ASCE

Fast and Reliable Map Matching from Large-Scale Noisy Positioning Records

Yanyu Wang, S.M.ASCE[1]; Ruoxin Xiong, S.M.ASCE[2]; Pingbo Tang, P.E., M.ASCE[3]; and Yongming Liu, A.M.ASCE[4]

Abstract: Optimizing airport operational performance requires analyzing large-scale and noisy taxiing aircraft trajectory data on the ground, such as the airport surface detection equipment, model X (ASDE-X) data. Map matching techniques can interpret sensor-based noisy aircraft trajectories to traffic events occurring on specific airport roads (i.e., runways and taxiways). Such interpretation served as the foundation of the following traffic analysis. However, inevitable measurement noise and errors originating from multisensor systems pose substantial challenges in achieving accurate map matching. In addition, existing map matching methods are typically inefficient in processing tens of millions of noisy aircraft positioning records generated from large metropolitan airports. In this paper, the authors propose a new map matching algorithm that can achieve computational efficiency and high accuracy in interpreting large-scale and noisy aircraft trajectories on the ground into coherent road representations. The new algorithm consists of three main components: (1) dense trajectory compression, (2) complex road network segmentation, and (3) map matching based on multiple candidate nodes. These three components collectively speed up the matching process without losing accuracy. The authors evaluated and compared the proposed algorithm with state-of-the-art map matching algorithms on an established airport data set consisting of over 100 real-world trajectories with a total length of 581.8 km. The proposed algorithm achieved nearly linear time complexity for matching aircraft positioning records with ground transportation networks, while other methods with similar accuracy need exponential time complexity. Also, the new algorithm outperformed a state-of-the-art fast map matching method, spatial temporal (ST)-mapping, by 79.5% and 78.6% in segment and length accuracy, respectively. **DOI: 10.1061/(ASCE)CP.1943-5487.0001054.** *© 2022 American Society of Civil Engineers.*

Practical Applications: Map matching is vital for many transportation applications to be able to identify and analyze bottlenecks in current traffic management based on historical vehicle spatial records. Large-scale historical spatial records with noise require rapid and accurate map matching methods. In this research, the authors propose a fast and reliable automatic map matching method that promises to process massive historical spatial records with noise for offline usage. The proposed map matching method has three advantages: (1) high speed—the proposed map matching method works on linear time and therefore can be scaled very efficiently, (2) high accuracy—the authors used the proposed map matching method to process 100 aircraft trajectories on a complex airport map, and the results showed that with proper setting of the parameters, the proposed map matching method consistently achieved above 95% matching accuracy, and (3) only needs raw vehicle positioning records capturing sequences of vehicle positions without requiring any other information or metadata.

Author keywords: Air traffic management; Map matching; Airport ground operation.

Introduction

The increasing growth of air traffic poses a significant challenge to the safety and efficiency of airport ground operations (Dixit and Jakhar 2021; Liu et al. 2016; Song et al. 2018). From 1995 to 2008, approximately 429 commercial aircraft were involved in ground collisions (i.e., in push-back, taxiing, and so on), resulting in 973 fatalities (Wilke et al. 2014). In addition, air traffic accounts for approximately 2% of total carbon emissions produced by human activities, and 36% of fuel is consumed during the taxiing phase on the ground (Stettler et al. 2011, 2018; Zhang et al. 2019). Deficient airport operational performance requires necessary and urgent improvements through the analysis and identification of bottlenecks in complex air traffic trajectories and events.

The widely installed airport surface detection equipment, model X (ASDE-X), which uses radar, multilateration, and satellite technologies, can monitor and track aircraft surface movements. ASDE-X data contain flight IDs and corresponding sequential location information for flights operating in an airport. Large-scale ASDE-X data allows for postanalysis of deficient and unsafe aircraft surface trajectories based on coherent map matching results. Map matching can interpret aircraft positioning data into meaningful traffic events and routing strategies by matching recorded aircraft locations with elements of ground road networks. Specifically, map matching combines a time series of coordinate tuples with a road network consisting of nodes and directed links to describe

[1]Ph.D. Student, Dept. of Civil and Environmental Engineering, Carnegie Mellon Univ., P.O. Box 15213, Pittsburgh, PA 15213. Email: yanyuwan@andrew.cmu.edu

[2]Ph.D. Student, Dept. of Civil and Environmental Engineering, Carnegie Mellon Univ., P.O. Box 15213, Pittsburgh, PA 15213. Email: ruoxinx@andrew.cmu.edu

[3]Associate Professor, Dept. of Civil and Environmental Engineering, Carnegie Mellon Univ., P.O. Box 15213, Pittsburgh, PA 15213 (corresponding author). ORCID: https://orcid.org/0000-0002-4910-1326. Email: ptang@andrew.cmu.edu

[4]Professor, School for Engineering of Matter, Transport, and Energy, Arizona State Univ., P.O. Box 85287, Tempe, AZ 85287. ORCID: https://orcid.org/0000-0001-7369-2974. Email: Yongming.Liu@asu.edu

Note. This manuscript was submitted on February 8, 2022; approved on July 5, 2022; published online on September 23, 2022. Discussion period open until February 23, 2023; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801.

a traveler's spatial movement (Knapen et al. 2018). The outcome of map matching is a sequence of links in road networks crossed by the travelers in chronological order. Map matching is a critical preprocessing step for many intelligent transportation applications and services, such as navigation aids, route choice behavior analysis, road pricing management, and road design (Giannotti et al. 2007; Knapen et al. 2018; Tanaka et al. 2021; Toledo-Moreo et al. 2009; Velaga and Pangbourne 2014). The accuracy of map matching results directly influences the results of the aforementioned applications and services (Kong et al. 2009; Pereira et al. 2009; Velaga and Pangbourne 2014; Velaga et al. 2010; Wu et al. 2020). Inaccurate map matching results may result in inappropriate updates of applications and may then result in larger errors in practice (Wang et al. 2019).

Accurate map matching results from ASDE-X data can support the development of intelligent transportation applications in airports, improving the safety and efficiency of airport ground operations (Barnett et al. 2015). However, three challenges have impeded airport operators from mapping aircraft moving events with ASDE-X data containing millions of points with noise. The first is inevitable noise in aircraft positioning data, including errors caused by fluctuations in sensor signals and approximations required for merging multisensor signals in the ASDE-X system (FAA 2020). Deviations between aircraft positioning data and the centerline of an airport road caused by inevitable noise can reach up to 67.79 m (Ku et al. 2018). Second, a massive amount of data requires numerous memory and computational resources for existing map matching techniques. For a metropolitan airport, the daily number of ASDE-X data points may be in the millions (FAA 2021), as over 1,000 aircraft land and take off in a single day, and each of the aircraft may produce over 2,000 positioning data points. The developed map matching method should process ASDE-X data with millions of points in a fast and scalable way. Third, the geometry of an airport's road network is complex. Airport road networks contain multiple roads crossing at one intersection and involve numerous intersections with angles significantly different from regular rectangles. For example, there are over 150 intersections at Los Angeles International airport (OpenStreetMap 2017). Existing map matching techniques based on vehicle road networks may produce mismatching errors around these complex intersections (Hashemi and Karimi 2014).

This paper presents a computationally efficient (linear time) and accurate map matching method that uses historical aircraft spatial data to reconstruct meaningful airport ground operations. Specifically, this new method integrates three techniques to speed up the matching process without decreasing accuracy: (1) dense trajectory compression, (2) complex road network segmentation, and (3) map matching based on multiple candidate nodes. The authors validated that the proposed map matching method is robust for a wide range of parameter settings. In addition, because there is a lack of real-world public aircraft trajectory data sets with ground truths (Knapen et al. 2018), this study established an ASDE-X trajectory data set with 100 aircraft trajectories by manual annotation. Last, the authors validated the proposed map matching method on the established data set using a visualization module that helps increase manual checking efficiency and the reliability of map matching results.

The organization of the remaining sections of this paper is as follows. Section two reviews previous studies related to map matching and demonstrates the advantages of the proposed map matching method. Section three illustrates a map matching problem in airport ground operations and the data set used in the research. Section four describes technical details of the proposed map matching method with workflow figures and pseudocodes. Section five presents the proposed map matching method's performance and evaluates its robustness. Section six discusses research limitations and future research. Section seven presents a summary of the research findings.

Background Research

This section synthesizes studies relevant to various map matching techniques. Specifically, the first part compares two map matching techniques: online map matching and offline map matching. The second part provides a detailed review of offline map matching methods and explains why offline methods were adopted in this study.

Map Matching Methods

Map matching algorithms consist of two categories: online and offline map matching. Online map matching locates the road segment that a user is traveling in real-time (Hashemi and Karimi 2014; Taguchi et al. 2018). Online map matching is mainly used for navigation aids, using spatial records up to the current time to implement map matching (Knapen et al. 2018; Tanaka et al. 2021; Toledo-Moreo et al. 2009). The process of online map matching is fast but has lower accuracy compared to offline map matching (Hashemi and Karimi 2014).

Offline map matching methods use all spatial records for each trip to reconstruct sequences of traffic events of an entire travel session, providing the entire sequence of road segments on which a user traveled (Hashemi and Karimi 2014). Such entire trip map matching usually involves large-scale data processing and intensive computation. The offline matching method mainly works for applications in which capturing the actual path for an entire trip is more important than knowing the instantaneous position of the user. Offline map matching is typically used for long-term capacity management and short-term traffic flow analysis, such as route choice behavior and road pricing management (Giannotti et al. 2007; Tanaka et al. 2021; Velaga and Pangbourne 2014). Accurate map matching results are also essential for reconstructing daily traffic flow, locating inefficient and unsafe operations in current practice, and revising such operations in the planning phase. This research focused on offline map matching because it offers more accurate matching results to guide the systematic diagnosis of airport ground operation strategies.

Offline Map Matching Methods

There are three existing types of offline map matching techniques: (1) pure geometric methods, (2) topological and probabilistic methods, and (3) machine learning–based methods. Pure geometric methods have three subcategories: (1) point-point matching (finding the nearest node), (2) point-curve matching (finding the polyline to which the distance is minimal), and (3) curve-curve matching (matching vehicle trajectory against known roads) (Knapen et al. 2018; Quddus et al. 2007). These methods have lower computational complexity, but the results have relatively low accuracy, especially at road intersections (Quddus et al. 2007).

Topological and probabilistic map matching typically use hidden Markov models (HMM). These methods use the topological relationships between spatial records and road networks as the probability measurements in HMM (Lou et al. 2009; Tran et al. 2020; Wei et al. 2012). Such HMM-based methods are generally more accurate around road intersections than pure geometric map matching (Quddus et al. 2007). However, HMM-based map matching methods consume much memory and computational time to compare and process topological relationships.

