et al., 2018). However, UMS has the same computational complexity ($O(n \times m)$) as LCSS, DTW, Hausdorff, and SSPD, and its computational efficiency is not high enough to meet the demand for real-time or near real-time high-performance computing in the era of the internet of things (IoT).

Recent decades have witnessed a rapid development of computer technology and geographic information technology, as well as the increasingly complicated representation, update, and analysis of geospatial information. The value of the geographic grid model has been recognized by previous research due to its advantageous features such as uniformity, discretization, multiresolution, and cell indexing (Gibb et al., 2022; Goodchild, 2018; Li & Stefanakis, 2020; Peterson, 2016; Yao et al., 2020). The geographic grid model can be defined as a system for representing a geographic entity using a series of discrete, regular cells according to some predefined rules (OGC, 2017; Ulmer et al., 2020; Zhou et al., 2010, 2023). Commonly used geographic grid geometries are triangle, quadrilateral, and hexagon (Hojati et al., 2021; OGC, 2017). Compared to triangular and hexagonal grids, the quadrilateral grid offers some advantages including perfectly hierarchical data structures, compatibility with existing quadtree-based algorithms, suitability for hardware and display devices, as well as integration with prevailing geospatial reference systems (Béjar et al., 2023; Bowater & Stefanakis, 2020; Gibb, 2016). Accordingly, the quadrilateral grid exhibits remarkable potential for trajectory data mining applications (Bowater & Wachowicz, 2020; May Petry et al., 2020; Qian et al., 2019). For example, Li, Liu, et al. (2022)

proposed gsstSIM, a synchronized spatiotemporal trajectory similarity method based on the quadrilateral grid. This method simultaneously measures trajectory similarity in both spatial and temporal dimensions. gsstSIM improves the robustness and computational efficiency compared to the existing algorithms, excelling in time-sensitive applications such as moving flock pattern mining. However, gsstSIM does not support applications that focus on spatial information mining such as trajectory spatial clustering and anomaly detection. This limitation stems from gsstSIM's strict integration of spatial and temporal dimensions during trajectory data modeling, preventing the independent measurement of trajectory spatial similarity. As depicted in Figure 2, trajectories T_a and T_b do not overlap in space–time, resulting in a spatiotemporal similarity of 0, that is, gsstSIM $(T_a, T_b) = 0$, despite their spatial similarity. Thus, there is a demand for a fundamental, robust, and high-performance trajectory spatial similarity method based on quadrilateral grids.

Based on the above observation, this article proposes QuadGridSIM, a quadrilateral grid-based method for trajectory similarity analysis, to provide a high-performance algorithm for trajectory data mining without the loss of effectiveness indicated by directionality, symmetry, and robustness to sampling rate variations. Furthermore, the effectiveness, robustness, and performance of the proposed method are verified using real and simulated trajectory data. The major contributions of this article are summarized as follows:

- We model trajectories via the quad-grid coding with quadtree structure, to implement time-sequence, multi-scale, and one-dimensional representation of trajectory space information and simplify trajectory data structure.
- A novel and high-performance method of trajectory similarity analysis, named QuadGridSIM, is proposed based on the quadrilateral grid. QuadGridSIM shows better effectiveness properties such as directionality, symmetry, and robustness to sampling rate variations while accelerating the trajectory similarity measure with low computational complexity$O\left(N_A + N_B\right)$, where N_A and N_B are the number of codes of the two trajectories.
- We conducted extensive experiments using both real-world and simulated taxi trajectory data to demonstrate the effectiveness, robustness, and performance of QuadGridSIM compared to the other state-of-the-art methods.

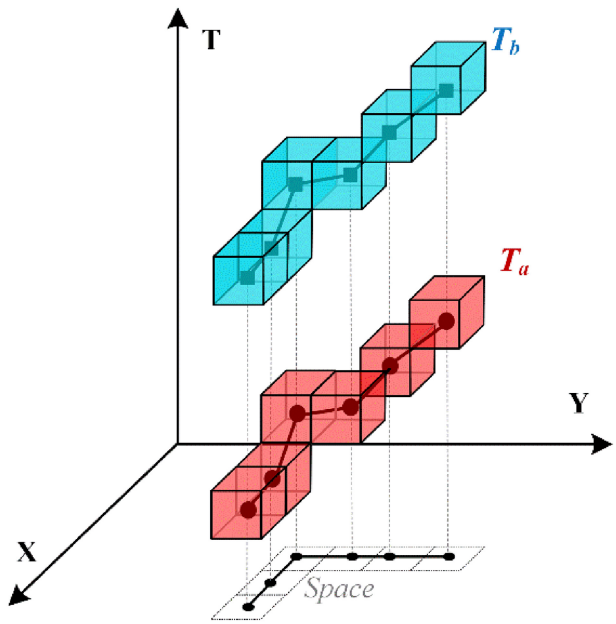

FIGURE 2 An example of trajectory similarity analysis where trajectories exhibit spatial similarity but lack spatiotemporal similarity.

The remainder of the article is organized as follows. Section 2 proposes our new similarity measure QuadGridSIM and describes its workflow. Section 2.1 provides a detailed description and process of quad-grid coding representation of trajectories, and Section 2.2 introduces the main principles of QuadGridSIM. Section 3 develops experiments using real-world taxi trajectory data and simulates the data to verify our proposed algorithm. Finally, we conclude the article and outline further works in Section 4.

2 | PROPOSED METHOD—QUADGRIDSIM

The architecture of the quadrilateral grid-based method for trajectory similarity measure (QuadGridSIM) is shown in Figure 3. QuadGridSIM conducts the trajectory similarity measure in two main phases: (1) the quad-grid coding representation of the trajectory; (2) the definition of the trajectory similarity measure. First, a one-dimensional representation of trajectory data is encoded based on the quadrilateral grids to address the issues of two-dimensional vector representation such as inconsistent sampling rates and low computing efficiency (Section 2.1). In other words, the original trajectory data (i.e., T_1, T_2, ..., T_n) based on vector representation are converted to trajectory codes (i.e., S_1, S_2, ..., S_n). Second, in contrast to the high computational complexity of current similarity measures under vector representation, we construct a novel and low-complexity trajectory similarity measure

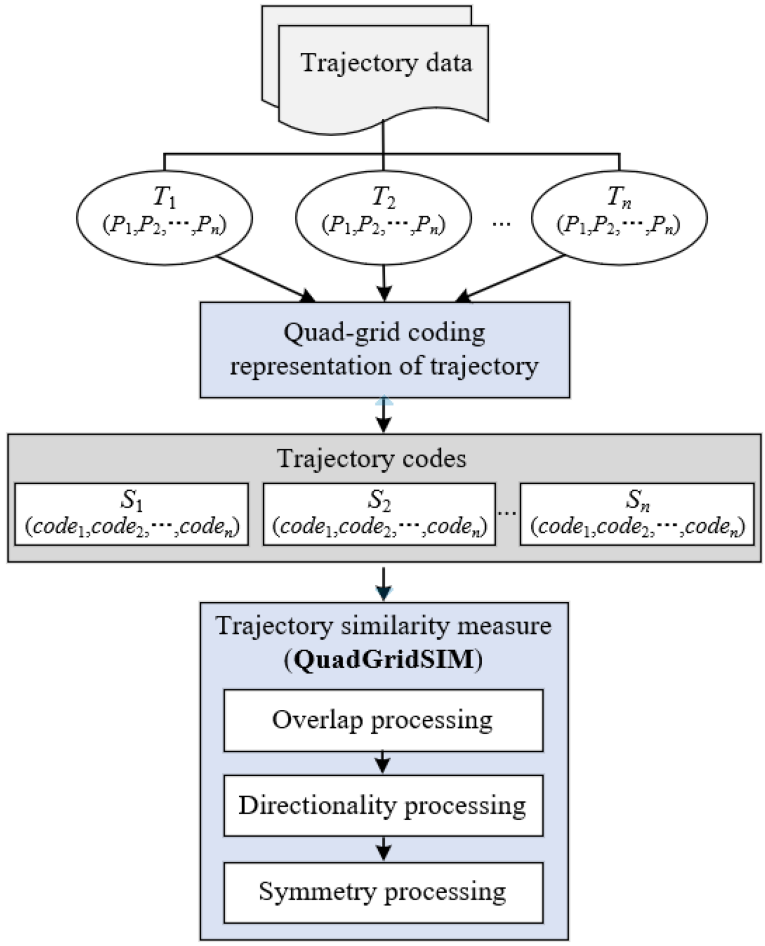

FIGURE 3 Overview of the architecture of QuadGridSIM.

(QuadGridSIM) for trajectory codes. The mathematical properties such as overlap processing, directionality processing, and symmetry processing of QuadGridSIM are described and proved in detail in Section 2.2.

2.1 | Quad-grid coding representation of trajectory

The grid-coding representation of trajectories consists of two parts: (1) the quad-grid subdivision and encoding in geographic space and (2) a multiscale code representation of trajectory. The former describes how to encode the spatial position on the Earth's surface, and the latter explains the principle of mapping trajectories to the quadrilateral grids and encoding them at various granularities.

