Received August 2, 2019, accepted August 12, 2019, date of publication October 2, 2019, date of current version October 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2945205

A Generalized Approach for Anomaly Detection From the Internet of Moving Things

JUNFENG TIAN[1,2], WEI DING[1], CHUNRUI WU[2], AND KWANG WOO NAM[1]
[1]Department of Computer and Information Engineering, Kunsan National University, Kunsan 54150, South Korea
[2]School of Information Science and Technology, Jiujiang University, Jiujiang 332005, China

Corresponding author: Kwang Woo Nam (kwnam@kunsan.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) grant through the Korea Government (MSIT) under Grant 2018R1A2B6007982, and in part by the Industry-University Cooperation Research and Development Program 2018 by the LX Spatial Information Research Institute.

ABSTRACT Internet of Moving Things are connected to a variety of different types of sensors to form a world of moving things, including people, animals, vehicles, drones, and boats, etc. As the data of collectible moving things continue to increase, anomaly detection of moving things has become an increasingly popular data mining task. Traditional trajectory outlier detection algorithms can detect common anomalies effectively, but it is hard to detect generalized anomalies, such as viewable direction anomalies, gravity anomalies, and magnetic field anomalies which can be collected by the accelerometer, gyroscope, magnetometer, and RPM sensor, etc. For this, we proposed a generalized approach for anomaly detection from the Internet of Moving Things, called the moving things outlier detection algorithm (*MTOD*). We propose the distance of moving things, which is equal to the weighted sum of the location distance and the multi-sensor distance, and then use the multi-sensor data generalization and moving things partitioning and anomaly detection three-step framework to detect the generalized anomaly. The experimental results show that our *MTOD* algorithm can detect moving things anomaly efficiency and accurately.

INDEX TERMS Internet of moving things, trajectory, anomaly detection, multi-sensor, normalization, quantization, and generalization.

I. INTRODUCTION

In recent years, the continuous development of the Internet of Things technology has promoted the progress of society effectively. The IoT (Internet of Things) —embedding wireless network connectivity, multi-sensors, and technology to traditionally non-smart everyday things — is slowly making the idea of the 'smart city' practically possible. Taking it a step further is the IoMT (Internet of Moving Things), connected moving things like vehicles, mobile phones, robots or mobile devices [1]–[3]. The IoMT encompasses moving things – cars, buses, trucks, trains, people, wearable devices, mobile phones, and tablets, etc. That can be tracked, exchange, or interact with bits of data via mobile network or Wi-Fi Internet connections. Moving things play a pivotal role in modern life, so it naturally follows that they would be connected to networks [4]–[7].

When we think of the IoT, we typically conjure objects in a fixed location: appliances, home automation systems, or individual stationery items or the Good Night Lamp. But some of the essential pieces of the IoT are likely to be the moving things: Those that move by themselves (cars, robots) and those that walk with us (mobile devices and wearables) [8], [9].

Consider the central role that moving things play in our modern life, like transporting people and goods–it makes sense that things such as cars, trucks, and trains, would become connected to networks. Find out more about location, fuel efficiency, and the relationship to other vehicles is of interest to both companies and people while being able to track data across multiple environments gives a fuller picture of our lives, which helping companies design and create products that make better sense for people. For example, Insurance companies can monitor real-time driving habits to give good drivers a discount on insurance (and, presumably, raise the rates of bad drivers). Audis Traffic Light Detection system adjusts the speed of the vehicle to coordinate with traffic lights, saving time, and fuel [10]–[14].

The Internet of Moving Things has collected the data of communication and the relationship between moving sensors

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Piccialli.