Table 1. Summary of different offline map matching methods

Map matching method	Advantages	Disadvantages
Pure geometric	Fast map matching speed with low computational complexity for small data sets	Low accuracy for map matching results, especially for road intersections
Topological and probabilistic	High accuracy on the map matching result	Slow map matching speed with high computational complexity
Machine learning–based	Fast map matching speed and high accuracy for map matching results	Requires large labeled training data sets for map matching

Machine learning–based map matching methods typically use neural networks to train a model to identify mapped road segments (Shen et al. 2020). Machine learning–based methods perform well with large amounts of labeled training data sets (Shen et al. 2020). However, most real-world map matching applications usually have few labeled data sets. Some researchers consider that "collecting a large set of trajectories with the true mapped road segment is nearly impossible" (Knapen et al. 2018).

Table 1 provides a summary of the aforementioned three offline map matching methods. Offline map matching aims to obtain accurate map matching results for future planning work. The aforementioned studies have contributed to developing an offline map matching method, but none of them were able to simultaneously achieve fast and reliable map matching practically. Meanwhile, the aforementioned studies need to identify possible nodes or edges in the road network for every spatial record in chronological order to achieve high accuracy. As a result, for large-scale spatial records, the computation time of these methods grows substantially and exponentially. In this study, the authors investigated a new offline map matching method to overcome these limitations.

Map Matching Problem Statement

The authors used aircraft taxiing trajectories from Los Angeles International airport (LAX) in this research to develop and test a new map matching approach that overcomes the aforementioned challenges. LAX is consistently among the top five busiest airports in the United States (US Bureau of Transportation Statistics 2020). The demand for aircraft throughput at LAX is heavy. The number of daily throughputs is greater than 1,000 (FAA 2021). In this research, the authors used a digital map of the LAX network from public OpenStreet map data (OpenStreetMap 2017) and 100 aircraft taxiing records from ASDE-X data collected on September 5, 2017.

The proposed map matching method can be applied to different airports with different digital maps. This research used LAX to illustrate the methodology. The LAX airport has four runways, dozens of ramp areas, and various intersections, making the airport network's geometry complex. A map with high geometric complexity brought additional challenges for the fast and accurate map matching method developed. The black lines in Fig. 1 represent the airport road network; the black circles represent intersections. The LAX digital map consists of 693 road segments. The mean length of the road segments is 101.65 m. The minimum length of all LAX road segments is 0.004 m, and the maximum length of all LAX road segments is 808.22 m.

ASDE-X records the positioning of moving aircraft around an airport, reflecting airport ground operation scenarios. Each arrival or departure aircraft has thousands of ASDE-X records, which contain an aircraft's latitude and longitude information for nearly every second. Because the ASDE-X system uses multiple sensors to provide location information and has no annotation of sensor resources for each location data point, location information usually contains much inherent noise and uncertainty with regard to location information deriving from various sensors.

As shown in Fig. 2, the black lines and nodes represent the airport network. The dots represent ASDE-X data for one aircraft trajectory in the airport. The authors classified the noise in the spatial data into two types: (1) noisy spatial data around intersections; and (2) noisy spatial data along parallel roads. As shown in Fig. 2 in the box for noise type 1, noisy spatial data around intersections can significantly deviate from an actual road segment. The selected spatial record is closer to road segment 1 than it is to road segment 2. A map matching algorithm would have low accuracy if it used the nearest road segment method to map each spatial record. In this case, a map matching algorithm based on the nearest road segment would inaccurately map the selected spatial record to road segment 1 instead of to road segment 2. If we want to use the nearest node instead of the nearest edge to fix the problem raised by noise type 1, we will have inaccurate map matching results in spatial records with noise type 2.

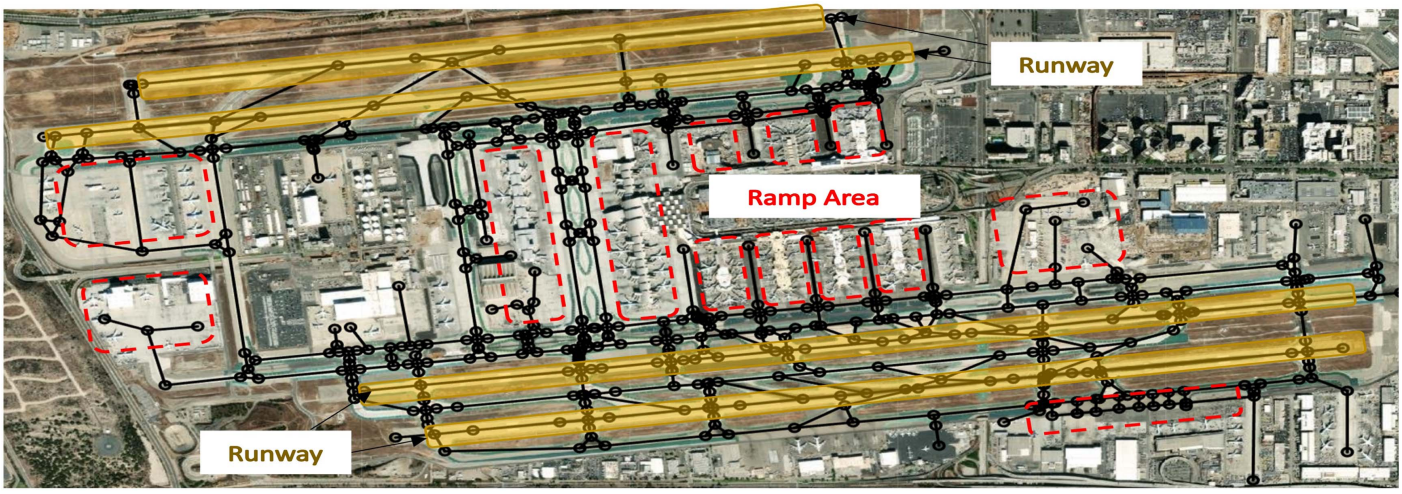

Fig. 1. Digital map of LAX. (Base map © OpenStreetMap contributors.)

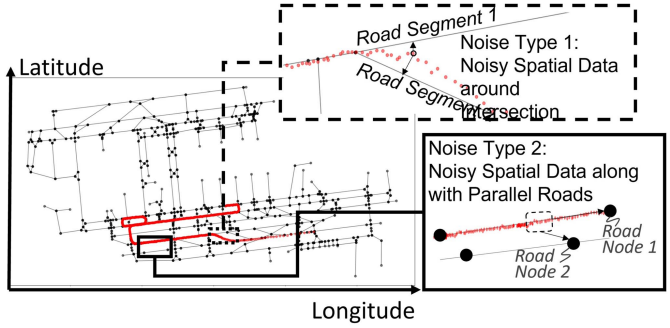

Fig. 2. Noise spatial data (ASDE-X) for one aircraft trajectory at LAX.

The box labeled noise type 2 in Fig. 2 shows circled noisy spatial data along parallel roads. In some cases, some records between two nodes on the map are closer to a node on a line parallel to the current road segment. For example, inaccurate map matching will label the circled records as belonging to the road node 2 instead of road node 1, because they are closer to road node 2 than they are to road node 1. Computer algorithms need a consistent map matching method for all spatial records, but humans can intuitively correct noise and select the correct road segment for mapping. A human can extract critical spatial records from many noisy records and use these critical records to guide the connections of all records into the most likely road segments followed by an aircraft. Inspired by the cognitive process of map matching by humans extracting and using critical point records among many noisy records along moving paths, the authors propose the following map matching process, which is fast and accurate.

Methodology

The proposed offline map matching framework consists of three steps: (1) compress a dense trajectory with noise to reduce computational complexity; (2) segment airport road networks to further decrease computational complexity and enhance accuracy using functional area annotations; and (3) identify the most accurate mapped paths from candidate nodes queried from the segmented airport road network.

Fig. 3 shows the proposed map matching method implemented as three components. Different components in each algorithm have been annotated using different shapes; the legend is shown in the bottom right-hand corner of Fig. 3. The first step compresses the input ASDE-X spatial records and identifies critical records among all spatial records (Algorithm 1). This step reduces the number of spatial records for map matching and maintains the geometric shape of spatial records. The second step segments an airport road network to a K-dimensional (K-D) tree representing the airport network's intersections and roads (Algorithm 2). The K-D tree method segments the airport road network and enables fast queries for the distance between airport network nodes and ASDE-X spatial records. Annotations of functional areas (runway and ramp) in the K-D tree enhance matching accuracy. The third step generates map matching results based on multiple candidate nodes (Algorithm 3). The following three subsections illustrate the three steps in detail with workflow charts and pseudocodes.

Compress Trajectory Using Improved Douglas–Peucker Algorithm

Traditional map matching searches a mapped road edge or node for each spatial record. In a dense and noisy spatial record data set, such as ASDE-X, finding the corresponding road edge or nodes for each spatial record is tedious and error-prone. Therefore, we needed to compress spatial records before comparing records and map elements extensively. Uniform sample (Tobler 1966), Bellman algorithm (Bellman 1961), and Douglas–Peucker algorithm (Douglas and Peucker 1973; Ramer 1972) are three compression methods for spatial positioning data (Liu et al. 2019; Zhao and Shi 2018). The uniform sampling algorithm takes each ith point in trajectory

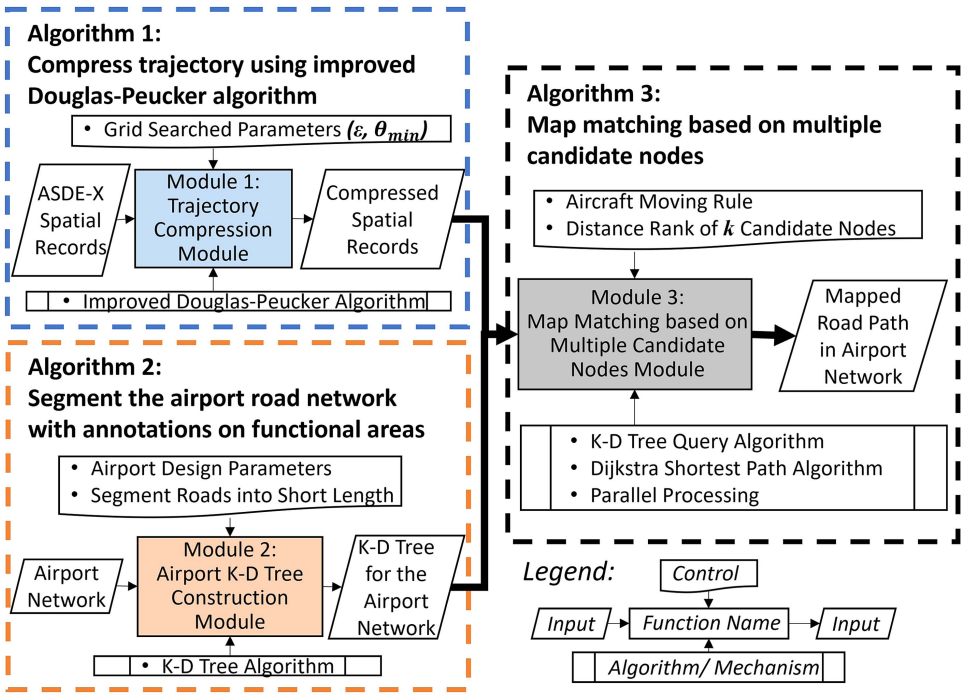

Fig. 3. Overall framework of the proposed map matching method following integrated definition for function modeling (IDEF0). (Data from NIST 1993.)