2.1.1 | Quad-grid subdivision and encoding

The quad-grid system is built upon the quadtree structure, a well-established data structure known for its outstanding compatibility with vector coordinates, mature indexing mechanism, and simple spatial analysis (Bowater & Stefanakis, 2020; Gibb, 2016; Sun et al., 2009). In this system, the size of the parent grid cell is twice that of the child grid cell, ensuring a smoother spatial scale transition between adjacent levels (Qian et al., 2019). Figure 4 illustrates the basic principles of the quad-grid subdivision and encoding. First, the study area is defined and divided into four equal-size sub-areas, where each sub-area is further divided into four finer sub-areas recursively. Thus, the study area is recursively divided into $2^N \times 2^N$ grid cells, where N is the subdivision level. A higher subdivision level means finer spatial resolution and leads to a more precise position of the geographic object. Second, the spatial locations of the study area are encoded based on the Z-order curve (i.e., Morton code) instead of the vector coordinate representation. The Morton code of each grid cell is calculated by interleaving the binary representations of its *row* and *column* numbers. As shown in Figure 4, the four grid cells at each level are marked by north-west (NW), north-east (NE), south-west (SW), and south-east (SE) and assigned codes 0, 1, 2, and 3, respectively. Connecting the assigned codes 0, 1, 2, and 3 in the NW-NE–SW-SE sequence generates the recursively Z-order curve. The row and column numbers i and j of P are 6 (110) and 2 (010), and interleaving the binary coordinate values produces quaternary (or binary) grid codes, that is, the code of the point P is 221 (101100). Figure 5 shows an example that the study area (116.015625° E,

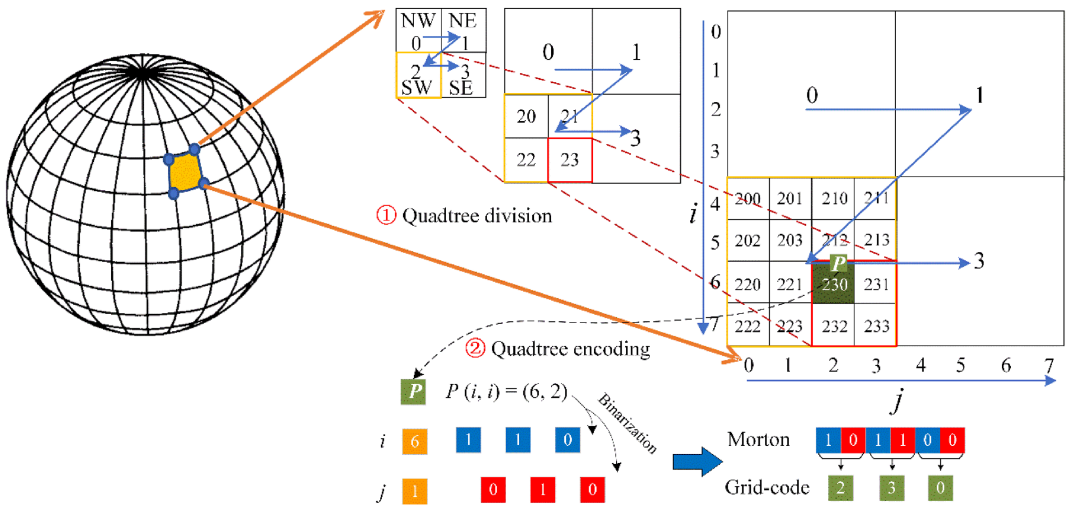

FIGURE 4 Quad-grid subdivision and encoding.

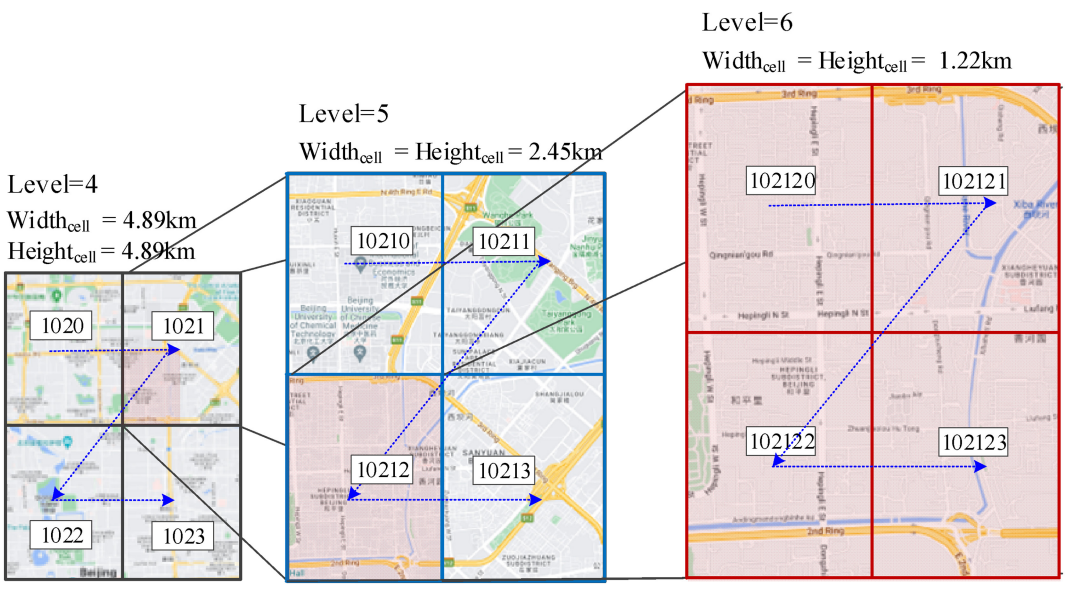

FIGURE 5 Example of quad-grid subdivision and encoding in the study area at Levels 4–6.

39.375° N, 116.71875° E, and 40.078125° N) has been encoded in the quad-grid code. It shows that the quadrilateral grid used in this study has a continuous spatial hierarchy with a twofold relationship regarding grid cell sizes between adjacent levels.

2.1.2 | Multiscale coding representation of trajectory

Similarity evaluation of vector-based trajectories with a two-dimensional data structure usually has a high computational complexity of $O(n \times m)$. This study, however, employs grid encoding (a raster data model) to map trajectories onto quadrilateral grids. This approach supports spatial trajectory encoding without the cost of high computational complexity. Trajectory data are represented as a sequence of time-ordered spatial grid codes ($code_1$, $code_2$, $code_3$, ..., $code_n$), which can facilitate large-scale trajectory data management and analysis (Definition 1).

> **Definition 1.** Trajectory codes. Under the geographic grid representation, a trajectory T is represented as a sequence of time-ordered spatial grid codes $S=(code_1, code_2, code_3, ..., code_n)$, where each code represents the location of a consecutive trajectory point in space.

The process of multiscale encoding of trajectories is illustrated in Figure 6. Initially, trajectory data can be mapped onto various grid levels for grid-based encoding. Subsequently, the suitable coding level is determined based on the spatial resolution requirements of diverse application scenarios, ensuring a balance between representation accuracy and data volume. As shown in Figure 6, a higher grid level corresponds to fine-grained trajectory grid codes and a larger quantity of trajectory codes. The trajectory code for a trajectory point is calculated using the row and column number of the corresponding grid cell, and the size of the grid cell is determined using Equation (1).

$$\begin{cases} \Delta L = (L - L_{\min})/2^N \\ \Delta B = (B - B_{\min})/2^N \end{cases} \quad (1)$$

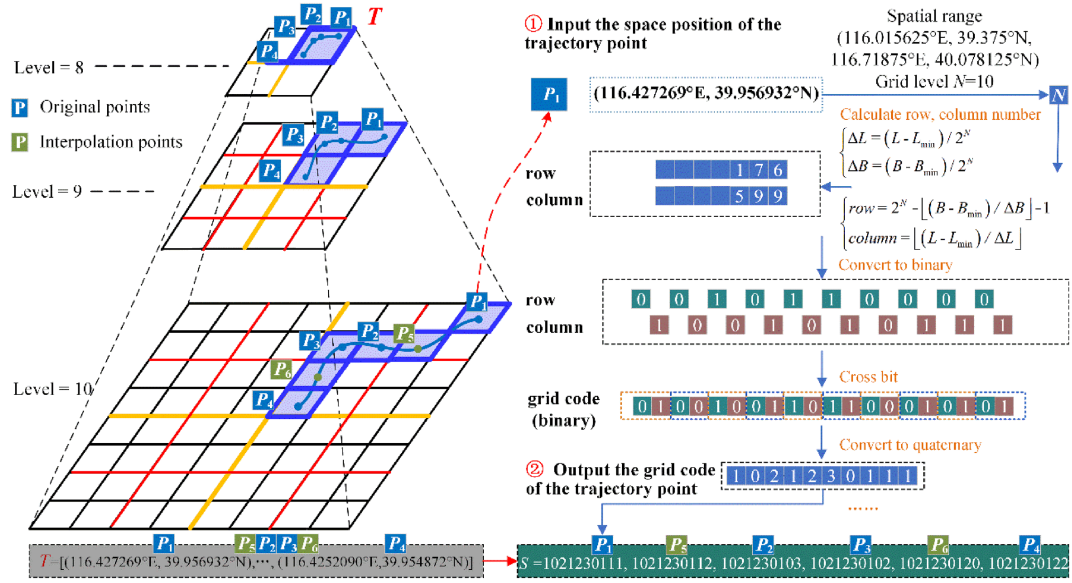

FIGURE 6 Pipeline of multiscale trajectory encoding.

where N is the grid level of the trajectory matched, L_{min} and B_{min} are the minimum longitude and latitude of the study area, and L and B are the longitude and latitude of the trajectory point to be encoded, respectively. The row and column numbers of the grid cell where the trajectory point is matched can be calculated using Equation (2).

$$\begin{cases} row = 2^N - \lfloor (B - B_{min})/\Delta B \rfloor - 1 \\ column = \lfloor (L - L_{min})/\Delta L \rfloor \end{cases} \tag{2}$$

Then, the code for each trajectory point is obtained by converting the row and column number of the corresponding grid cell into the binary format and applying separate cross-coding based on Morton's principle. Finally, to address the impact of different sampling rates on similarity measures, Equation (3) is used to interpolate the trajectory points (e.g., points P_5 and P_6 in Figure 6) and obtain the consecutive trajectory codes.

$$\begin{cases} X = X_{\text{Cell centerline}} \\ Y = \dfrac{Y_2 - Y_1}{X_2 - X_1}(X - X_1) + Y_1 \end{cases} \tag{3}$$

where $X_{\text{Cell centerline}}$ is the longitude of the centreline of a cell to be interpolated and (X_1, Y_1), (X_2, Y_2) are the coordinates of two adjacent points making up the trajectory. As a result, trajectories with varying sampling rates can be uniformly represented as a continuous set of trajectory codes.

Example of coding representation of trajectory

Figure 6 demonstrates an example of multiscale trajectory representation by the steps explained above, where L_{min}, B_{min}, L_{max}, and B_{max} of the area of interest are 116.015625° E, 39.375° N, 116.71875° E, and 40.078125° N, respectively. First, the trajectory T is mapped to the quadrilateral grids at Level 10. Then, the row and column numbers of the grid cell representing the first point P_1 are 176 and 599, respectively, according to Equations (1) and (2). Finally, the code of point P_1 is converted to 1021230111 by cross-coding the binary representation of its row and

column numbers. The trajectory is interpolated by adding P_5 and P_6 using Equation (3) when mapping trajectory T at Level 10 to ensure directionality in the trajectory grid representation and robustness to the variability in the sampling rate (Figure 6). Finally, the grid code of each trajectory point is calculated to obtain the code of the trajectory T, which transforms the vector-represented trajectory T=((116.427269° E, 39.956932° N), ..., (116.4252090° E, 39.954872° N)) to the trajectory code S=(1021230111, 1021230112, 1021230103, 1021230102, 1021230120, 1021230122).

2.2 | Similarity measure for trajectory codes

In this section, we propose QuadGridSIM, a quadrilateral grid-based method for the trajectory similarity measure. Two trajectories with high overlap in space and the same direction are viewed as having high similarity. Specifically, QuadGridSIM evaluates trajectory similarity from three perspectives:

1. *Overlap*. Determine the shared spatial path between two trajectories.
2. *Directionality*. Trajectories with the same direction of the shared spatial path exhibit higher similarity.
3. *Symmetry*. The similarity is the same between the two trajectories as either the reference or the target trajectory.