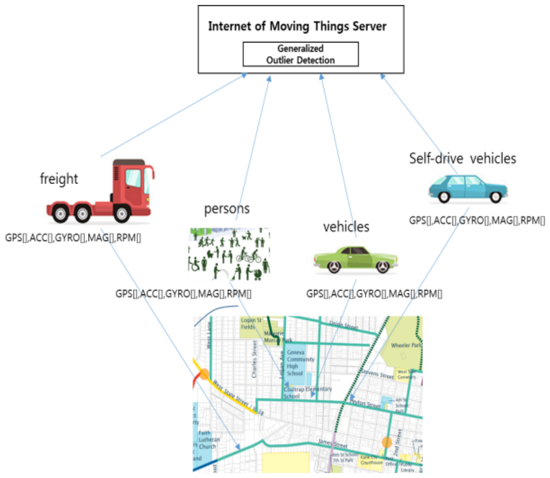

Internet of Moving Things Server
Generalized Outlier Detection
freight
GPS[],ACC[],GYRO[],MAG[],RPM[]
persons
GPS[],ACC[],GYRO[],MAG[],RPM[]
vehicles
GPS[],ACC[],GYRO[],MAG[],RPM[]
Self-drive vehicles
GPS[],ACC[],GYRO[],MAG[],RPM[]

FIGURE 1. An example of the Internet of Moving Things with different moving things, such types of things contains many GPS data, accelerometer data, gyroscope sensor data, magnetometer data, and RPM sensor data, etc.

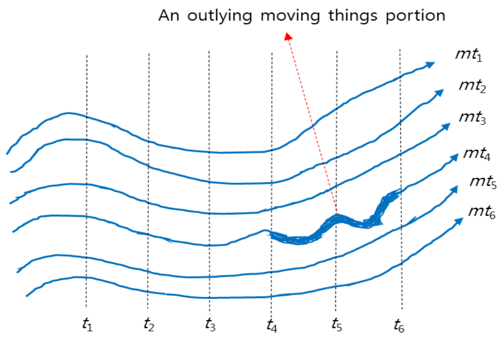


FIGURE 2. An example of an outlying moving things portion.

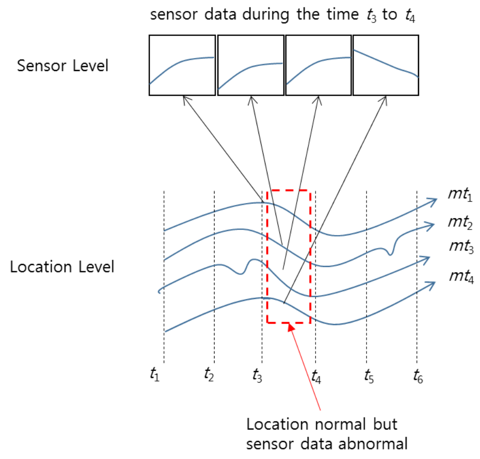


FIGURE 3. An example of moving thing mt_4 with location normal but sensor data abnormal.

and servers on the Internet of Things. In real life, moving things data generated rapidly, such as self-drive vehicles, peoples, freight, robots, and drones data. These moving things can collect a large amount of related trajectory data through different sensors, and upload them to the server through the network, and we can analyze and detect anomaly data, the example as shown in Figure 1. The Internet of Moving Things presents urban leaders today with enormous challenges, but also significant opportunities to tap into the mobile-technology boom to improve everything from city services to air quality-and offer new insights into public safety and long-term urban planning [15]–[17].

The location information is generated in the form of a trajectory, and at the same time, multi-sensor data is generated when the above moving things move. Human analysis of these moving things, especially the analysis of outliers, has important practical significance. People usually compare moving things as a whole, and the naive unit of outlier detection is the whole moving thing. In this way, we might not be able to detect outlying Portions.

Example 1: Consider the six moving things in Figure 2. It is obvious that the thick portion of a moving thing mt_4 is quite different from neighboring moving things in the time t_4 to t_6. However, the previous technique [18] cannot detect this unusual behavior since the differences are averaged out over the whole moving things; the overall behavior of the moving thing mt_4 is similar to those of the neighboring moving things. Thus, we miss this possibly important information.

The traditional trajectory outlier detection algorithm can detect anomaly based on the distance between the trajectories determined by the location information collected by GPS, which has been widely used in intelligent transportation, intelligent monitoring, and road condition warning and other application areas. GPS is a type of onboard sensor, and there are also many conventional sensors such as accelerometer, gyroscope, magnetometer, and RPM sensor, etc. If only the GPS data is taken into account and the other sensors data are ignored, some generalized anomaly may not be appropriately detected, such as snakes caused by drunk driver driving, deep pits on the road surface or theft of the utility hole cover.