coordinates. The arbitrary setting of i in uniform sampling leads the compressed trajectory to be either oversimplified (lose geometric features such as turning points) with large values of i, or remain too complex with small values of i. Bellman algorithm can keep the geometry of a trajectory well, but it requires computation time up to $O(n^3)$. The authors adopted the Douglas–Peucker (DP) algorithm with a constraint on an aircraft's directional change to compress noisy dense spatial trip data. The average computational complexity of the DP algorithm is $O(n \cdot \log(n))$. In the worst case, if the link between spatial data is a zigzag line with consistent increasing amplitude, the computational complexity of the DP algorithm is $O(n^2)$. Because such zigzag lines rarely happen in transportation trajectories, the improved DP algorithm is computation-friendly and can retain the geometry of a trajectory. There are two predefined parameters in the proposed Algorithm 1, that is, distance threshold ε and minimal change of the heading angle $\theta_{\min}$. In this research, the numerical values of these two parameters were learned through a grid search (see the section on parameter initialization). Grid search is a comprehensive search of parameters in the algorithm. The search space is bound by a predefined upper and lower bound. Users can set several interpolating points inside the bounds and choose points with high performance of the algorithm. The following illustrates the workflow of the proposed algorithm.

Steps 1 to 6 in Fig. 4 show a schematic of the DP algorithm. The original trajectory records a turning behavior from the right direction to the down direction. The original trajectory consists of a series of dense and noisy spatial records denoted as $[a_1, a_2, \ldots, a_T]$. The DP algorithm needs distance threshold ε to be set before calculation. This threshold can help the algorithm retain critical geometry information points, such as turning points. As shown in Fig. 4, the DP algorithm first draws a line between the last spatial point a_T and the initial spatial point a_1. Then, the algorithm computes the distance between the drawn line and the remaining spatial points following the time series sequence. When the algorithm finds the first spatial point that has a larger distance than the threshold—which is a_3 in Fig. 4—the algorithm discards the spatial point between a_1 and a_3. Then, the algorithm marks a_1 and a_3 as the first two compressed spatial points, d_1 and d_2. At this moment, d_2 equals the last compressed spatial point, $d_{\max}$. Next, the algorithm runs iteratively between a_T and the last compressed spatial point,$d_{\max}$, until there is no spatial data between a_T and $d_{\max}$.

A heading direction constraint can process the compressed spatial data from the DP algorithm to simplify the trajectory further and retain turning point information. As shown in steps 7 and 8 in Fig. 4, the trajectory compression module iteratively calculates the heading angles between the compressed spatial data. The trajectory compression module removes spatial data with a heading angle smaller than the predefined minimal change of the heading angle. The compressed trajectory maintains the geometric shape of the turning behavior and contains fewer spatial records than the original spatial records. Algorithm 1 illustrates the overall calculation process.

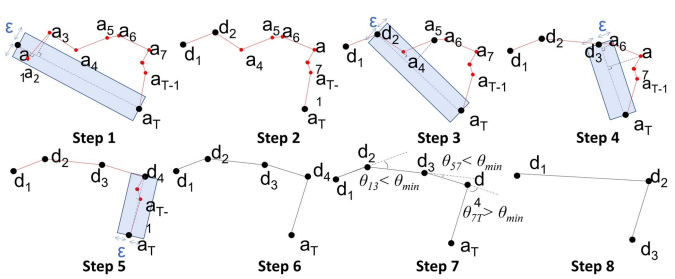

Fig. 4. Schematic of the improved Douglas–Peucker algorithm.

Algorithm 1. Trajectory Compression Module

Module Name: Trajectory Compression Module
Input: ASDE-X location record $< a_1, a_2, \ldots a_T >$ for trajectory Z
Output: compressed location record $< d_1, d_2, \ldots d_j > (j \ll T)$ for the input trajectory Z

set the minimal distance ε, and minimum angle $\theta_{\min}$ ($\varepsilon = 10$ *m*, $\theta_{\min} = 10$)
set $d_1 = a_1$, *lst_critical_node* = $[d_1]$
for i in range $(1, T)$:
  create *line* $i{-}T$ between a_i and a_T
  for j in range $(i + 1, T)$:
    calculate distance between a_j and *line* $i{-}T$
    if *distance* $> \varepsilon$:
      add a_j to *lst_critical_node as* d_i
      $i = j$
      break
add a_T to *lst_critical_node*
for k in range(length(*lst_critical_node*)-2):
  calculate the angle θ between the two lines of line (d_k, d_{k+1}), and line (d_{k+1}, d_{k+2})
  if $\theta < \theta_{min}$:
    delete d_{k+1} from *lst_critical_node*
return *lst_critical_node* $<d_1, d_2, \ldots d_j>$

Segment Airport Road Network with Annotations on Functional Areas

The next step is to segment the airport road network with annotations on functional areas (runway and ramp areas) for fast query candidate nodes for ASDE-X records in the compressed trajectory. For large metropolitan airports with highly complex spatial structures with thousands of nodes in the road network, such segmentation can significantly improve computation efficiency. In addition, annotations of functional areas (i.e., runway and ramp area) enhance the accuracy of map matching results. Annotations help the proposed map matching find correct start nodes and end nodes in the airport road network for each trajectory, as illustrated in Algorithm 3.

The objective of segmenting the airport road network is to enable fast query of the nearest node around the spatial records. Table 2 shows three methods for constructing and querying the nearest node in a map for spatial data. For one airport, the proposed method only needs to construct a spatial tree for the airport road network once, as shown in Algorithm 2 (Fig. 3). For each aircraft trajectory, the proposed method needs to query the k-nearest candidate nodes in the spatial tree, as shown in Algorithm 3 (Fig. 3). Because the number of operations in constructing a K-D tree and querying in the K-D tree are different in the proposed approach, Table 2 lists the computational complexity for these two operations separately. In the map matching complexity computation, N represents the number of nodes in a map. Construction complexity

Table 2. Computational complexity comparison

Algorithm	K-D Tree	Ball tree	Brute force
Construction complexity	$O[N\log(N)]$	$O[N(\log(N))^2]$	$O[N^2]$
Query complexity	$O[\log(N)]$	$O[\log(N)]$	$O[N]$

Source: Data from Munaga and Jarugumalli (2011).

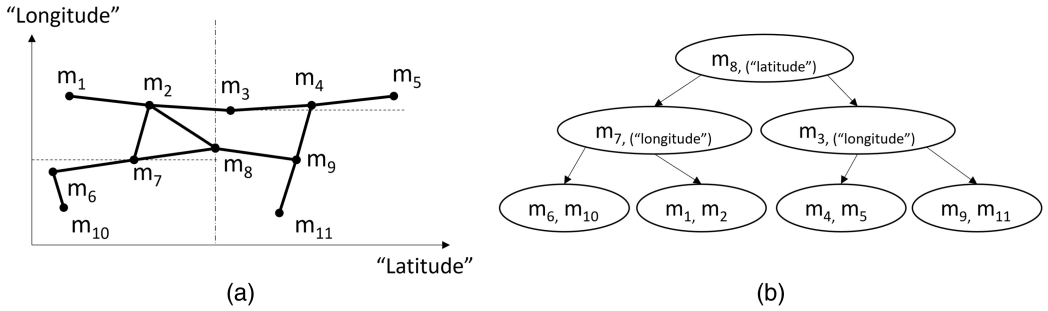

Fig. 5. Example of constructing a K-D tree for an airport map: (a) partition of the airport map; and (b) K-D tree structure for the partition.

measures computation times for segmenting a map node into a specific data structure. The following paragraph gives a detailed illustration of the construction of a K-D tree. Query complexity is a concept for quantifying needed computation times for querying the nearest node in the specific data structure for each spatial record. Both K-D and ball trees have lower spatial query complexity than brute force. In the construction phase, a K-D tree requires lower construction time; therefore, we adopted K-D trees to generate candidate nodes in the proposed map matching method.

K-D tree is a static data structure that enables effective tracing of the nearest data with a space-division tree structure. A typical binary K-D tree has two child nodes for each node in the tree. Each node corresponds to an outcome of the comparison of two records based on a particular key chosen as a discriminator (Lee and Wong 1977). To trace the nearest map node for each spatial record, we constructed a K-D tree for the nodes in the airport map. Fig. 5 gives an example of the construction of a K-D tree for an airport map. The discriminator adopted was the transformed longitude and latitude values. A projection of the original longitude and latitude coordinates to a plane generates the transformed longitude and latitude values. Such transformation can minimize geometry errors raised by the inconsistency of using Euclidean distance in tree construction based on latitude and longitude values. The algorithm repeats two queries that use the median of the transformed latitude value and the transformed longitude value for building the K-D tree. In the end, the K-D tree contains all map nodes.

The distributions of data points are different due to the variety of aircraft speeds. For example, aircraft speed on a runway may be high, so that spatial data on runways are sparse. Fig. 6 shows an example of the description and proposed solution for map matching with such sparse data on some road segments. The dots represent the spatial data in the trajectory, and the circles represent the nearest map nodes queried for each spatial data point. The arrows represent the heading direction implied by the spatial records. The shortest path mechanism was adopted to link the corresponding nodes in the digital map. The arrows represent map matching results for the spatial data with the shortest path mechanism. Scenario (a) [Fig. 6(a)] illustrates map matching errors resulting from sparse spatial data. Because of the inevitable noise in spatial data, the queried nearest node can be a mismatched node on a parallel runway, such as the two middle spatial data points in Scenario (a). Based on the mismatched nodes, the matched results in Scenario (a) imply that the aircraft changed runways. However, when landing or taking off at an airport, aircraft use one runway at one time. Scenario (b) provides the correct map matching results with the dense airport map. Specifically, in this research, we densified the airport map to half of the distance between the parallel roads in LAX airport, which is 50 m. Algorithm 2 illustrates the pseudocode for constructing the K-D tree for an airport map.