We first present the *overlap* processing. Intuitively, more identical codes in two trajectory codes mean a larger overlap between them. In other words, more overlap of the codes between two trajectories leads to more spatial adjacency and higher similarity of the two trajectories. Thus, it is straightforward to measure the overlap between two trajectories by identifying the intersection between their trajectory codes. As illustrated in Figure 7a, two trajectories, T_{A1} and T_{B1}, are considered similar if their trajectory codes are identical after mapping to the grid at the same level. There are special cases where two trajectories are spatially similar while coincidentally falling in different sets of trajectory grid cells (e.g., T_{A2} and T_{B2} in Figure 7b). In this case, the intersection of the two

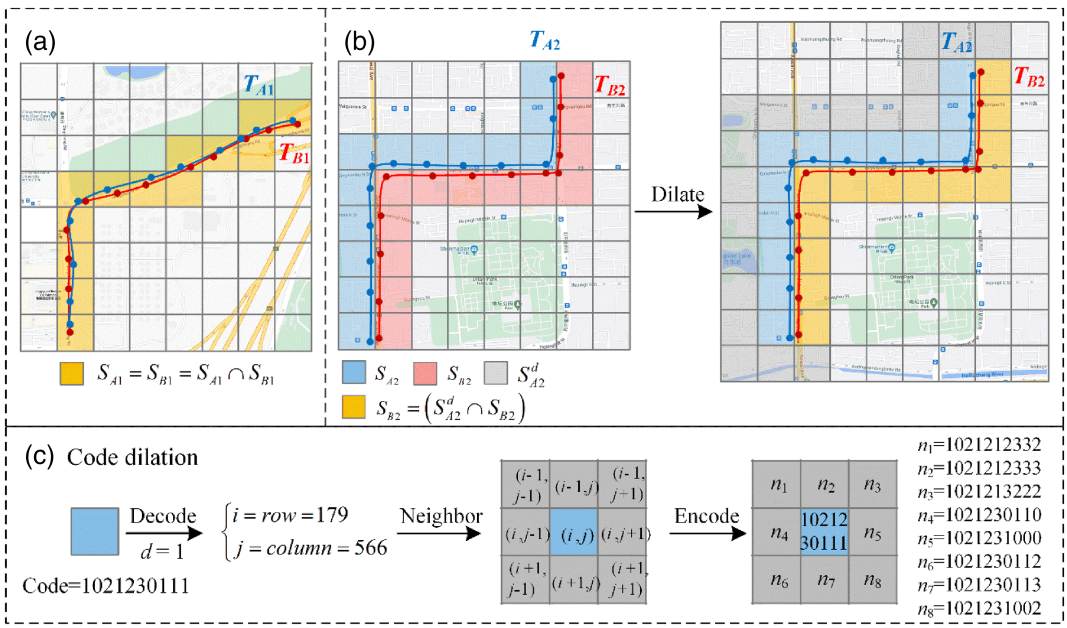

FIGURE 7 Overlap calculation for trajectory codes: (a) the typical case where two trajectories are spatially similar and their trajectory codes are identical; (b) the special case where two trajectories are spatially similar but happen to fall into different grid cells; and (c) code dilation operation used to extend the trajectory codes.

trajectory codes is an empty set, which causes low similarity between the two trajectories. To address this issue, we introduce the operator named code dilation to compute the adjacent grid codes. The basic idea of code dilation is similar to buffering in vector coordinates, aiming to capture similar trajectories in adjacent grids. The pseudo-code is shown in Algorithm 1. First, the target code c is decoded to derive the grid's *row* and *column* numbers (line 1), a process that inversely interleaves Morton code. Specifically, the target code is converted to binary, and its bits are separated into dimension-specific groups to generate the corresponding *row* and *column* numbers. Then, the row and column numbers of adjacent grid cells are calculated by adding or subtracting the expansion radius d (lines 2–6). Finally, adjacent grid codes c^d are obtained by encoding these adjacent grid cells.

Algorithm 1: Code dilation

Input: Trajectory code c, radius d
Output: Adjacent trajectory code c^d
1 $row, column \leftarrow decode\ (c)$
2 **for** each m in range $(-d, d)$ **do**
3 **for** each n in range $(-d, d)$ **do**
4 c^d.add $(encode\ (row+m, column+n))$
5 **end for**
6 **end for**

In Figure 7c, an example of code dilation is presented with an expansion radius of $d=1$, meaning the eight adjacent grid codes are calculated. First, the target grid code 1021230111 is decoded to obtain the row ($i=179$) and column ($j=566$) numbers of the corresponding grid cell. Then, the row and column numbers of the eight adjacent grid cells are calculated. For instance, for the bottom-right grid cell, the row and column numbers are computed as ($i+1=180, j+1=567$). Finally, the eight adjacent grid codes (n_1–n_8) are generated through grid encoding.

Using code dilation, the target trajectory T_{A2} can obtain its extended codes S^d_{A2}, which can realize the overlap calculation between two trajectories spatially close while not in the same grid cells. In summary, we define the *overlap* in Definition 2, and Algorithm 2 shows the pseudocode of *overlap*.

Definition 2. Overlap. For two trajectory code sets S_A and S_B, give a trajectory code $c_k \in S_A$, the code c_k is dilated to c^d_k, the function $Overlap\ (c^d_k, S_B) \rightarrow O^k_c$, $S_A \times S_B \rightarrow O_S$ returns a list O_S with all codes $c_l \in S_B$ such that c_l in c^d_k, ordered by the sequence of S_A.

It is not sufficient to rely on *overlap* alone as a measure of the similarity between two trajectories, since *overlap* only captures their spatial proximity and neglects their directionality. For example, as shown in Figure 1b, trajectories $T_{O'}$ and $T_{B'}$ are spatially close to each other but move forward in opposite directions. If the similarity of the two trajectories is measured solely by overlap, the calculated similarity values by methods such as Hausdorff and SSPD are 1 (indicating perfect similarity, i.e., 100% overlap), which is not true considering the directionality of the trajectories. Therefore, we refer to the continuity of the UMS and incorporate it into QuadGridSIM to correct the overlap between two trajectories S_A and S_B by considering their directional properties. Specifically, we need to determine the first overlap position for each O^k_c, as the overlap O^k_c between the c^d_k and trajectory S_B may contain several codes. The definition of the first overlap position is given in Definition 3. The first overlap position of O^k_c is defined as the smallest position that is greater than or equal to the first overlap position of the previous O^{k-1}_c. Figure 8 presents an example that computes the first overlap position between trajectories S_{A2} and S_{B2}. In the initial step, Figure 8a shows the overlap O^1_c (two yellow grid cells) between the dilation c^1_1 ($d=1$) and trajectory S_{B2}. O^1_c contains the first two codes of the trajectory S_{B2}, yielding the positions of O^1_c as [1,2]. As per Definition 3, *first*

Algorithm 2: Overlap

Input: Trajectory codes S_A and S_B, level N, radius d
Output: *Overlap* between S_A and S_B, O_S

$H \leftarrow 2^N \times 2^N$ array of empty lists
for c_l in S_B **do**
    $row, column \leftarrow decode\ (c_l)$
    $H[row][column]$.add (c_l)
end for
for c_k in S_A **do**
    $c_k^d \leftarrow Code\ dilation\ (c_k)$
    for each *code* in c_k^d **do**
        $i, j \leftarrow decode\ (code)$
        if $H[i][j]$ **is not** Null **then** // $O(1)$ time expected
            O_c^k.add $(H[i][j])$
        end if
    end for
    O_S.add (O_c^k)
end for

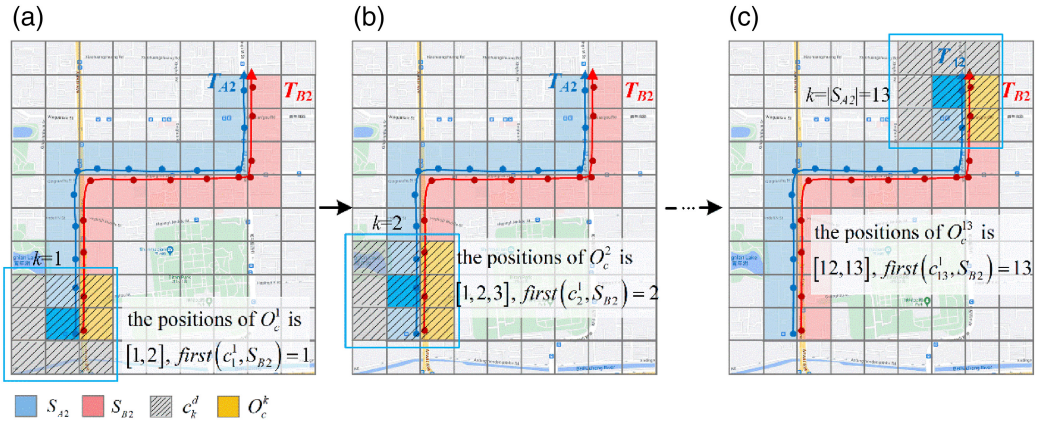

FIGURE 8 First overlap position calculation for trajectory codes with dilation radius $d=1$: (a) the positions of overlap O_c^1 for the first code ($k=1$) of trajectory S_{A2}; (b) the positions of overlap O_c^2 for the second code ($k=2$) of trajectory S_{A2}; and (c) the positions of overlap O_c^{13} for the last code ($k=|S_{A2}|=13$) of trajectory S_{A2}.

(c_1^1, S_{B2}) identifies the lowest value from the set [1,2] when $k=1$, that is, *first* $(c_1^1, S_{B2})=1$. Upon proceeding to $k=2$, as delineated in Figure 8b, the positions of overlap O_c^2 between c_2^1 and trajectory codes S_{B2} are [1, 2, 3]. Accordingly, *first* (c_2^1, S_{B2}) is the lowest value from [1, 2, 3] that is equivalent to or surpasses the value of *first* $(c_1^1, S_{B2})=1$, thus *first* $(c_2^1, S_{B2})=2$. Similarly, the first overlap position between the last code dilation of trajectory S_{A2} and S_{B2} is *first* $(c_{13}^1, S_{B2})=13$ (Figure 8c).