Example 2: Consider the four moving things in Figure 3. It is obvious that the location of the four moving things are normal, but the sensor data of mt_4 is abnormal in the time t_3 to t_4. Thus, mt_4 can be detected as an outlying moving thing in our proposed algorithm, but the traditional algorithms may ignore it.

In this paper, the contributions are as follows:

(1) We proposed a generalization-partition-detection three-step framework, which generalized multi-sensor data first and then partitioning moving things, finally detect the anomaly. Among them, the generalization step is divided into quantization and normalization.

(2) In addition to detecting the anomaly of the location information of the moving things, we have proposed a new possibility of an anomaly, that is, multi-sensor data. Then we define the distance between the moving things equal to the weighted sum of the location distance and the multi-sensor distance.

(3) We proposed a generalized approach moving things outlier detection (*MTOD*) for anomaly detection from the Internet of Moving Things.

The rest of the paper is organized as follows. Section II discusses related work. Section III presents the problem statement. Section IV proposes our *MTOD* algorithm. Section V presents the results of the experimental evaluation. Finally, Section VI concludes the study.

II. RELATED WORK

The trajectory outlier detection algorithms are mainly classified into five categories: distribution-based [19]–[21], distance-based [22], [23], density-based [24], bias-based [25], and depth-based [26], [27]. For our study, we only review the distance-based.

A. DISTANCE FUNCTION

The distance-based trajectory outlier detection method mainly goes through three development stages: the detection of trajectory points, the detection of the entire trajectory, and the detection of the trajectory segments. The main difference between the three phases is that the primary objects considered in the test are different, which leads to differences in some test results [28]. In this paper, we mainly talk about the distance between two trajectory segments.

1) MBR DISTANCE FUNCTION

A distance measure for trajectory segments is based on the Minimum Bounding Rectangles (MBR) of segments [29]. The MBRs of two segments (L_i, L_j) are (B_1, B_2), each of which is described by the coordinates of the low bound point (x_l, y_l) and upper bound point (x_u, y_u). The MBR-Based distance $Dmin(B_1, B_2)$ is defined as the minimum distance between any two points from (B_1, B_2), calculated as:

$$\sqrt{\left(\Delta\left([x_l, x_u], [x_l', x_u']\right)\right)^2 + \left(\Delta\left([y_l, y_u], [y_l', y_u']\right)\right)^2} \quad (1)$$

Where the distance between two intervals is defined as:

$$\Delta\left([x_l, x_u], [x_l' x_u']\right) = \begin{cases} 0 & [x_l, x_u] \cap [x_l', x_u'] \neq \emptyset \\ x_l' - x_u & x_l' > x_u \\ x_l - x_u' & x_l > x_u' \end{cases} \quad (2)$$

The distance between L_i and L_j is 0 and $y_l' - y_u$, respectively.

2) Hausdorff Distance Function

Lee *et al.* [30] proposed another distance function, entitled Trajectory-Hausdorff Distance (D_{Haus}), which is composed of three components:

(i) The perpendicular distance (d_{pe}),

(ii) The parallel distance (d_{pa}),

(iii) The angle distance (d_α).

They are adapted from similarity measures used in the area of pattern recognition.

Then formally defining the three components through Eq. (3-5). Suppose there are two line segments $L_i = x_i y_i$ and $L_j = x_j y_j$. L_i is a longer line segments and L_j is a shorter one to without losing generality.

The perpendicular distance between L_i and L_j is defined as Eq. (3). Suppose the projection points of the points x_j and y_j onto L_i are p_x and p_y, respectively. L_{pe1} is the Euclidean distance between x_j and p_x; l_{pe2} is that between y_i and p_y.

$$d_{pe}(L_i, L_j) = \frac{l_{pe1}^2 + l_{pe2}^2}{l_{pe1} + l_{pe2}} \quad (3)$$

The parallel distance between L_i and L_j is defined as Eq. (4). Suppose the projection points of the points x_j and y_j onto L_i are p_x and p_y, respectively. L_{pa1} is the minimum of the Euclidean distances of p_x to x_i and y_i. Likewise, l_{pa2} is the minimum of the Euclidean distances of p_y to x_i and y_i.