Algorithm 2. Airport K-D Tree Construction Module

Module Name: Airport K-D Tree Construction Module
Input: airport map G(N, E)
Output: K-D tree for airport map with shorter length of edges $G'(N', E')$

```
for edge e in G(N, E):
    if length(e)>50 m:
        divide edge e into multiple edges e' with length less than 50 m
    if e is in the runway area:
        annotate e as a runway element
    if e is in the ramp area:
        annotate e as a ramp element
construct K-D tree KD^tree (N') for G'(N', E')
```

Map Matching Based on Multiple Candidate Nodes

The final step is to match a compressed trajectory with corresponding map nodes or edges in the airport network. The matching process based on candidate nodes speeds up the map matching process by limiting the number of possible corresponding nodes in the digital map to the ASDE-X point records. Candidate nodes are the top k nearest nodes queried for each spatial record in the compressed trajectory. Using the top k candidate nodes for each

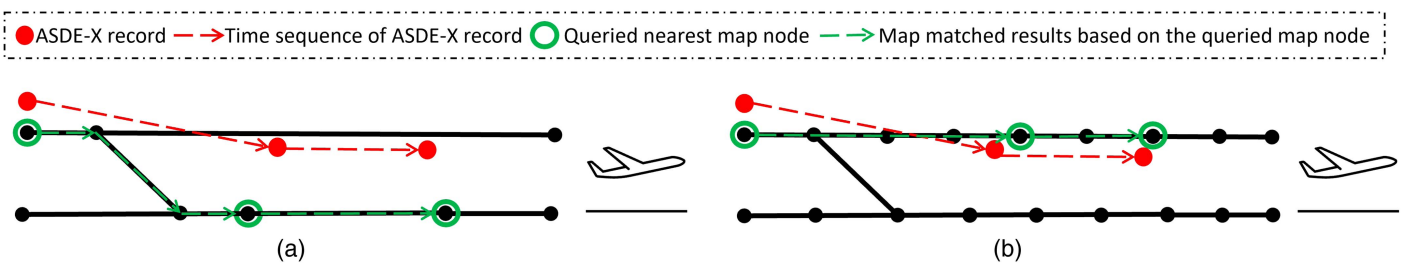

Fig. 6. Illustrations of sparse spatial data in map matching: (a) sparse spatial data with unorganized airport map leads to map matching errors; and (b) sparse spatial data with organized dense airport map leads to correct map matching results.

J. Comput. Civ. Eng., 2023, 37(1): 04022040

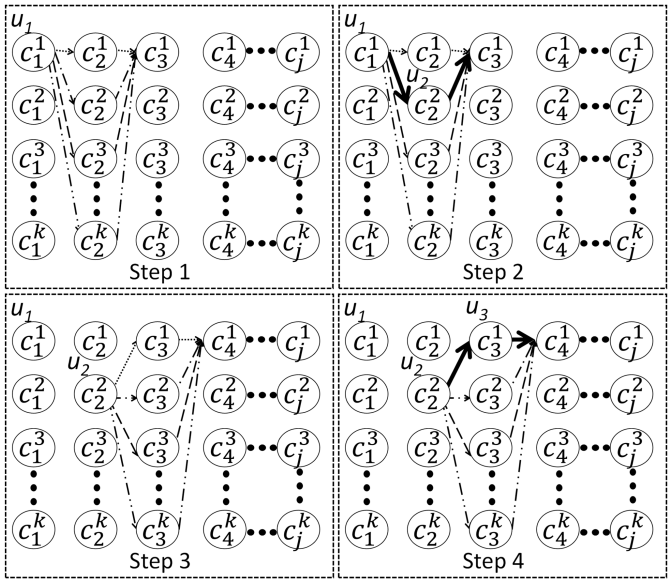

Fig. 7. Visualization of map matching based on the top k candidate nodes.

spatial data point instead of the nearest node can improve the accuracy level of map matching results. The aircraft moving rule adopted in the map matching module means that aircraft usually do not make U-turns in taxiing, because aircraft engines do not reverse. This rule suggests no reverse movements along one road segment in consecutive steps. The mapped path typically forms the shortest path between the sequential spatial records without back-and-forth motions. Therefore, the authors constructed the map matching module based on the Dijkstra shortest path algorithm (Dijkstra 1959) to link all possible nodes to find the most likely path on the digital map.

The proposed map matching first queried the top k nearest node in the constructed K-D tree of the airport for each spatial record in the compressed trajectory. Particularly, for the initial and last spatial records in each trajectory, the proposed map matching queried candidate nodes in corresponding functional areas, as shown in lines 2 through 7 in Algorithm 3. Because aircraft taxi between runway and ramp, the start and end nodes in the matched results must belong to one of the functional areas. Map matching using annotations prevents incorrect matching results for start and end nodes, because the distances between each runway and ramp area are significant. Then, Algorithm 3 identifies map matching results based on the generated candidate nodes. Algorithm 3 uses the nearest node as the mapped node for the first and last spatial records. For the rest of the spatial data, it takes the latest identified mapped node (u_i), all candidate nodes identified for the next spatial record ($c_{i+1}^1, \ldots, c_{i+1}^k$), and the nearest candidate node for the spatial record after the next spatial record (c_{i+2}^1) to find the mapped node for the next spatial record (u_{i+1}).

Fig. 7 provides an intuitive illustration. In the first step, c_1^1 equals the first identified mapped node u_1. The algorithm calculates the distances for all possible paths from u_1 to c_3^1. If the path passing through c_2^2 has the shortest distance, we select c_2^2 as updated candidate node u_2 in step 2. The bold lines represent the path with the shortest distance. Then, we repeat the process to find u_3 in steps 3 and 4. Algorithm 3 provides the pseudocode for the map matching module based on the top k candidate nodes.

Thus far, the overall framework has been presented with detailed illustrations. To summarize, Table 3 compares the characteristics and computational complexity for each algorithm in the proposed map matching method. In Table 3, n represents the number of spatial records in the input of an algorithm. Algorithm 1 helps to compress trajectories, delivering less spatial data, using the improved Douglas–Peucker algorithm. Compared to a uniform sampling strategy, Algorithm 1 can efficiently keep the geometric shape of a trajectory, that is, turning points in a trajectory. Compared to the Bellman algorithm, Algorithm 1 is more efficient for compressing two-dimensional (2D) ground trajectories. Algorithm 2 constructs the K-D tree of a dense airport map. Compared to brute force and ball trees, a K-D tree provides a more efficient data structure for querying the nearest node in an airport map. Algorithm 3 uses a successive updates strategy in finding correct map matching results based on k-nearest map nodes. Compared to the enumeration algorithm, Algorithm 3 reduces computational complexity from an exponential level to a linear level.

Table 3. Comparison of different algorithms in the map matching process

Function	Algorithm	Time complexity	Characteristics
Compress trajectory	Improved DP algorithm **(Algorithm 1)**	$(n \cdot \log(n))$	Low computational complexity; keep geometric shapes of trajectories well
	Bellman algorithm	$O(n^3)$	High computational complexity; keep geometric shapes of trajectories well
	Uniform sampling	$O(1)$	Low computational complexity; not good at keeping the geometric shapes of trajectories
Query Map Node	K-D tree **(Algorithm 2)**	$O(n \cdot \log(n)) + O(\log(n))$	Low computational complexity in low dimensional space ($d \leq 3$)
	Ball tree	$O(n \cdot (\log(n))^2) + O(\log(n))$	Low computational complexity in high dimensional space ($d > 3$)
	Brute force	$O(n^2) + O(n)$	High computational complexity
Identify map matched results	Successive updates based on k-nearest candidate nodes **(Algorithm 3)**	$O(k \cdot n)$ [k: number of candidate nodes]	Low computational complexity
	Enumeration algorithm based on k-nearest candidate nodes	$O(n^k)$ [k = number of candidate nodes]	High computational complexity

References for time complexity: (Bellman 1961; Munaga and Jarugumalli 2011; Tran et al. 2020).

Algorithm 3. Map Matching based on Multiple Candidate Nodes Module

Module Name: Map Matching based on Multiple Candidate Nodes Module
Input: airport's K-D tree KD^{tree} *(N')* and compressed location record $< d_1, d_2, \ldots d_j >$ for trip Z
Output: the matched path $P\,[p_1, p_2, \ldots, p_{j-1}]$ for the input trip Z
 # Generate candidate nodes for map matching
for compressed location record d_i in $< d_1, d_2, \ldots d_j >$:
  if $i = 1$ or $i = j$:
    identify the edge e', which has the shortest distance between d_i and functional areas
    if $e' \in$ runway area:
      search the k nearest node c_i^1 for d_i in KD^{tree} *(N')* with runway area annotations
    if $e' \in$ ramp area:
      search the k nearest node c_i^1 for d_i in KD^{tree} *(N')* with runway area annotations
  else:
    search the nearest k nodes for d_i in KD^{tree} *(N')*
  record the search result as $[c_i^1, c_i^2, \ldots, c_i^k]$
 # Map matching based on candidate nodes
for i in range(j):
  if $i = 1$ *or* $i = j$:
    set $u_i = c_i^1$
  else:
    for m in range$(k-1)$:
      for m' *in range*$(m+1, k)$:
        find the shortest path p_m *from* u_{i-1}, c_i^m *to* c_{i+1}^1
        find the shortest path $p_{m'}$ from u_{i-1}, $c_i^{m'}$ *to* c_{i+1}^1
        if the length of $p_{m'}$ < the length of p_m:
          $m = m'$
      set $u_i = c_i^m$
      set $p_{i-1} = p_{m'}$
return the mapped path $[p_1, p_2, \ldots, p_{j-1}]$

Map Matching Accuracy Evaluation Metric

In the context of map matching, accuracy is typically measured by the percentage of correct mapped segments of a road network (Tanaka et al. 2021). However, the authors noticed that such measurements ignore differences in road segment lengths; therefore, the authors proposed two accuracy measurements, as shown in the following:

$$\text{Segment Accuracy} = \frac{\text{Number of correct mapped road segments}}{\text{Number of mapped road segments of the trajectory}} \quad (1)$$

$$\text{Length Accuracy} = \frac{\text{Length of correct mapped road segments}}{\text{Length of mapped road segments of the trajectory}} \quad (2)$$

Results

This section presents the performance of the proposed map matching method over 100 aircraft taxiing records in ASDE-X data collected on September 5, 2017. The authors first characterize the proposed map matching method with different parameter settings to identify optimal parameter settings. Then, the authors show how the trajectory compression module can effectively reduce computation time for map matching. Last, the authors compare the computational efficiency and accuracy of the developed approach against two state-of-the-art map matching techniques.