Definition 3. First Overlap Position. For two trajectory code sets S_A and S_B, give a trajectory code $c_k \in S_A$, the first position of overlap between c_k^d and S_B is given by the function *first* (c_k^d, S_B) according to the following conditions:

1. **if** $k = 1 \wedge \exists\, c_l \in O_c^k$, **the lowest value of position** l **is returned;**
2. **if** $k > 1 \wedge \exists\, c_l \in O_c^k$, **the lowest value of position** l **such that** $l \geq$ *first* (c_{k-1}^d, S_B) **is returned;**
3. **otherwise, the value −1 is returned;**

Definition 4 gives the *directionality* of the overlap of two trajectories based on time-series property, and Algorithm 3 shows the pseudocode of *directionality*. Under the directional constraint, the similarity is only considered high when two trajectories have high overlap in space and high consistency in the movement direction.

Definition 4. Directionality. For two trajectory codes S_A and S_B, let the first overlap position set of S_A be $U = (first\ (c_1^d, S_B), first\ (c_2^d, S_B), \ldots, first\ (c_n^d, S_B))$, then directional overlap $Đ = \{u_k \mid u_k \in U, \text{ and } u_k \neq -1\}$.

Algorithm 3: Directionality

```
Input: Overlap between S_A and S_B, O_S
Output: Directional overlap Đ between S_A and S_B
for each O_c^k in O_S do
    if O_c^k is the first element of O_S then
        U.add (min (position))
    else
        for each position in O_c^k do
            if position >= U [-1] then
                temp.add (position)
            else
                temp.add (-1)
            end if
        end for
        U.add (min (temp))
    end if
end for
for each u in U do
    if u ≠ −1 then
        Đ.add (u)
    end if
end for
```

In summary, we obtain directional similar segments of the two trajectory codes S_A and S_B, namely the directional overlap $Đ$. For example, in Figure 8, the directional overlap $Đ_{AB2} = [1, 2, \ldots, 13]$ is evidenced between trajectories S_{A2} and S_{B2}. To further quantify the similarity, we define the *similarity* of the two trajectories in Definition 5. This function is implemented by dividing the length of the directional overlap $Đ$ by the average length of their respective trajectory codes, as expressed in Equation (4).

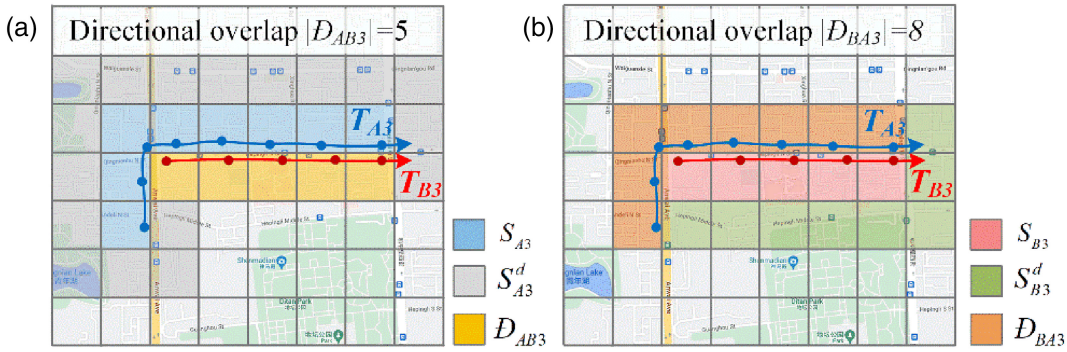

FIGURE 9 The difference in directional overlap Đ for dilation with (a) S_{A3} and (b) S_{B3}.

Definition 5. Similarity. Using two trajectory codes S_A and S_B, we define the *SIM* (S_A, S_B) as follows:

$$SIM(S_A, S_B) = \frac{2 \times |Đ|}{|S_A| + |S_B|} \tag{4}$$

where Đ is the directional overlap sequence of trajectory codes S_A and S_B, $|S_A|$ and $|S_B|$ are the numbers of codes on trajectories S_A and S_B, respectively.

While the function *SIM* is capable of measuring the similarity of two trajectories, it lacks symmetry, resulting in *SIM* (S_A, S_B) ≠ *SIM* (S_B, S_A). This disparity arises due to the dependence of the directional overlap Đ on the trajectory chosen for dilation. For example, as shown in Figure 9, dilation using S_{A3} yields the directional overlap $|Đ_{AB3}|=|S_{B3}|=5$ for S_{A3} and S_{B3}. However, this changes when dilating with S_{B3}, leading to $|Đ_{BA3}|=|S_{A3}|=8$.

To address this inherent asymmetry, we adopt a symmetrization strategy outlined in Definition 6. This strategy entails separately computing *SIM* (S_A, S_B) and *SIM* (S_B, S_A), followed by averaging them, yielding the final similarity measure between the trajectories in Equation (5). Finally, the formal definition of QuadGridSIM is established through Equation (5), supported by the pseudocode detailed in Algorithm 4. This symmetrization process rectifies the impact of trajectory selection during dilation, ensuring a balanced and unbiased similarity assessment.

Definition 6. Symmetry. Symmetrize the similarity between two trajectory codes S_A and S_B such that QuadGridSIM (S_A, S_B) = QuadGridSIM (S_B, S_A).

$$\text{QuadGridSIM}(S_A, S_B) = \frac{(SIM(S_A, S_B) + SIM(S_B, S_A))}{2} \tag{5}$$

Algorithm 4: QuadGridSIM

Input: Trajectory codes S_A and S_B, level N, radius d
Output: Similarity between T_A and T_B

1 $Os\_AB \leftarrow Overlap\ (S_A, S_B, N, d)$
2 $Os\_BA \leftarrow Overlap\ (S_B, S_A, N, d)$
3 $Đ_{AB} \leftarrow Directionality\ (Os\_AB)$
4 $Đ_{BA} \leftarrow Directionality\ (Os\_BA)$
5 $Similarity \leftarrow (|Đ_{AB}|+|Đ_{BA}|)/(|S_A|+|S_B|)$

The range of similarity values computed using QuadGridSIM falls within the interval [0, 1]. A similarity value of 0 is obtained when trajectories S_A and S_B exhibit no overlap. Conversely, a similarity value of 1 is attained when trajectory S_A perfectly overlaps with trajectory S_B, resulting in $Đ_{AB}$ and $Đ_{BA}$ being equal to the lengths of trajectories S_B and S_A, respectively. Additionally, we further define the distance between trajectories S_A and S_B in Equation (6):

$$D(S_A, S_B) = 1 - QuadGridSIM(S_A, S_B) \tag{6}$$

We conclude that the overall complexity of QuadGridSIM (S_A, S_B) is $O(N_A + N_B)$, as linear complexity, where N_A and N_B are the numbers of codes for trajectory codes S_A and S_B, respectively. QuadGridSIM holds the similarity properties of *reflexivity* and *nonnegativity*, besides *directionality* and *symmetry*. Give any two trajectories T_A and T_B with the number of points greater than or equal to 2:

Lemma 1. Reflexivity: *if Trajectory* T_A = *Trajectory* T_B, *then QuadGridSIM* (S_A, S_B) = *1*.

Proof: Direct from Equations (4) and (5).

Lemma 2. Nonnegativity: *QuadGridSIM* (S_A, S_B) ≥ *0*.

Proof: Known trajectories T_A and T_B with the number of points greater than or equal to 2, thus $|S_A| > 0$, $|S_B| > 0$, directional overlap $Đ \geq 0$. From Equations (4) and (5), QuadGridSIM (S_A, S_B) ≥ 0.

3 | EXPERIMENTS AND ANALYSIS

3.1 | Experiment dataset

This study uses real-world and simulated taxi trajectory data to evaluate the performance of QuadGridSIM. The real-world trajectory data were collected by approximately 20,000 taxis in Beijing, China during 1 week in 2012 (Figure 10). Based on different experimental scenarios, three datasets with varying sizes were processed (Table 1). Among them, TD_1 contains 1 million trajectories, including 46,561,710 trajectory points, which are used to evaluate the performance of the trajectory similarity measures on large-scale data. The length of each trajectory ranges from 5 to 70 km, with attributes including license plate number (encrypted for privacy protection), positioning time, longitude, latitude, speed, and direction (Table 2). According to the positioning accuracy of the trajectory data and their geographic coverage, the study area (116.015625° E, 39.375° N, 116.71875° E, 40.078125° N) is modeled by the quadrilateral grid at 10 levels where the grid cell size is about 60 m at the finest level.

3.2 | Experiment environment

The computing environment is a Dell computer with a four-core Intel i5-4210H 2.90 GHz processor, 16 GB of memory, 500 GB HDD, and Windows 10 64-bit operating system. All algorithms were implemented in Python 3.6, and the MySQL database was used to store the trajectory data.

3.3 | Baseline methods

In our experiment, we compared QuadGridSIM with seven other commonly used algorithms including ED, LCSS, DTW, Hausdorff, SSPD, UMS, and FastDTW, to comprehensively evaluate our proposed algorithm.