$$d_{pa}(L_i, L_j) = Min(l_{pa1} + l_{pa2}) \quad (4)$$

The angle distance between L_i and L_j is defined as Eq. (5). Here, $||L_j||$ is the length of L_j, and α ($0 < \alpha < \pi$) is the smaller intersecting angle between L_i and L_j.

$$d_\alpha(L_i, L_j) = \begin{cases} \|L_j\| \times sin(\alpha), & if\ 0 \le \alpha \le \frac{\pi}{2} \\ \|L_j\|, & if\ \frac{\pi}{2} \le \alpha \le \pi \end{cases} \quad (5)$$

Finally defining the distance between two line segments as follows:

$$dist(L_i, L_j) = w_{pe} * d_{pe}(L_i, L_j) + w_{pa} * d_{pa}(L_i, L_j) + w_\alpha * d_\alpha(L_i, L_j) \quad (6)$$

where the weights w_{pe}, w_{pa} and w_α are determined depending on applications.

The Hausdorff distance function is one of the traditional distance calculation methods, which is usually used to calculate the distance between the trajectories. This paper will also use this method to calculate the location distance between moving things.

B. TRAJECTORY PARTITIONING

In many scenarios, we need to partition a trajectory into segments for a further process. The partitioning does not only reduce the computational complexity but also enables us to mine richer knowledge, such as sub-trajectory patterns, beyond what we can learn from an entire trajectory. In general, there are three types of partition methods.

The first category is based on ***the time interval***. If the time interval between two consecutive sampling points is more significant than a given threshold, a trajectory is divided into two parts at the two points. Sometimes, we can partition a trajectory into segments of the same time length.

The second category of methods is based on the ***shape of a trajectory***. We can partition a trajectory by the turning points with heading direction changing over a threshold. Alternative, we can employ the line simplification algorithms, such as the Douglas-Peucker algorithm, to identify the key points maintaining a trajectory's shape. The trajectory is then partitioned into segments by these key points. Similarly, Lee *et al.* [30], [31] proposed to partition a trajectory by using the concept of Minimal Description Language (MDL), which is comprised of two components: $L(H)$ and $L(D|H)$. $L(H)$ is the length, in bits, of the description of the hypothesis H; and $L(D|H)$ is the length, in bits, of the description of the data when encoded with the help of the hypothesis. The best hypothesis H to explain D is the one that minimizes the sum of $L(H)$ and $L(D|H)$. More specifically, they use $L(H)$ to denote the total length of partitioned segments, while letting $L(D|H)$ to represent the total (perpendicular and angle) distance between the original trajectory and the new partitioned segments. Using an approximation algorithm, they find a list of characteristic points that minimize $L(H) + L(D|H)$ from a trajectory. The trajectory is partitioned into segments by these characteristic points.

The third category of methods is based on the ***semantic meanings*** of points in a trajectory. A trajectory can be divided into segments, based on the stay points it contains. Whether we should keep the stay points in the split results depends on applications. For example, in a task of travel speed estimation, we should remove the stay points (from a taxi's trajectory) where a taxi was parked to wait for passengers [32]–[34]. On the contrary, to estimate the similarity between two users [30], [31], we can only focus on the sequences of stay points, while skipping other raw trajectory points between two consecutive stay points.

For trajectory outlier detection, Lee *et al.* [30] proposed a trajectory outlier detection algorithm based on the segment detection framework. Firstly, the trajectory is partitioned by the MDL method, and then detected the anomaly based on the Hausdorff distance of two trajectory segments. Finally, the abnormal trajectory is determined by the TRAOD algorithm according to the length of the abnormal trajectory segments and the trajectory length. Our proposed algorithm considers not only location information but also multi-sensor data between the two moving things segments based on the TRAOD algorithm, which can detect more generalized anomaly effectively.

III. PROBLEM STATEMENT

When the moving things move, location information and sensor data are generated. We designed the *MTOD* algorithm to detect the anomaly. Given a set of moving things data $\mathcal{MT} = \{mt_1, .., mt_n\}$, our algorithm generates a set of outliers $\mathcal{O} = \{o_1, \ldots, o_m\}$, with outlying moving things partitions for each outlier O_i, where n and m is the number of moving things and outliers respectively. The moving things, outlier, and outlying moving things partition are defined as follows.