Parameter Initialization

Three parameters in the proposed method influence map matching accuracy and computational performance: $\theta_{\min}$, ε, and number of candidate nodes k (Fig. 3). Because grid search is reliable in low-dimensional spaces and the parallelization of grid search is trivial (Bergstra and Bengio 2012), the authors first conducted a grid search with an arbitrary number of candidate nodes to find the optimal values of $\theta_{\min}$ and ε with 30 randomly selected trajectories at LAX. The average length and the average number of road segments in the 30 trajectories were 5,785.04 m and 60.03, respectively. The second part tests the different numbers of candidate nodes in the algorithm and finds the acceptable number of candidate nodes with another randomly selected 10 trajectories. The average length of these 10 trajectories was 5,645.60 m, and the average segment number was 56.70. Multiple trials with different parameter settings were conducted. The following illustrates the results of these trials.

Based on the mechanism of the improved DP algorithm, the larger the values of ε and $\theta_{\min}$, the fewer spatial records maintained after trajectory compression. However, if we set ε and $\theta_{\min}$ overly large, crucial geometric information may be lost, such as turning points in the spatial records. To find acceptable parameter settings, we used a grid search to exhaustively explore combinations of parameter values between 1 and 40. The interval settings in the grid search were around 5. To ensure that the proposed map matching method had multiple candidate nodes in Algorithm 3, we set the number of candidate nodes to five. After the grid search, we conducted several trials to explore if five candidate nodes were enough for map matching for LAX.

Fig. 8 provides the comprehensive grid search results on ε and $\theta_{\min}$ with the 30 trajectories. According to Quddus et al. (2007), the accuracy level should be above 0.95 to support transportation applications with map matching results. Figs. 8(a and b) show the length accuracy and segment accuracy for the proposed map matching method with five candidate nodes, respectively. As shown in Fig. 8, length accuracy and segment accuracy are both above 0.95 within the circled range of ε and $\theta_{\min}$; this qualifies for map matching usage. The large range of suitable combinations shows that the proposed map matching algorithm can be robust over different combinations of parameters. To maintain a high accuracy of map matching results, the authors used the settings of ε and $\theta_{\min}$ with the highest map matching results. A larger value of $\theta_{\min}$ could potentially provide less compressed spatial records, which could help speed up the map matching process. Therefore, the authors used $\theta_{\min} = 10$, and $\varepsilon = 10$ in the following.

Multiple candidate nodes can increase map matching accuracy, because more candidate nodes can provide more feasible road segments for matching. Because computational complexity is $O(k \cdot n)$ (Table 3), more candidate nodes require more computation resources. To test if five candidate nodes are enough for map matching, the authors conducted map matching with different candidate nodes from 1 to 10 with another randomly picked 10 trajectories. The detailed map matching results from 1 to 5 candidate nodes are shown in Fig. 9. Fig. 9 also provides average accuracy and processing time for candidate nodes from 1 to 10. Computation time measures the computational complexity for generating candidate nodes and map matching based on these candidate nodes. The results in Fig. 9 show that the proposed algorithm with more than one candidate node had higher accuracy. Five candidate nodes are acceptable for the proposed map matching method, because this setting

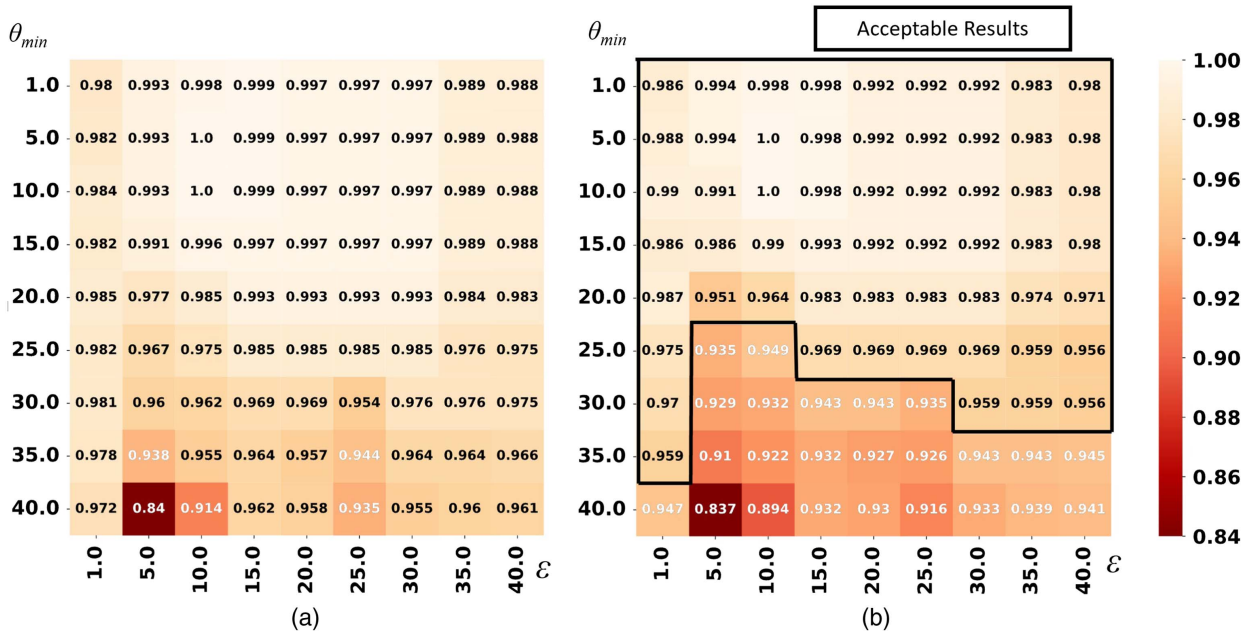

Fig. 8. Map matching accuracy over different parameters: (a) segment accuracy; and (b) length accuracy.

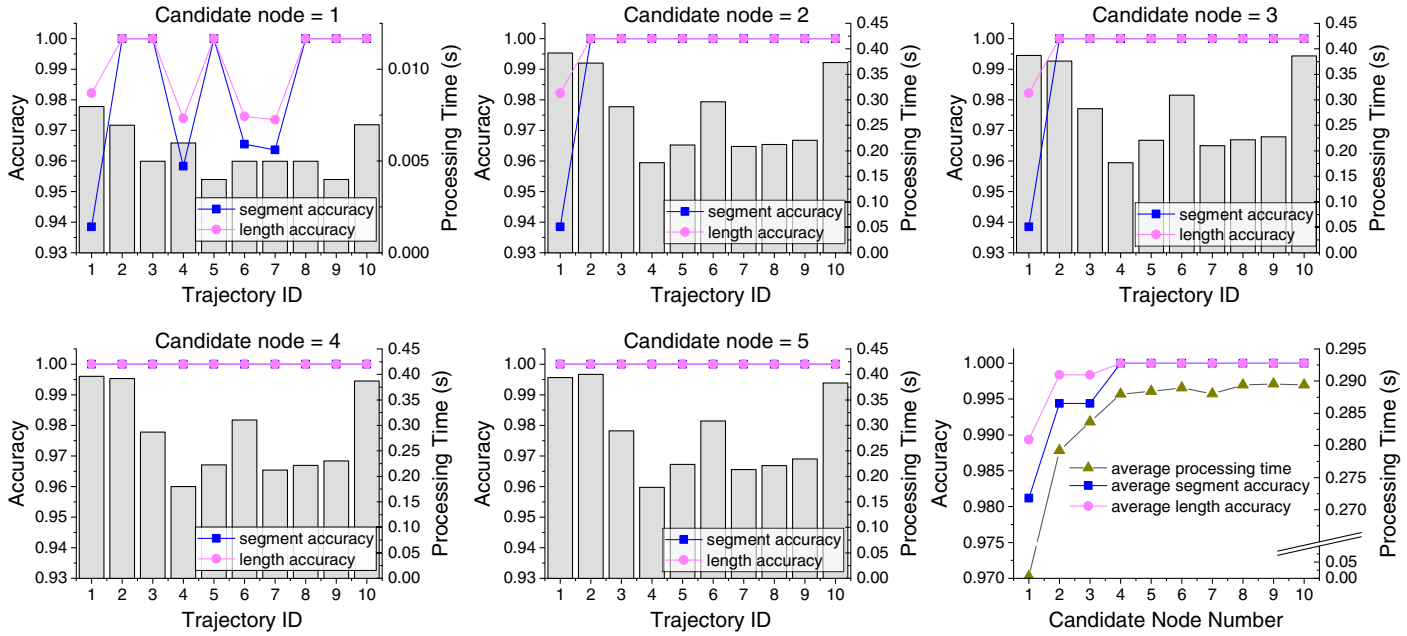

Fig. 9. Map matching accuracy with different values for candidate nodes.

achieved high accuracy. Although the computation time increases with the number of candidate nodes, the increment is negligible, as shown in the last graph in Fig. 9.

Effectiveness of Compression of Trajectory Records

This section visualizes the performance of the trajectory compression module (Algorithm 1). As shown in Fig. 10, the trajectory compression module selects less than 40 spatial records from thousands of spatial records for each trajectory. The selected spatial records keep the geometric trends of the original spatial records with much less memory.

Fig. 11(a) shows that the original ASDE-X records were dense and noisy. In Fig. 11, the black lines denote airport roads, and the dots show an example of one aircraft trajectory from the ASDE-X records. The number of data points obtained from the original ASDE-X records was 2,517. Fig. 11(b) shows the spatial data of the compressed trajectory. Using Algorithm 1, the number of spatial data points decreases to 22. Therefore, the critical spatial records for the next map matching step shrink to less than 1% of the original trajectory records. The fictitious links between the identified critical spatial records are the shortest path connecting the compressed spatial records. The fictitious links capture the geometric trends of the original ASDE-X records, including self-knotting behavior,

J. Comput. Civ. Eng., 2023, 37(1): 04022040

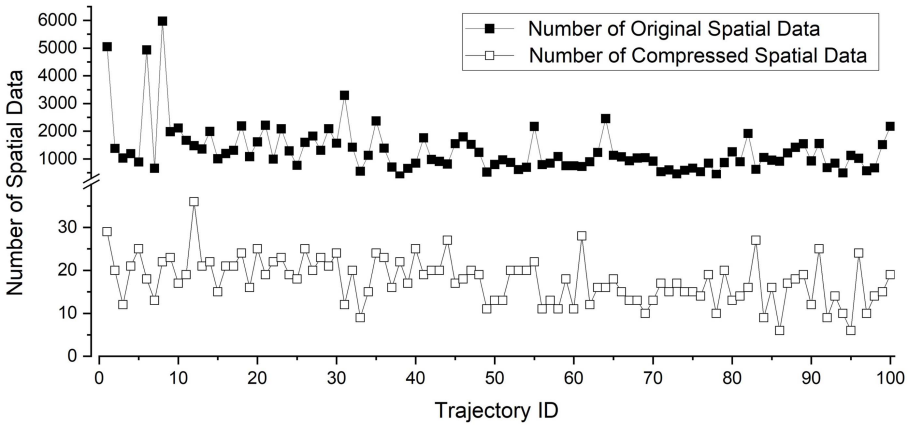

Fig. 10. Comparison of original spatial data and compressed spatial data.