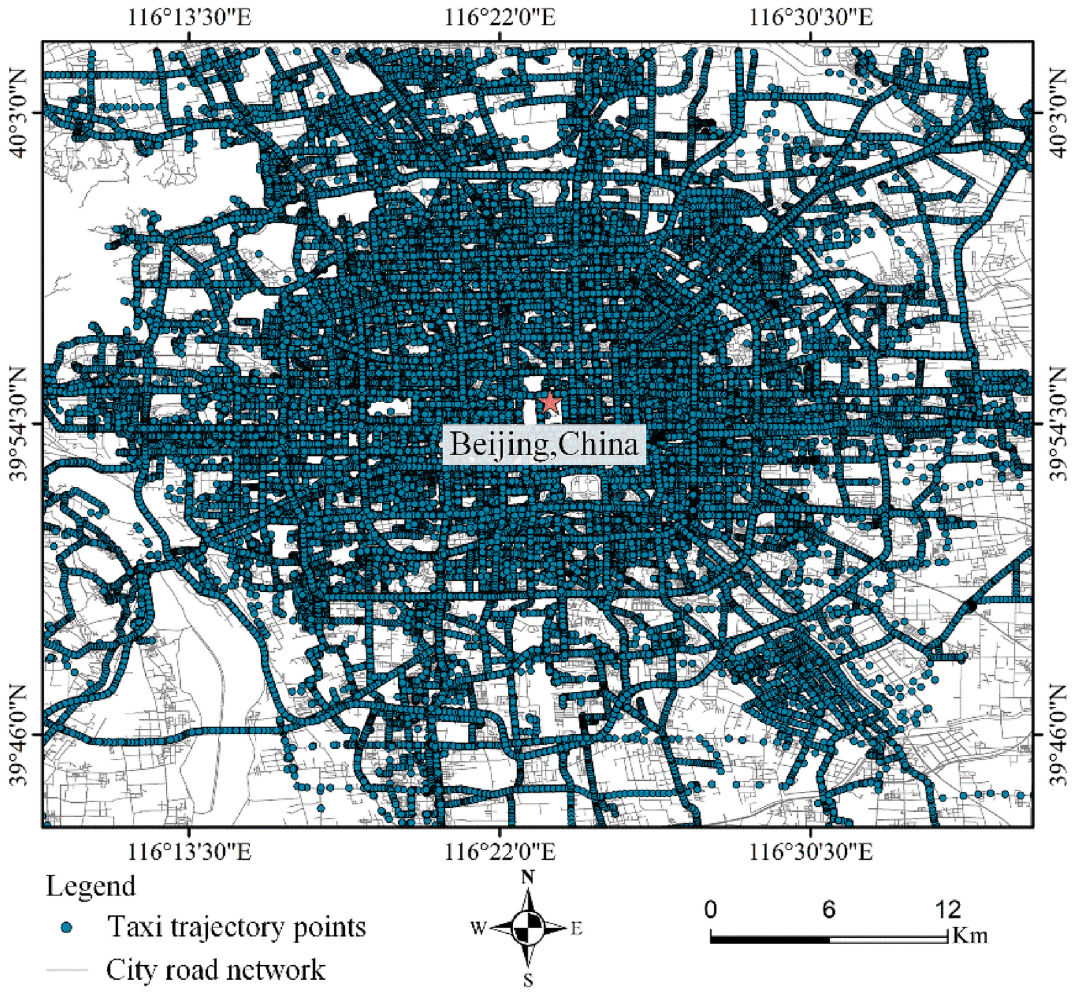

FIGURE 10 Spatial distribution of the experimental trajectory data in the study area.

TABLE 1 Summary of experimental datasets.

Trajectory dataset	No. of trajectories	No. of points	Avg. sampling interval/s
TD_1	1,000,000	46,561,710	11
TD_2	100,000	5,584,305	13
TD_3	283	32,637	10

TABLE 2 Example of taxi trajectory data.

Taxi ID	Time	Longitude	Latitude	Speed (km/h)	Direction
111402	2012-11-01 07-07-06	116.309166	39.963924	40.51	248
111402	2012-11-01 07-07-16	116.307999	39.963455	40.05	247
		...			
163408	2012-11-07 12-06-57	116.524978	39.953712	52.46	12

There are three reasons for adopting these algorithms as baselines for comparison. First, they have high computational performance, where ED and FastDTW have computational complexity as low as ($O(n)$). Second, they are widely used in various fields, especially LCSS, Hausdorff, and DTW. Third, they are optimized in various aspects. Particularly, UMS is a recently proposed and advanced method using an elliptical representation of trajectories, which makes it more robust when dealing with movement uncertainty. Furthermore, SSPD is optimized for Hausdorff in terms of robustness to the sampling rate. Comparison with other trajectory similarity measures has been carried out in other related work (Mariescu-Istodor & Fränti, 2017; Su et al., 2020; Toohey & Duckham, 2015). Open-source codes for these baseline methods are available in the previous research (Besse et al., 2016; Furtado et al., 2018).

3.4 | Effectiveness evaluation of QuadGridSIM

The classic retrieval metric *precision at recall* (Baeza-Yates & Ribeiro-Neto, 1999; Furtado et al., 2018) was utilized to verify the effectiveness of QuadGridSIM in measuring trajectory similarity. The basic idea of this metric is that a higher precision in querying the K most similar (top K) trajectories at different recall levels implies better effectiveness of the similarity method. Initially, (1) the crowded route is selected and all trajectories within the route are manually identified as the ground truth ϑ. Generally, the trajectories in ϑ have a high similarity to each other. As shown in Figure 11, this experiment selected three commuting routes (ϑ_1, ϑ_2, and ϑ_3) in Beijing, containing 16, 20, and 24 trajectories, respectively. The selection of these routes is based on three criteria, including a crowded path, different sampling rates, and opposite motion directions, to verify the effectiveness of similarity measures from multiple aspects. Then,

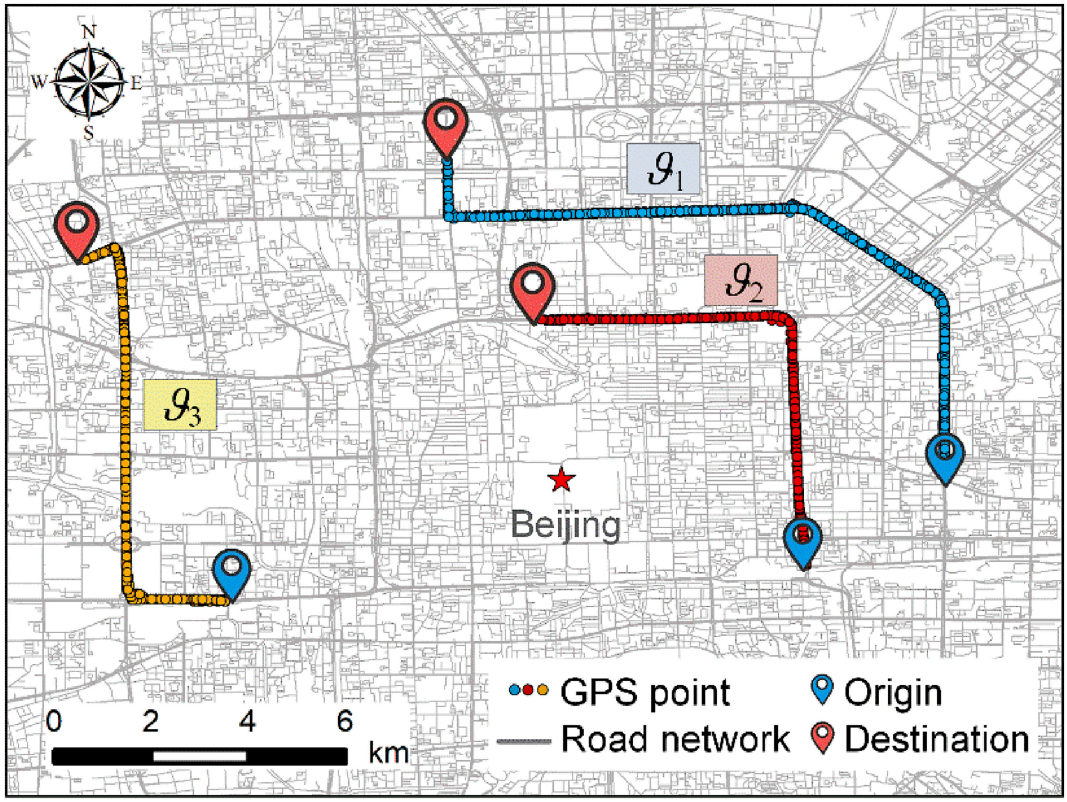

FIGURE 11 Trajectories in the ground truth ϑ_1, ϑ_2, and ϑ_3.

(2) the dataset TD_1 is chosen as the candidate trajectory dataset, and the similarity is calculated between all trajectories in the ground truth ϑ_k with each trajectory in the entire TD_1. Next, (3) the trajectories are ranked based on their similarity, and the top K similar trajectories are returned, and (4) the precision is computed for size (ϑ_k) levels of recall. Generally, a "perfect" method is expected to retrieve top K similar trajectories that should belong to ϑ_k regardless of the recall levels. Consequently, when the recall level is 1 (i.e., all similar trajectories inside ϑ_k are recalled), a perfect method would retrieve top K similar trajectories that belong exclusively to ϑ_k (i.e., size (ϑ_k)=K and precision is 1). Finally, (5) two accuracy metrics, MAP (mean average precision) and BEP (break-even point), are chosen to evaluate the overall performance of QuadGridSIM and other baseline methods. MAP measures the average precision across all recall levels, while BEP is the intersection point of the precision and recall curve. MAP and BEP take values in the range of [0,1], and higher MAP and BEP indicate higher effectiveness of the similarity measure.

The experimental results are presented in Table 3, indicating that QuadGridSIM and UMS outperform all other algorithms. DTW, Hausdorff, and SSPD had the second-best performance, followed by LCSS and ED. The ability to handle the complexities of realistic trajectories is the key factor in determining the performance of each method. The QuadGridSIM and UMS algorithms possess the essential properties of a similarity method, such as spatial overlap, directionality, and symmetry, and are also capable of handling different sampling rates (more details are provided in Section 3.5). DTW is highly sensitive to the sampling rate, and the results of trajectory distance calculation can vary significantly based on the number of trajectory points in the same path. The main limitation of Hausdorff and SSPD is their exclusive focus on the spatial overlap of trajectories, neglecting the directionality of trajectories. Consequently, they are unable to distinguish between similar trajectories that overlap but move in opposite directions. LCSS focuses on local similarity, particularly for small trajectories, and can result in the highest similarity in cases where the query trajectory is a sub-trajectory of the candidate trajectory, resulting in a maximum similarity score of 1 without penalizing dissimilar parts. Moreover, ED lacks dynamic time-warping characteristics, resulting in its extreme sensitivity to the trajectory's sampling rate and incapable to accurately measure the similarity between two trajectories.

3.5 | Robustness to sampling rate evaluation of QuadGridSIM

This section evaluates the robustness of QuadGridSIM and the baseline methods when dealing with different sampling rates. The experimental dataset used was TD_3, which contained a total of 283 trajectories with a sampling interval of approximately 10 s. All trajectories are resampled at intervals of 20, 30, 40, 50, 60, 70, 80, 100, 120, and 150 s. The trajectory data with a sampling rate of 60 s were selected as the baseline for calculating similarity with trajectories at other sampling rates. Thus, the trajectories with sampling rates of 20, 30, 40, and 50 s, relative to 60 s, have an increase rate of 200, 100, 50, and 20%, respectively. Similarly, the trajectories with sampling rates of 70, 80, 100, 120, and 150 s, relative to 60 s, have a missing rate of 14, 25, 40, 50, and 60%, respectively. DTW, Hausdorff, SSPD,

TABLE 3 Mean average precision (MAP) and break-even point (BEP) for all the methods.

	ϑ_1		ϑ_2		ϑ_2	
	MAP	BEP	MAP	BEP	MAP	BEP
QuadGridSIM	0.73	0.73	0.82	0.82	0.83	0.81
UMS	0.75	0.69	0.8	0.75	0.85	0.76
LCSS	0.06	0.09	0.01	0.02	0.07	0.12
DTW	0.42	0.39	0.47	0.46	0.39	0.42
Hausdorff	0.37	0.37	0.42	0.42	0.45	0.45
SSPD	0.34	0.37	0.4	0.41	0.43	0.46
ED	0.02	0.07	0.06	0.09	0.11	0.12

and ED were normalized by the maximum distance so that the interval for distance calculation was [0,1]. The distance threshold ϵ for LCSS was set to 200 m and the distance was normalized based on the length of the shorter trajectory. All methods are evaluated based on their similarity, which is computed as 1 minus the normalized distance. In theory, if a method is not vulnerable to various sampling rates, the similarity between the 60 s trajectory and the other sampled trajectories should remain 1 (or the distance should remain 0).