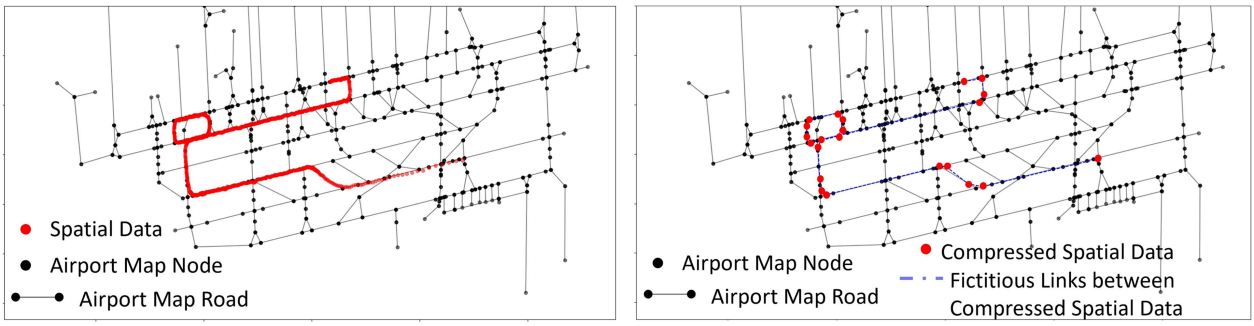

Fig. 11. Visualized comparison of original spatial data and compressed spatial data: (a) original spatial data for one trajectory (ASDE-X); and (b) compressed spatial data for one trajectory.

with much fewer data points. The map matching process requires much less in terms of computation resources and complexity than matching the original ASDE-X records.

Trajectory compression requires manual settings for ε and minimum angle in lines 7 and 14 of Algorithm 1, respectively. Fig. 11(b) shows the compressed result using $\varepsilon = 10$ m and a minimum angle of 10 degrees. The original average spatial records number was 1,305.83, and the number of compressed spatial data points was 17.69. The proposed compression module effectively reduced the number of spatial records to 1.35% of the original; this can efficiently help speed up the map matching process.

Computational Complexity Analysis

This section compares the computation efficiency of the proposed map matching method with a recently published map matching algorithm called "the Map Matching Algorithm for Ground Movement"(MAGM) (Tran et al. 2020). The MAGM algorithm uses change of heading direction and distance to each road segment to find the route underlying a sequence of spatial records (Tran et al. 2020). The MAGM algorithm selects multiple candidate nodes for each spatial data point and calculates the score of every possible route that links the candidate nodes. The algorithm works well when input spatial data is limited (e.g., dozens of input spatial data). Therefore, the MAGM algorithm must resample spatial data by uniform sampling. However, this method does not work well on the time scale needed for a large airport with long taxiing trajectories. As shown in Fig. 12, the MAGM algorithm would need nearly 245 seconds to process a trajectory with eight data points. For a long trajectory at an airport with complex geometry, such as the trajectory shown in Fig. 11, such a low number of data points could hardly capture a trajectory's geometric shapes. Considering the long computation time the MAGM algorithm requires, the comparison between the MAGM algorithm and the new approach could

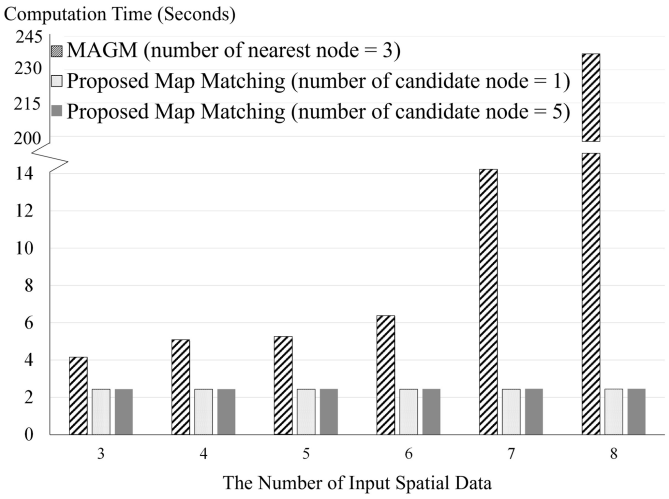

Fig. 12. Computation time comparison between MAGM and the proposed map matching method.

only be conducted on a small-scale—up to eight spatial records as the inputs.

In addition, the computation time of the MAGM algorithm grows exponentially, because this method needs to calculate over all possible mapping routes. For example, if the setting for candidate nodes is three and the number of input spatial data is n, the MAGM algorithm needs to calculate at least $O(n^3)$ times to find the correct mapping route. In addition, the MAGM algorithm also requires additional aircraft movement constraints to guarantee correct mapping results. However, some of these constraints are not applicable at large airports. For example, one aircraft will never pass by a map node more than one time. A counterexample is shown in Fig. 11, in which the aircraft circled in the taxing phase.

The proposed map matching algorithm works on linear time with regard to input spatial data (Fig. 12). If the number of candidate nodes increases, computation time does not grow exponentially. The number of candidate nodes is the value of k in line 10 of Algorithm 3. If the input spatial data number is n, the computation times for the proposed map matching with one candidate node and five candidate nodes are $O(n)$ and $O(5n)$, respectively. As shown in Fig. 12, the increment of computation time with respect to the input spatial records is slight in the proposed map matching method. For the 100 aircraft taxiing records shown in Fig. 10, the average map matching time was about 2.8 s with five candidate nodes. The proposed map matching can easily handle more candidate nodes to increase accuracy and maintain computation efficiency.

Comparative Analysis of the Map Matching Accuracy of Multiple Algorithms

This section first compares accuracy performance between the proposed map matching method and spatial temporal (ST) mapping (Lou et al. 2009). Then, this section discusses the robustness of the proposed map matching method with different parameter settings. Fig. 13 shows the accuracy performance for 100 trajectories with their trajectory lengths. The results show that the proposed map matching method mostly had higher accuracy than the ST mapping method. The proposed map matching method with five candidate nodes can achieve 100% accuracy. Table 4 shows the average accuracy results for 100 trajectories for different map matching techniques. For the 100 LAX airport aircraft trajectories, the proposed map matching method with five candidate nodes was able to achieve full accuracy, but the ST mapping method only achieved 20.5% and 21.4% in segment accuracy and length accuracy, respectively. The segment accuracy and length accuracy improvements were 79.5% and 79.6%, respectively.

We visualized the mapping results of different map matching methods to analyze incorrect mapping results. Specifically, we chose the 40th and 54th trajectories in Fig. 13 for visualization and denoted them as aircraft i and j, respectively. In Fig. 14, the black lines show airport roads, and the dots represent the original ASDE-X spatial data. The different shaded lines represent true map-matched road segments, map matching results based on the ST map matching method, and map matching results generated by the proposed method with one candidate node and five candidate nodes.

As shown in Figs. 14(c and d), the performance of the ST mapping method generates many matching errors around the junction areas of the map. ST mapping tries to tackle map matching of spatial data around intersections through the following computations: (1) locating the first spatial record data at the nearest road (r_1), (2) calculating the distance between the first spatial record data point and the second spatial record data point (d_1), (3) calculating the length between the kth candidate road and the first mapped road (l_k), and (4) selecting the road that has the smallest value

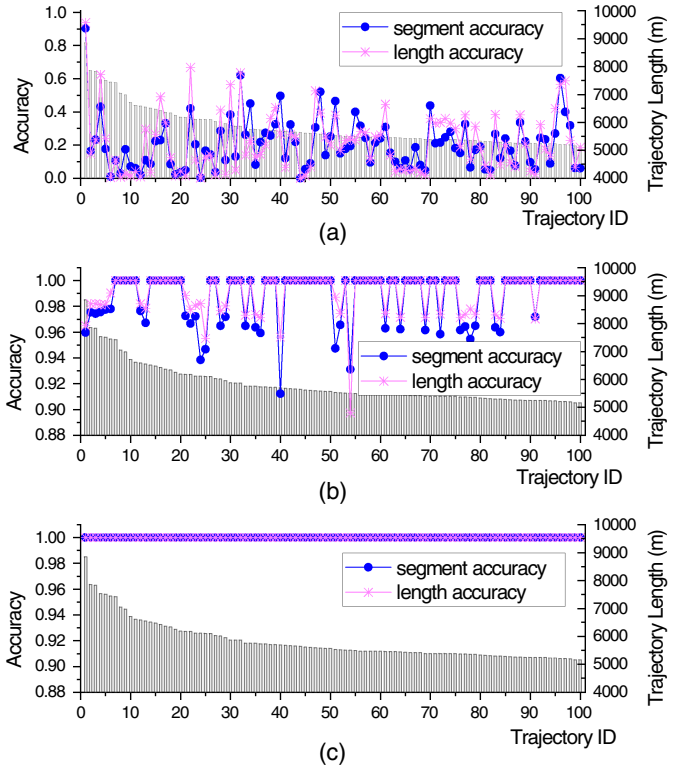

Fig. 13. Accuracy comparison between the proposed method and ST mapping: (a) ST map matching algorithm; (b) proposed map matching algorithm (number of candidate nodes = 1); and (c) proposed map matching algorithm (number of candidate nodes = 5).

Table 4. Average accuracy comparison between the proposed method and ST mapping

Accuracy evaluation metric	ST mapping	Proposed map matching (candidate nodes = 1)	Proposed map matching (candidate nodes = 5)
Segment accuracy	0.205	0.987	1
Length accuracy	0.214	0.991	1

of $|d_1 - l_k|$. This algorithm works well when deviations of the spatial record data from the correct road are relatively small. However, if the spatial record data largely deviate from the correct roads, the algorithm generates matching errors, especially when we seek to speed up computation time and use simplified geometry representation of the digital map. For example, in Fig. 14(c), road segment 1 is a straight line instead of an arc. Road segment 1 should be the ground truth, while road segment 2 should not be included. The ST mapping method will always incorrectly select road segment 2 in the map matching results.

The proposed map matching method resolves the aforementioned issues. As shown in Figs. 14(e and f), the proposed map matching method with one candidate node accurately captures the expected mapping roads in most situations. However, this method may not work well at intersections connecting more than three roads. For example, in Fig. 14(e), road segment 3 should be the ground truth, while road segments 4 and 5 should not be included. For one candidate node, the proposed map matching method may inaccurately identify the map node between the road segments 4 and 5 as part of the map matching results due to noise

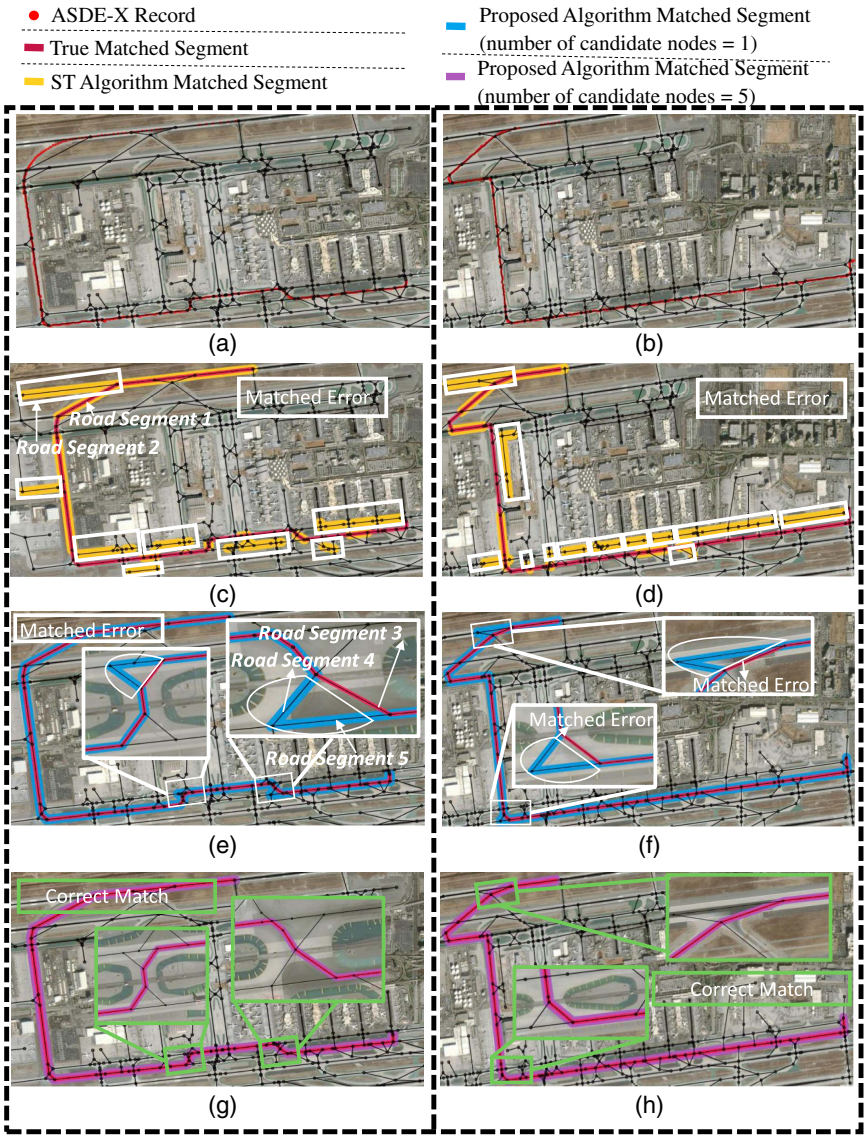

Fig. 14. Map matching result comparison between the proposed map matching method and ST mapping: (a) ASDE-X records for aircraft *i*; (b) ASDE-X records for aircraft *j*; (c) ST map matched for aircraft *i*; (d) ST map matched for aircraft *j*; (e) proposed method map matched for aircraft *i* (number of candidate nodes = 1); (f) proposed method map matched for aircraft *j* (number of candidate nodes = 1); (g) proposed method map matched for aircraft *i* (number of candidate nodes = 5); and (h) proposed method map matched for aircraft *j* (number of candidate nodes = 5). (Base map © OpenStreetMap contributors.)

in the spatial data. Model performance could be improved if more candidate nodes are used in the proposed map matching method, as shown in Figs. 14(g and h).

Fig. 15 shows segment accuracy and length accuracy over the 100 trajectories with different ε and $\theta_{\min}$ values from 1 to 40. The combinations of parameters in the circled range provide above 0.95 map matching accuracy for both segment and length criteria. The large range of acceptable parameter settings implies the robustness of the proposed map matching method.

Limitations and Future Research

The proposed map matching method has several limitations for future improvements. First, in order to improve map matching accuracy and reduce computation time, the proposed map matching method needs the spatial record data from the entire trajectory instead of using parts of the trajectory data. Second, the proposed method is designed for matching dense trajectories and applies compression algorithms to such massive records to reduce computational complexity. The proposed map matching may not guarantee accurate matching results for dealing with significantly sparse spatial records. In addition, this research used ASDE-X data as the spatial records. The high accuracy of the proposed method relies on accurate compressed spatial records. If the noise in the spatial data is high, the geometry of the compressed trajectory is affected, and the proposed map matching method may fail. Third, the proposed map matching method depends on the quality of the digital map. If the digital network misses too many road segments, map matching results may not be accurate. Last, although the proposed approach can perform well under a wide range of parameters, it still needs several trials to find optimal parameters in a new airport for the values of $\theta_{\min}$, ε, and the number of candidate nodes used.

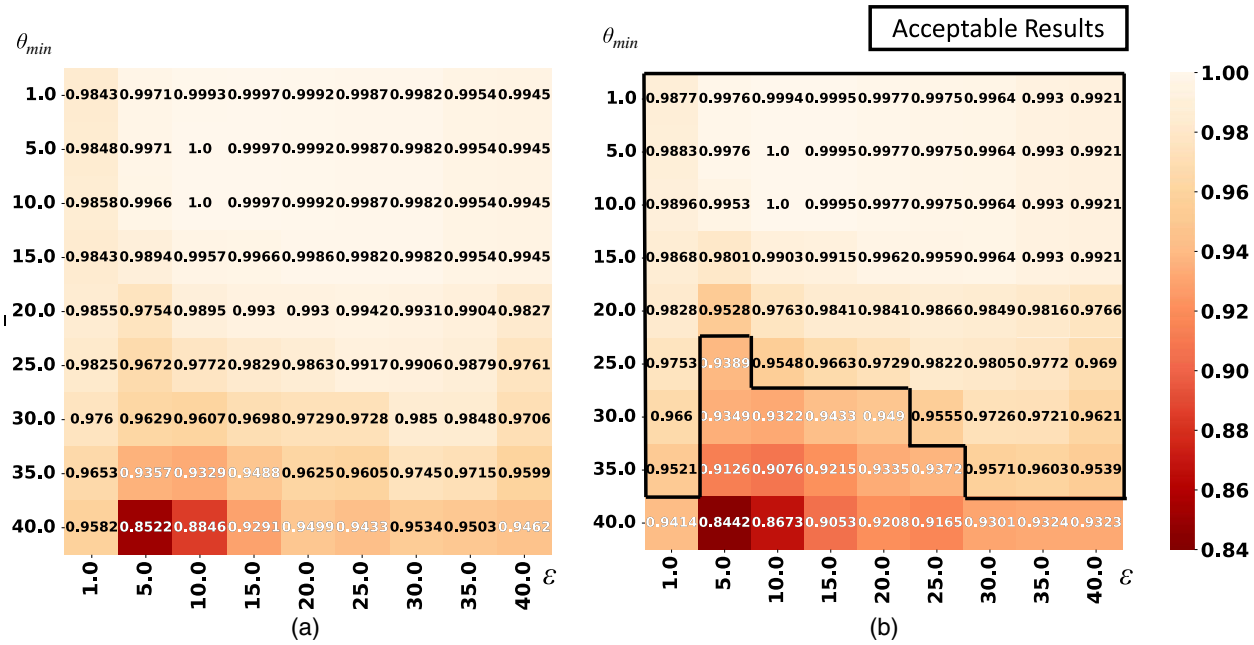

Fig. 15. Accuracy over 100 aircraft trajectories with different parameters (number of candidate nodes = 5): (a) segment accuracy; and (b) length accuracy.

In future work, the research team will improve the current work from the following aspects for efficiency and practical benefits. First, the research team will consider extending the algorithm to different vehicle trajectories. Currently, the research team has only studied aircraft trajectories on an airport without considering other types of vehicles running on the grounds of the airport. Second, the research team will consider integrating computer vision techniques in the visualization part to identify the outcomes of map matching automatically. In this way, the workload of manual checking could be reduced. Last, based on the proposed map matching approach, the research team could improve the predefined standard operations in an airport, such as predefined taxiing routes between runways and gates. Based on the proposed approach, the research team could efficiently and accurately recover airport traffic operation regularity patterns (such as frequently used routes between runways and gates). Comparing the recovered regularity patterns with predefined taxiing routes, airport managers could identify deviations between real-world taxiing operations and predefined routes. Furthermore, airport managers could identify deficiencies in predefined routes and adjust them to avoid these deviations in order to maintain the efficiency and safety of airport ground operations. In such real-world applications, airport managers may need additional technologies and protocols to address the challenges of sharing aircraft surveillance data, considering cyber security issues and business confidentiality concerns from multiple stakeholders in airport operations.

Conclusion

Map matching is critical for advanced applications for both long-term capacity management and short-term traffic flow analysis. For large-scale historical spatial records, a fast and reliable offline map matching method is needed. This research carried out map matching on 100 aircraft trajectories at LAX with ASDE-X data. The average length of these aircraft trajectories was 5,817.645 m, and the number of the average spatial records was 1,305.83. The proposed map matching method achieved full accuracy with nearly linear computational complexity for matching aircraft positioning records with ground transportation networks (i.e., the average processing time for each aircraft trajectory was 0.316 s). The proposed map matching method is promising for offline usage to process massive historical spatial records with noise.

Nevertheless, the proposed map matching method has several limitations, and the authors suggested possible directions for further improvements. In order to achieve high accuracy map matching results, the proposed map matching method requires restrictions on aircraft movements. The proposed map matching method may not achieve 100% accurate mapping results for vehicles with more flexible moving behaviors, such as movements that include U-turns or moving back and forth along one road. Future research would examine the performance of using the proposed map matching method for vehicle trajectories.