Figure 12 shows the influence of the sampling rate (increasing and missing ratio) on various similarity measures: (1) For the increasing ratio (Figure 12a), QuadGridSIM outperforms all other methods due to continuous coding representation, which makes it highly resilient to varying sampling rates. UMS, LCSS, Hausdorff, and SSPD performed well in the experiments. LCSS is effective in handling increasing sampling ratios, as the similarity between the query trajectory sampled at the 60 s and the denser trajectory approaches 1. This is because as the sampling rate of the trajectory increases, the 60-s query trajectory becomes a sub-trajectory of the higher sampling rate trajectory and is completely covered. Hausdorff and SSPD generated smaller distances as the spatial interval between points became smaller. DTW and ED are most affected by the sampling rates because they compute the cumulative sum of the distances between all points, causing the trajectory similarity to decrease as the sampling rate increases. (2) For the missing ratio, QuadGridSIM also exhibits higher robustness than other algorithms, and its effectiveness decreases relatively smoothly as the missing rate increases (Figure 12b). UMS shows the best stability because it uses an elliptical representation of trajectories, indicating that it is particularly robust to extremely sparse trajectories. The similarity error of SSPD changes more slowly than that of Hausdorff, indicating that SSPD is more robust in terms of sparse sampling rate. This is because SSPD improves upon Hausdorff by taking the mean of point-to-trajectory distance, rather than the maximum distance. ED, DTW, and Hausdorff change more dramatically as the missing rate increases, indicating that the robustness of these two measures is lower than that of QuadGridSIM, UMS, LCSS, and SSPD. Our results were in line with the research by Toohey and Duckham (2015), Mariescu-Istodor and Fränti (2017), and Gong et al. (2020).

3.6 | Performance evaluation of QuadGridSIM

In this section, we compared the performance between QuadGridSIM and baseline methods from the perspectives of trajectory lengths, dataset sizes, and grid levels. This experiment relies on a top-K similar trajectory query

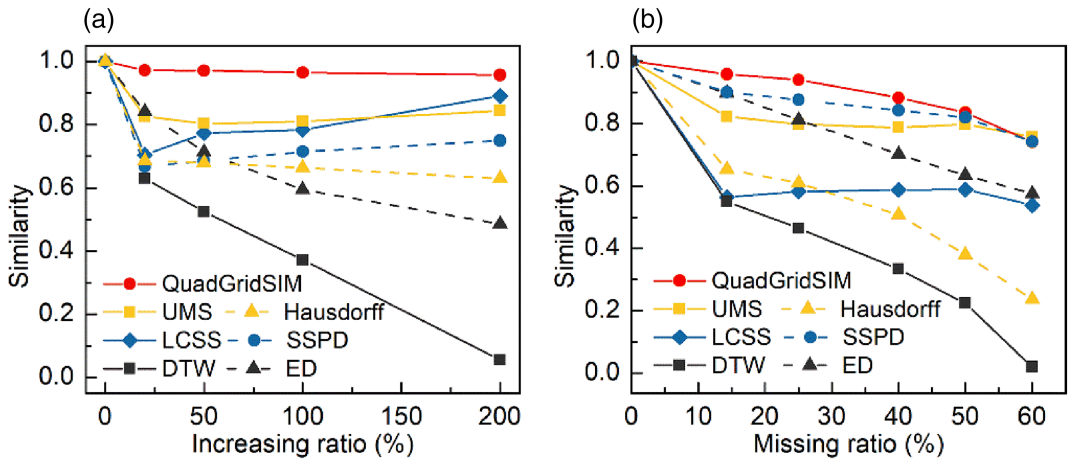

FIGURE 12 The influence of the sampling rate includes (a) increasing and (b) missing ratio of the trajectory on different similarity measures.

to evaluate the performance of all methods, which is a common and important application for similarity analysis. The top K trajectory similarity query is to search K trajectories that are most similar to the trajectory T in space based on the similarity measure in a dataset. Besides, in order to validate the performance of QuadGridSIM from different aspects, the trajectory similarity measures of baseline methods used at the refining stage include the methods with high computational complexity of $O(n \times m)$ (i.e., UMS, LCSS DTW, Hausdorff, and SSPD) and the computational efficient methods with complexity of $O(n)$ (i.e., ED and FastDTW). All methods use the Grid-cell (Gong et al., 2020) as the spatial index in the filtering stage.

3.6.1 | Performance comparison of various trajectory lengths

Figure 13 shows the computational time efficiency of QuadGridSIM, UMS, LCSS, DTW, Hausdorff, SSPD, ED, and FastDTW over different trajectory lengths for dataset TD_2, which is the timing of similar trajectories queries from 100 k trajectories. Overall, the time efficiency of QuadGridSIM is approximately one order of magnitude better than that of UMS, LCSS, DTW, SSPD, and Hausdorff. This is because QuadGridSIM is optimized with a computational complexity of $O(n)$, while other methods have a higher complexity of $O(n \times m)$. ED is approximately one order of magnitude more efficient than QuadGridSIM. As previously verified, however, it has the lowest effectiveness among the evaluated algorithms. Despite FastDTW being an improved version of DTW with a complexity of $O(n)$, its improvement in efficiency is not as significant as that of QuadGridSIM. This is due to that code calculations are generally more efficient than geographic operations with vector-based representation. Moreover, the time efficiency of QuadGridSIM is relatively stable and does not significantly deteriorate as the trajectory length increases, unlike the other baseline methods.

3.6.2 | Performance comparison of various dataset sizes

Figure 14 shows the computational time efficiency of the similarity query on different sizes of trajectory datasets. The time consumption of all algorithms increases as the number of trajectories increases. The UMS, LCSS, DTW,

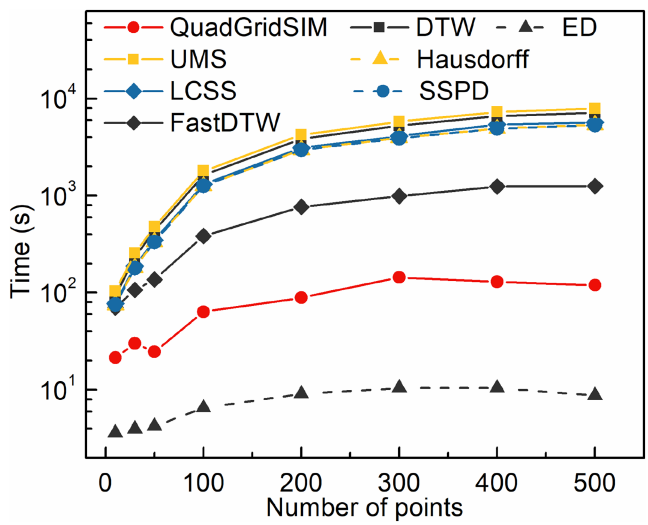

FIGURE 13 Time efficiency comparison of trajectory similarity queries among different methods based on 100 k trajectories.

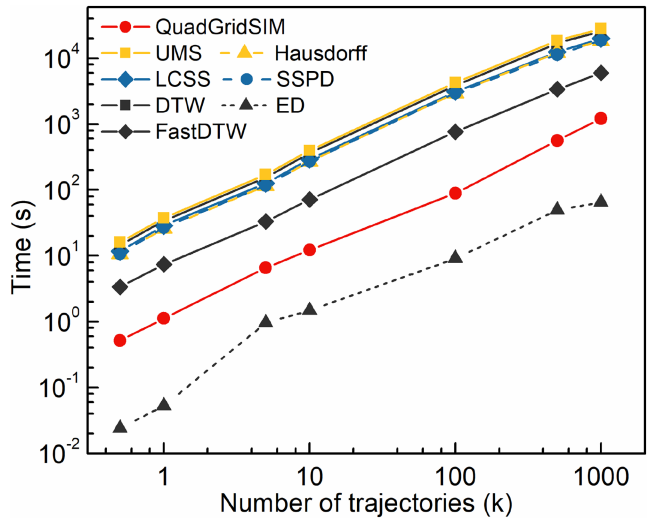

FIGURE 14 Time efficiency comparison of trajectory similarity queries across all methods based on different numbers of trajectories.

SSPD, and Hausdorff algorithm consumes the longest computational time, and the performance of FastDTW is approximately four times better than the five measures (UMS, LCSS, DTW, SSPD, and Hausdorff). Except for ED, QuadGridSIM demonstrates superior performance and is approximately one order of magnitude faster than the other algorithms, regardless of the dataset size.

3.6.3 | Performance comparison of various grid levels

Figure 15 shows the time cost of querying similar trajectories by QuadGridSIM based on the dataset TD_2 with 100 k trajectories encoded on different grid levels. The computational time of QuadGridSIM increases on finer grid levels because a finer grid means a greater amount of trajectory codes to be processed. Nevertheless, the increasing trend of time cost by QuadGridSIM is relatively stable, and the time cost is much less than the other tested similarity measures. QuadGridSIM shows different performance at different grid levels, where its computing efficiency is about two orders of magnitude higher than that of baseline methods at low grid levels (Levels 7–9), and one order of magnitude higher even at high grid levels (Levels 10–11). Therefore, the multiscale characteristics inherent in grid representation offer promising solutions for applications across various spatial scales.

4 | DISCUSSION AND CONCLUSION

Previously, diverse trajectory similarity measures have been developed and each method has profound characteristics in particular application scenarios. However, existing methods have difficulty in balancing efficiency and effectiveness. On one hand, their high computational complexity and low computational performance make it difficult to meet performance challenges when dealing with large-scale trajectory data. On the other hand, it is challenging to guarantee effectiveness properties such as directionality and robustness to the sampling rate. This study proposes QuadGridSIM, a quadrilateral grid-based method for trajectory similarity analysis, to provide a high-performance algorithm for trajectory data mining while ensuring effectiveness. Experimental results

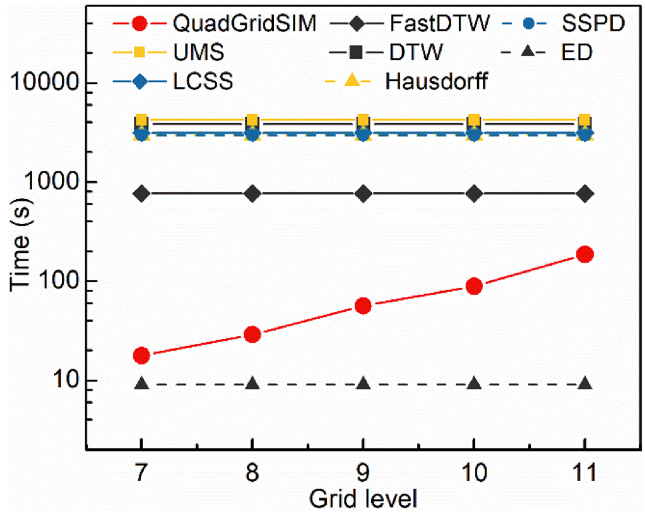

FIGURE 15 Time efficiency comparison of trajectory similarity queries across all methods at different grid levels.