Data Availability Statement

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

Acknowledgments

This research was supported by funds from NASA University Leadership Initiative program (Contract No. NNX17AJ86A; Project Officer: Dr. Anupa Bajwa; Program coordinator: Koushik Datta; Principal Investigator: Dr. Yongming Liu; Co-Principal Investigator: Dr. Pingbo Tang), Airport Cooperative Research Program Graduate Research Awards Program (2021–2022) made by the Old Dominion University Research Foundation on behalf of the Virginia Space Grant Consortium, and a Dean's Fellowship from the College of Engineering (Project Officer: David A. Vey). This support is gratefully acknowledged. Map data is copyright OpenStreetMap contributors and is available from https://www.openstreetmap.org.

References

Barnett, A., M. Ball, G. Donohue, M. Hansen, A. Odoni, and A. Trani. 2015. "Collision course? The north airfield safety study at Los Angeles International Airport (LAX)." *Transp. Res. Part A Policy Pract.* 77 (May): 14–34. https://doi.org/10.1016/j.tra.2015.03.003.

Bellman, R. 1961. "On the approximation of curves by line segments using dynamic programming." *Commun. ACM* 4 (6): 284. https://doi.org/10.1145/366573.366611.

Bergstra, J., and Y. Bengio. 2012. "Random search for hyper-parameter optimization." *J. Mach. Learn. Res.* 13 (2): 281–305.

Dijkstra, E. W. 1959. "A note on two problems in connexion with graphs." *Numer. Math.* 1 (1): 269–271. https://doi.org/10.1007/BF01386390.

Dixit, A., and S. K. Jakhar. 2021. "Airport capacity management: A review and bibliometric analysis." *J. Air Transp.* 91 (10): 102010. https://doi.org/10.1016/j.jairtraman.2020.102010.

Douglas, D. H., and T. K. Peucker. 1973. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature." *Cartographica* 10 (2): 112–122. https://doi.org/10.3138/FM57-6770-U75U-7727.

FAA (Federal Aviation Administration). 2020. "Airport surface detection equipment, Model X (ASDE-X)." Accessed December 19, 2020. https://www.faa.gov/air_traffic/technology/asde-x/.

FAA (Federal Aviation Administration). 2021. "Traffic flow management system counts." Accessed February 2, 2021. https://aspm.faa.gov/tfms/sys/Airport.asp.

Giannotti, F., M. Nanni, F. Pinelli, and D. Pedreschi. 2007. "Trajectory pattern mining." In *Proc., 13th ACM SIGKDD Int. Conf.*, 330–339. New York, NY: Association for Computing Machinery.

Hashemi, M., and H. A. Karimi. 2014. "A critical review of real-time map-matching algorithms: Current issues and future directions." *Comput. Environ. Urban Syst.* 48 (4): 153–165. https://doi.org/10.1016/j.compenvurbsys.2014.07.009.

Knapen, L., T. Bellemans, D. Janssens, and G. Wets. 2018. "Likelihood-based offline map matching of GPS recordings using global trace information." *Transp. Res. Part C Emerging Technol.* 93 (5): 13–35. https://doi.org/10.1016/j.trc.2018.05.014.

Kong, Q.-J., Y. Chen, and Y. Liu. 2009. "A fusion-based system for road-network traffic state surveillance: A case study of Shanghai." *IEEE Intell. Transp. Syst. Mag.* 1 (1): 37–42. https://doi.org/10.1109/MITS.2009.932719.

Ku, S., H. Baik, and T. Kim. 2018. "Analysis of surveillance position error for airfield detection." *Aircr. Eng. Aerosp. Technol.* 90 (6): 962–966. https://doi.org/10.1108/AEAT-09-2017-0207.

Lee, D. T., and C. K. Wong. 1977. "Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees." *Acta Inform.* 9 (1): 23–29. https://doi.org/10.1007/BF00263763.

Liu, J., H. Li, Z. Yang, K. Wu, Y. Liu, and R. W. Liu. 2019. "Adaptive Douglas-Peucker algorithm with automatic thresholding for AIS-based vessel trajectory compression." *IEEE Access* 7 (Jan): 150677–150692. https://doi.org/10.1109/ACCESS.2019.2947111.

Liu, S., W. H. Chen, and J. Liu. 2016. "Robust assignment of airport gates with operational safety constraints." *Int. J. Autom. Comput.* 13 (1): 31–41. https://doi.org/10.1007/s11633-015-0914-x.

Lou, Y., C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang. 2009. "Map-matching for low-sampling-rate GPS trajectories." In *Proc., 17th ACM SIGSPATIAL Int. Conf.*, 352–361. New York, NY: Association for Computing Machinery.

Munaga, H., and V. Jarugumalli. 2011. "Performance evaluation: Ball-tree and KD-tree in the context of MST." In *Proc., Int. Joint Conf. Advanced Signal Process Information Technology*, 225–228. New York, NY: Springer.

NIST. 1993. "Integration definition of function modeling (IDEF0)." Accessed February 2, 2021. https://nvlpubs.nist.gov/nistpubs/Legacy/FIPS/fipspub183.pdf.

OpenStreetMap. 2017. "Planet dump retrieved from https://planet.osm.org." Accessed February 2, 2021. https://www.openstreetmap.org.

Pereira, F. C., H. Costa, and N. M. Pereira. 2009. "An off-line map-matching algorithm for incomplete map databases." *Eur. Transport Res. Rev.* 1 (3): 107–124. https://doi.org/10.1007/s12544-009-0013-6.

Quddus, M. A., W. Y. Ochieng, and R. B. Noland. 2007. "Current map-matching algorithms for transport applications: State-of-the art and future research directions." *Transp. Res. Part C Emerging Technol.* 15 (5): 312–328. https://doi.org/10.1016/j.trc.2007.05.002.

Ramer, U. 1972. "An iterative procedure for the polygonal approximation of plane curves." *Comput. Graphics Image Process.* 1 (3): 244–256. https://doi.org/10.1016/S0146-664X(72)80017-0.

Shen, Z., W. Du, X. Zhao, and J. Zou. 2020. "DMM: Fast map matching for cellular data." In *Proc., 26th Annual Int. Conf. MOBICOM*, 1–14. New York, NY: Association for Computing Machinery.

Song, I., I. Cho, T. Tessitore, T. Gurcsik, and H. Ceylan. 2018. "Data-driven prediction of runway incursions with uncertainty quantification." *J. Comput. Civ. Eng.* 32 (2): 04018004. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000733.

Stettler, M., S. Eastham, and S. Barrett. 2011. "Air quality and public health impacts of UK airports. Part I: Emissions." *Atmos. Environ.* 45 (31): 5415–5424. https://doi.org/10.1016/j.atmosenv.2011.07.012.

Stettler, M., G. Koudis, S. Hu, A. Majumdar, and W. Ochieng. 2018. "The impact of single engine taxiing on aircraft fuel consumption and pollutant emissions." *Aeronaut. J.* 122 (1258): 1967–1984. https://doi.org/10.1017/aer.2018.117.

Taguchi, S., S. Koide, and T. Yoshimura. 2018. "Online map matching with route prediction." *IEEE Trans. Intell. Transp. Syst.* 20 (1): 338–347. https://doi.org/10.1109/TITS.2018.2812147.

Tanaka, A., N. Tateiwa, N. Hata, A. Yoshida, T. Wakamatsu, S. Osafune, and K. Fujisawa. 2021. "Offline map matching using time-expanded graph for low-frequency data." *Transp. Res. Part C Emerging Technol.* 130 (Apr): 103265. https://doi.org/10.1016/j.trc.2021.103265.

Tobler, W. R. 1966. *Numerical map generalization*. Ann Arbor, MI: Univ. of Michigan.

Toledo-Moreo, R., D. Bétaille, and F. Peyret. 2009. "Lane-level integrity provision for navigation and map matching with GNSS, dead reckoning, and enhanced maps." *IEEE Trans. Intell. Transp. Syst.* 11 (1): 100–112. https://doi.org/10.1109/TITS.2009.2031625.

Tran, T. N., D. T. Pham, and S. Alam. 2020. "A map-matching algorithm for ground movement trajectory representation using A-SMGCS data." In *Proc., 2020 IEEE AIDA-AT Conf.*, 1–8. New York, NY: IEEE.

US Bureau of Transportation Statistics. 2020. "Passengers boarded at the top 50 U.S. airports." Accessed December 19, 2020. https://www.bts.gov/content/passengers-boarded-top-50-us-airports.

Velaga, N. R., and K. Pangbourne. 2014. "Achieving genuinely dynamic road user charging: Issues with a GNSS-based approach." *J. Transp. Geogr.* 34 (Sep): 243–253. https://doi.org/10.1016/j.jtrangeo.2013.09.013.

Velaga, N. R., M. A. Quddus, and A. L. Bristow. 2010. "Detecting and correcting map-matching errors in location-based intelligent transport systems." In *Proc., 12th WCTR*. Woodhouse, UK: World Conference on Transport Research Society.

Wang, S., L. Li, W. Ma, and X. Chen. 2019. "Trajectory analysis for on-demand services: A survey focusing on spatial-temporal demand and supply patterns." *Transp. Res. Part C Emerging Technol.* 108 (Jan): 74–99. https://doi.org/10.1016/j.trc.2019.09.007.

Wei, H., Y. Wang, G. Forman, Y. Zhu, and H. Guan. 2012. "Fast Viterbi map matching with tunable weight functions." In *Proc., 20th ACM SIGSPATIAL*, 613–616. New York, NY: Association for Computing Machinery.

Wilke, S., A. Majumdar, and W. Y. Ochieng. 2014. "Airport surface operations: A holistic framework for operations modeling and risk management." *Saf. Sci.* 63 (Oct): 18–33. https://doi.org/10.1016/j.ssci.2013.10.015.

Wu, Z., J. Xie, Y. Wang, and Y. M. Nie. 2020. "Map matching based on multi-layer road index." *Transp. Res. Part C Emerging Technol.* 118 (Feb): 102651. https://doi.org/10.1016/j.trc.2020.102651.

Zhang, M., Q. Huang, S. Liu, and H. Li. 2019. "Assessment method of fuel consumption and emissions of aircraft during taxiing on airport surface under given meteorological conditions." *Sustainability* 11 (21): 6110. https://doi.org/10.3390/su11216110.

Zhao, L., and G. Shi. 2018. "A method for simplifying ship trajectory based on improved Douglas–Peucker algorithm." *Ocean Eng.* 166 (May): 37–46. https://doi.org/10.1016/j.oceaneng.2018.08.005.