TABLE 4 Characteristics of trajectory similarity measures.

Similarity measure	Time complexity	Different length	Directionality	Symmetry	Robust to sampling
QuadGridSIM	$O(n)$	✓	✓	✓	✓
UMS	$O(n \times m)$	✓	✓	✓	✓
LCSS	$O(n \times m)$	✓	✓	✓	✗
FastDTW	$O(n)$	✓	✓	✓	✗
DTW	$O(n \times m)$	✓	✓	✓	✗
SSPD	$O(n \times m)$	✓	✗	✓	✓
Hausdorff	$O(n \times m)$	✓	✗	✓	✗
ED	$O(n)$	✗	✓	✓	✗

indicated that QuadGridSIM had strong performance and robustness, providing superior solutions to the above challenges, and can be applied in application scenarios where high efficiency is in need. To make a direct comparison between QuadGridSIM and other commonly used similarity methods, we accounted for five characteristics: time complexity, different lengths, directionality, symmetry, and robustness to sampling rate variations. A comparison of characteristics has been summarized in Table 4.

Our results are concluded as follows.

1. QuadGridSIM provides a significant improvement in performance compared to classical methods. Overall, QuadGridSIM is 20–30 times more computationally efficient than UMS, LCSS, DTW, Hausdorff, and SSPD, and 6–9 times faster than FastDTW. The performance improvement of QuadGridSIM is due to our optimization in two aspects: simplifying the representation of trajectory data through grid coding, and reducing the computational complexity of the similarity measure from $O(n \times m)$ to $O(n)$. Therefore, QuadGridSIM has the potential to provide technical support for data mining and analysis with high temporal requirements in massive trajectory datasets.

2. QuadGridSIM exhibits effectiveness regarding similarity measures, including the ability to handle trajectories of different lengths, directionality, and symmetry, and is robust to the sampling rate. Unlike Hausdorff and SSPD, we have optimized the QuadGridSIM to recognize the direction of trajectory motion, ensuring that the similarity of trajectories in opposite directions is comprehensively considered. Additionally, during the trajectory encoding process, trajectories with different sampling rates are interpolated into spatially continuous trajectory codes (as shown in Figure 6, Level 10), which makes QuadGridSIM more robust to sampling rate than other methods such as LCSS, DTW, Hausdorff, and SSPD.
3. The multiscale encoding scheme empowers QuadGridSIM to excel in top-K similar trajectory querying and multisource trajectory mining. Initially, we employ a coarse-scale coding representation to effectively filter a candidate set of similar trajectories from the database, followed by the computation of trajectory similarities within this refined candidate set, under a fine-scale representation using QuadGridSIM. This "filter-refine" strategy, rooted in the multiscale property, significantly reduces unnecessary computational overhead and processing time in top-K similar trajectory querying. Moreover, it is crucial to acknowledge the significant variation in spatial resolution among multisource trajectories, attributable to differences in localization accuracy and sampling intervals. For instance, human walking trajectories typically exhibit a spatial resolution of 5 m, while bicycle trajectories encompass 20 m, and taxi trajectories extend to 100 m. QuadGridSIM's multiscale property harmonizes trajectory spatial representations at the appropriate grid level, thereby facilitating similarity analysis across multisource trajectory data.

Although QuadGridSIM has advantages in terms of robustness and performance, it consumes more storage space due to interpolation during the trajectory encoding, especially when the sampling rate is low and the moving speed is fast. Therefore, it is particularly critical to determine an appropriate grid level based on different application scenarios, sampling rates, and positioning accuracy to balance accuracy and data volume (efficiency). For example, a grid model with a cell size of 300–1000 m was adopted to represent global flight trajectories (Zheng et al., 2019), while a cell size of 20–60 m (road width) was used to analyze taxi trajectories within cities (Li, Liu, et al., 2022). In addition, QuadGridSIM does not have triangle inequality and identity. This is because QuadGridSIM, like LCSS and EDR, builds a buffer by a distance threshold of points (code dilation) and assumes that the points in this buffer satisfy the similarity metric.

Given the characteristics of QuadGridSIM outlined above, we can delineate its applicability and limitations in application scenarios. QuadGridSIM, in general, offers versatility to support a broad spectrum of applications, including (1) similar trip recommendations. This involves extracting similar trips based on taxi trajectories to provide recommendations for popular routes in an unfamiliar city. (2) Anomalous trajectory detection. QuadGridSIM can effectively detect anomalous trajectories that exhibit low similarity to common routes derived from historical taxi trajectories. This capability aids in mitigating fraudulent activities, such as intentional detours and overcharging. (3) Close contact identification in epidemics. It can also be employed to identify the trajectories of individuals at high risk of exposure to epidemic patients, facilitating measures to mitigate the spread of pandemics. However, QuadGridSIM predominantly emphasizes spatial similarity measures, without taking into account the temporal dimension. As a result, gsstSIM (Li, Liu, et al., 2022), which effectively measures both spatial and temporal similarity, outperforms QuadGridSIM in scenarios where time sensitivity is critical. For instance, while QuadGridSIM may excel in identifying the busiest routes based on floating car trajectories, it may not perform as effectively as gsstSIM in pinpointing the specific time of day when congestion typically occurs.

Additionally, QuadGridSIM enhances the functional capabilities of Discrete Global Grid Systems (DGGS), which are global-scale geographic grid models (Li, McGrath, et al., 2022; Mahdavi-Amiri et al., 2015; Robertson et al., 2020). Specifically, QuadGridSIM can seamlessly integrate with established DGGS, such as quadrilateral rHEALPix, particularly when dealing with global-scale trajectories, for example, global flight trajectories. Nevertheless, it should be noted that while we presented the principles of QuadGridSIM based on the

quadrilateral grid, it is not difficult to imagine that it can be transformed into triangular and hexagonal DGGS, such as Uber's H3 and DGGrid. Because the fundamental functions of *Directionality*, *Similarity*, *and Symmetry* remain almost unaffected by the geometry of the grid cells, only the trajectory encoding and *dilation* operation need to be changed. The generality of grid geometry simplifies the extension of QuadGridSIM and enhances its flexibility in applications. For example, when performing a similarity analysis of trajectories within the platform's preexisting hexagonal DGGS, such as Uber H3, QuadGridSIM can be seamlessly integrated, eliminating the need for quad-grid reconstruction.

Given the hot topics in trajectory mining, more research should be conducted in the future. For example, integrating temporal and semantic information will enrich the trajectory similarity measure, making it more comprehensive and intriguing. Moreover, as previously mentioned, the discreteness is one core advantage of geographic grid models. It is practical to incorporate the QuadGridSIM into a high-performance computing framework, such as GPU parallel computing or Spark distributed computing, to accelerate similarity queries and trajectory data mining. Additionally, the grid-coding representation of trajectories and the corresponding improved algorithms can provide a reference for other trajectory data mining and analysis methods. A trajectory computing and analysis framework based on geographic grid models is required to meet the ever-increasing demand for real-time or near real-time trajectory data applications in various fields.

ACKNOWLEDGMENTS

This research was financially supported by the National Natural Science Foundation of China (Grant No. 41971355) and the Yueqi Young Scholar Project of the China University of Mining and Technology at Beijing.

CONFLICT OF INTEREST STATEMENT

None.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Addison-Wesley Longman.

Béjar, R., Lacasta, J., Lopez-Pellicer, F. J., & Nogueras-Iso, J. (2023). Discrete Global Grid Systems with quadrangular cells as reference frameworks for the current generation of earth observation data cubes. *Environmental Modelling & Software*, *162*, 105656. https://doi.org/10.1016/j.envsoft.2023.105656

Berndt, D. J., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, Seattle, WA (pp. 359–370). AAAI.

Besse, P., Guillouet, B., Loubes, J. M., & François, R. (2016). Review and perspective for distance based trajectory clustering. *IEEE Transactions on Intelligent Transportation Systems*, *17*(11), 306–3317. https://doi.org/10.1109/TITS.2016.2547641

Bowater, D., & Stefanakis, E. (2020). An open-source web service for creating quadrilateral grids based on the rHEALPix Discrete Global Grid System. *International Journal of Digital Earth*, *13*(9), 1055–1071. https://doi.org/10.1080/17538947.2019.1645893

Bowater, D., & Wachowicz, M. (2020). Modelling offset regions around static and mobile locations on a discrete global grid system: An IoT case study. *ISPRS International Journal of Geo-Information*, *9*(5), 335. https://doi.org/10.3390/ijgi9050335

Buchin, K., Buchin, M., Gudmundsson, J., Löffler, M., & Luo, J. (2011). Detecting commuting patterns by clustering subtrajectories. *International Journal of Computational Geometry & Applications*, *21*(3), 253–282. https://doi.org/10.1142/S0218195911003652

Chekol, A. G., & Fufa, M. S. (2022). A survey on next location prediction techniques, applications, and challenges. *EURASIP Journal on Wireless Communications and Networking*, *2022*(1), 29. https://doi.org/10.1186/s13638-022-02114-6

Chen, L., Özsu, M. T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. In *ACM SIGMOD International Conference on Management of Data*, Baltimore, MD (pp. 491–502). ACM.

Dodge, S., Gao, S., Tomko, M., & Weibel, R. (2020). Progress in computational movement analysis—Towards movement data science. *International Journal of Geographical Information Science*, *34*(12), 2395–2400. https://doi.org/10.1080/13658816.2020.1784425

Dodge, S., Laube, P., & Weibel, R. (2012). Movement similarity assessment using symbolic representation of trajectories. *International Journal of Geographical Information Science*, *26*(9), 1563–1588. https://doi.org/10.1080/13658816.2011.630003

Furtado, A. S., Alvares, L. O. C., Pelekis, N., Theodoridis, Y., & Bogorny, V. (2018). Unveiling movement uncertainty for robust trajectory similarity analysis. *International Journal of Geographical Information Science*, *32*(1), 140–168. https://doi.org/10.1080/13658816.2017.1372763

Gibb, R. G. (2016). The rHEALPix discrete global grid system. In *IOP Conference Series: Earth and Environmental Science* (Vol. *34*, 012012). IOP Publishing. https://doi.org/10.1088/1755-1315/34/1/012012

Gibb, R. G., Purss, M. B. J., Sabeur, Z., Strobl, P., & Tengteng, Q. (2022). Global reference grids for big earth data. *Big Earth Data*, *6*(3), 251–255. https://doi.org/10.1080/20964471.2022.2113037

Gong, X., Huang, Z., Wang, Y., Wu, L., & Liu, Y. (2020). High-performance spatiotemporal trajectory matching across heterogeneous data sources. *Future Generation Computer Systems*, *105*, 148–161. https://doi.org/10.1016/j.future.2019.11.027

Goodchild, M. F. (2018). Reimagining the history of GIS. *Annals of GIS*, *24*(1), 1–8. https://doi.org/10.1080/19475683.2018.1424737

Guo, Y., Li, B., Lu, Z., & Zhou, J. (2022). A novel method for road network mining from floating car data. *Geo-Spatial Information Science*, *25*(2), 197–211. https://doi.org/10.1080/10095020.2021.2003165

Hausdorff, F. (1914). Grundzüge der mengenlehre. *Monatshefte Für Mathematik Und Physik*, *26*(1), A34–A35. https://doi.org/10.1007/BF01999507

Hojati, M., Farmer, C., Feick, R., & Robertson, C. (2021). Decentralized geoprivacy: Leveraging social trust on the distributed web. *International Journal of Geographical Information Science*, *35*(12), 2540–2566. https://doi.org/10.1080/13658816.2021.1931236

Keler, A., Krisp, J. M., & Ding, L. (2017). Detecting vehicle traffic patterns in urban environments using taxi trajectory intersection points. *Geo-Spatial Information Science*, *20*(4), 333–344. https://doi.org/10.1080/10095020.2017.1399672

Kraemer, M. U., Yang, C. H., Gutierrez, B., Wu, C. H., Klein, B., Pigott, D. M., Du Plessis, L., Faria, N. R., Li, R., Hanage, W. P., & Brownstein, J. S. (2020). The effect of human mobility and control measures on the COVID-19 epidemic in China. *Science*, *368*(6490), 493–497. https://doi.org/10.1126/science.abb4218

Li, J., Li, Q., Zhu, Y., Ma, Y., Xu, Y., & Xie, C. (2019). An automatic extraction method of coach operation information from historical trajectory data. *Journal of Advanced Transportation*, *2019*(PT.1), 1–15. https://doi.org/10.1155/2019/3634942

Li, J., Liu, J., Qiao, L., Zhang, Y., Lu, W., Zhang, C., Huang, Q., & Wang, H. (2022). gsstSIM: A high-performance and synchronized similarity analysis method of spatiotemporal trajectory based on grid model representation. *Transactions in GIS*, *26*(5), 2206–2224. https://doi.org/10.1111/tgis.12944

Li, J., Wang, J., Zhang, J., Qin, Q., Jindal, T., & Han, J. (2016). A probabilistic approach to detect mixed periodic patterns from moving object data. *GeoInformatica*, *20*(4), 715–739. https://doi.org/10.1007/s10707-016-0261-2

Li, M., McGrath, H., & Stefanakis, E. (2022). Multi-resolution topographic analysis in hexagonal Discrete Global Grid Systems. *International Journal of Applied Earth Observation and Geoinformation*, *113*, 102985. https://doi.org/10.1016/j.jag.2022.102985

Li, M., & Stefanakis, E. (2020). Geospatial operations of discrete global grid systems—A comparison with traditional GIS. *Journal of Geovisualization and Spatial Analysis*, *4*(2), 1–21. https://doi.org/10.1007/s41651-020-00066-3

Liu, Y., Liu, X., Gao, S., Gong, L., Kang, C., Zhi, Y., Chi, G., & Shi, L. (2015). Social sensing: A new approach to understanding our socioeconomic environments. *Annals of the Association of American Geographers*, *105*(3), 512–530. https://doi.org/10.1080/00045608.2015.1018773

Liu, Y., & Seah, H. S. (2015). Points of interest recommendation from GPS trajectories. *International Journal of Geographical Information Science*, *29*(6), 953–979. https://doi.org/10.1080/13658816.2015.1005094

Magdy, N., Sakr, M. A., Mostafa, T., & El-Bahnasy, K. (2015). Review on trajectory similarity measures. In *IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, Cairo, Egypt (pp. 613–619). IEEE. https://doi.org/10.1109/IntelCIS.2015.7397286

Mahdavi-Amiri, A., Alderson, T., & Samavati, F. (2015). A survey of digital earth. *Computers & Graphics*, *53*, 95–117. https://doi.org/10.1016/j.cag.2015.08.005

Mariescu-Istodor, R., & Fränti, P. (2017). Grid-based method for GPS route analysis for retrieval. *ACM Transactions on Spatial Algorithms and Systems*, *3*(3), 1–28. https://doi.org/10.1145/3125634

May Petry, L., Leite Da Silva, C., Esuli, A., Renso, C., & Bogorny, V. (2020). MARC: A robust method for multiple-aspect trajectory classification via space, time, and semantic embeddings. *International Journal of Geographical Information Science*, *34*(7), 1428–1450. https://doi.org/10.1080/13658816.2019.1707835

OGC. (2017). *Topic 21: Discrete global grid systems abstract specification*. http://docs.opengeospatial.org/as/15-104r5/15-104r5.html

Oueslati, W., Tahri, S., Limam, H., & Akaichi, J. (2023). A systematic review on moving objects' trajectory data and trajectory data warehouse modeling. *Computer Science Review*, *47*, 100516. https://doi.org/10.1016/j.cosrev.2022.100516

Peterson, P. R. (2016). Discrete global grid systems. In D. Richardson (Ed.), *International Encyclopedia of Geography: People, the Earth, Environment and Technology* (pp. 1–10). John Wiley & Sons. https://doi.org/10.1002/9781118786352.wbieg1050

Petry, L. M., Ferrero, C. A., Alvares, L. O., Renso, C., & Bogorny, V. (2019). Towards semantic-aware multiple-aspect trajectory similarity measuring. *Transactions in GIS*, *23*(5), 960–975. https://doi.org/10.1111/tgis.12542

Qian, C., Yi, C., Cheng, C., Pu, G., Wei, X., & Zhang, H. (2019). Geosot-based spatiotemporal index of massive trajectory data. *ISPRS International Journal of Geo-Information*, *8*(6), 284. https://doi.org/10.3390/ijgi8060284

Robertson, C., Chaudhuri, C., Hojati, M., & Roberts, S. A. (2020). An integrated environmental analytics system (IDEAS) based on a DGGS. *ISPRS Journal of Photogrammetry and Remote Sensing*, *162*, 214–228. https://doi.org/10.1016/j.isprsjprs.2020.02.009

Salvador, S., & Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, *11*(5), 561–580. https://doi.org/10.3233/IDA-2007-11508

Sousa, R. S. D., Boukerche, A., & Loureiro, A. A. (2020). Vehicle trajectory similarity: Models, methods, and applications. *ACM Computing Surveys*, *53*(5), 1–32. https://doi.org/10.1145/3406096

Su, H., Liu, S., Zheng, B., Zhou, X., & Zheng, K. (2020). A survey of trajectory distance measures and performance evaluation. *The VLDB Journal*, *29*(1), 3–32. https://doi.org/10.1007/s00778-019-00574-9

Sun, W., Cui, M., Zhao, X., & Gao, Y. (2009). A global discrete grid modeling method based on the spherical degenerate quadtree. *2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing*, Boston, MA (pp. 308–311). IEEE.

Toohey, K., & Duckham, M. (2015). Trajectory similarity measures. *Sigspatial Special*, *7*(1), 43–50. https://doi.org/10.1145/2782759.2782767

Ulmer, B., Hall, J., & Samavati, F. (2020). General method for extending discrete global grid systems to three dimensions. *ISPRS International Journal of Geo-Information*, *9*(4), 233. https://doi.org/10.3390/ijgi9040233

Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. *18th International Conference on Data Engineering*, San Jose, CA (pp. 673–684). IEEE.

Wang, C., Zourlidou, S., Golze, J., & Sester, M. (2021). Trajectory analysis at intersections for traffic rule identification. *Geo-Spatial Information Science*, *24*(1), 75–84. https://doi.org/10.1080/10095020.2020.1843374

Wang, D., Miwa, T., & Morikawa, T. (2020). Big trajectory data mining: A survey of methods, applications, and services. *Sensors*, *20*(16), 4571. https://doi.org/10.3390/s20164571

Wang, Y., Qin, K., Chen, Y., & Zhao, P. (2018). Detecting anomalous trajectories and behavior patterns using hierarchical clustering from taxi GPS data. *ISPRS International Journal of Geo-Information*, *7*(1), 25. https://doi.org/10.3390/ijgi7010025

Yao, X., Li, G., Xia, J., Ben, J., Cao, Q., Zhao, L., Ma, Y., Zhang, L., & Zhu, D. (2020). Enabling the big earth observation data via cloud computing and DGGS: Opportunities and challenges. *Remote Sensing*, *12*(1), 62. https://doi.org/10.3390/rs12010062

Zheng, Y. (2015). Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, *6*(3), 1–41. https://doi.org/10.1145/2743025

Zheng, Y., Liu, J., Li, J., Xu, Y., Huang, Q., Zhu, Y., & Pei, Y. (2019). Design of fine management system for civil aviation airspace resources based on spatiotemporal grid model. *1st International IEEE Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, Kunming, China (pp. 547–551). IEEE.

Zhou, C., Ou, Y., & Ma, T. (2010). Progresses of geographical grid systems researches. *Progress in Geography*, *28*(5), 657–662. https://doi.org/10.11820/dlkxjz.2009.05.002

Zhou, J., Ben, J., Huang, X., Wang, R., Liang, X., Ding, J., & Liang, Q. (2023). Efficient cell navigation methods and applications of an aperture 4 hexagonal discrete global grid system. *International Journal of Geographical Information Science*, *37*(3), 529–549. https://doi.org/10.1080/13658816.2022.2125972

How to cite this article: Liu, J., Li, J., Qiao, L., Li, M., Stefanakis, E., Zhao, X., … Zhang, C. (2024). QuadGridSIM: A quadrilateral grid-based method for high-performance and robust trajectory similarity analysis. *Transactions in GIS*, *00*, 1–25. https://doi.org/10.1111/tgis.13126