The location information of moving things is generated in the form of the trajectory which is denoted as tr and $tr =< p_1, \ldots, p_i, \ldots, p_k >$, where p denotes the points and k is the number of points. Here $p_i = (x_i, y_i, t_i)$, where p_i is the point in the record time t_i, x_i and y_i are the longitude and latitude of point p_i respectively.

The multi-sensor data element ($msde$) of moving things is generated in the form of signal (i.e. wave) which is denoted as $msde = (msd, t)$, where the multi-sensor data $msd = \{sd_1, \ldots sd_i, \ldots, sd_l\}$, l is the number of multi-sensor in the moving things. Here $sd_i =< sd_{i_1}, \ldots, sd_{i_j}, \ldots sd_{i_h} >$ where

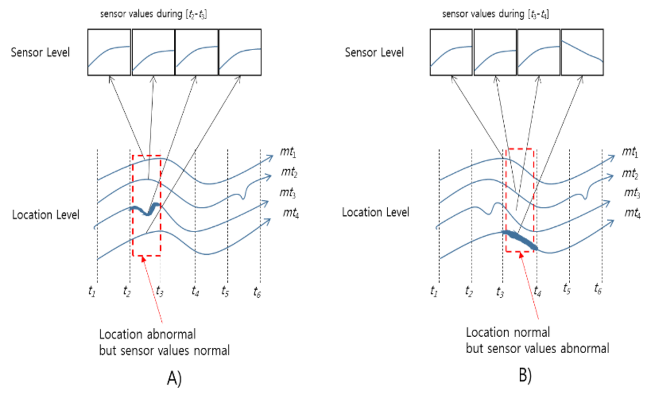

sensor values during [t_2-t_3]
Sensor Level
Location Level
mt_1
mt_2
mt_3
mt_4
t_1 t_2 t_3 t_4 t_5 t_6
Location abnormal but sensor values normal
A)
sensor values during [t_3-t_4]
Sensor Level
Location Level
mt_1
mt_2
mt_3
mt_4
t_1 t_2 t_3 t_4 t_5 t_6
Location normal but sensor values abnormal
B)

FIGURE 4. Two examples of location abnormal in part A) and sensor data abnormal in part B).

sd_i indicates the data of one of the multi-sensor and h is the number of sd_i, sd_{i_j} is the j-th data of i-th sensor in the record time st_{i_j}.

A moving thing is denoted as $mt_i = (tr_i, msd_i)$. In the moving things, location information and multi-sensor data all have a corresponding weight, and we define it as $w = (w_t, \{w_1, \ldots, w_l\})$, where w_t is the weight of location information trajectory and the weights w_1 to w_l are corresponding the sensor data sd_1 to sd_l.

The moving thing is divided into two parts: location information trajectory tr and multi-sensor data $msde$. For trajectory, the partition is a trajectory segment $p_x p_y$ $(x < y)$, where p_x and p_y are the points chosen from the same trajectory. A trajectory partition is called *t-partition* for short. For multi-sensor data, the partition is a sensor data segment $sd_{i_x} sd_{i_y}$ $(x < y)$, where sd_{i_x} and sd_{i_y} are the data chosen from the same sensor sd_i. A sensor data partition is called *s-partition* for short. In summary, moving things partition (*mt-partition*) include *t-partition* and *s-partition*, where *t-partition* and *s-partition* are the portions chosen from the same moving things.

An *mt-partition* is outlying if it does not have ''enough'' similar neighbors. An outlier is a moving thing that contains outlying moving things partitions.

Example 3: Consider the four moving things in Figure 4 A). It is obvious that the thick portion of a moving thing mt_4 is quite different from neighboring moving things at the location level. In Figure 4 B), it is obvious that the thick portion of a moving thing mt_4 is quite different from neighboring moving things at the sensor level. If using the previous algorithm, we will miss the important multi-sensor data and can't detect the anomaly in Figure 4 B).

Figure 5 shows the overall procedure of moving things outlier detection. First, in the multi-sensor data pre-processing phase, the multi-sensor data is quantized then normalized, and a base sensor will be selected with the maximum weight. Next, the moving things are partitioned into base units by MDL, and the outlier and outlying moving things are detected by using the proposed *MTOD* algorithm finally.

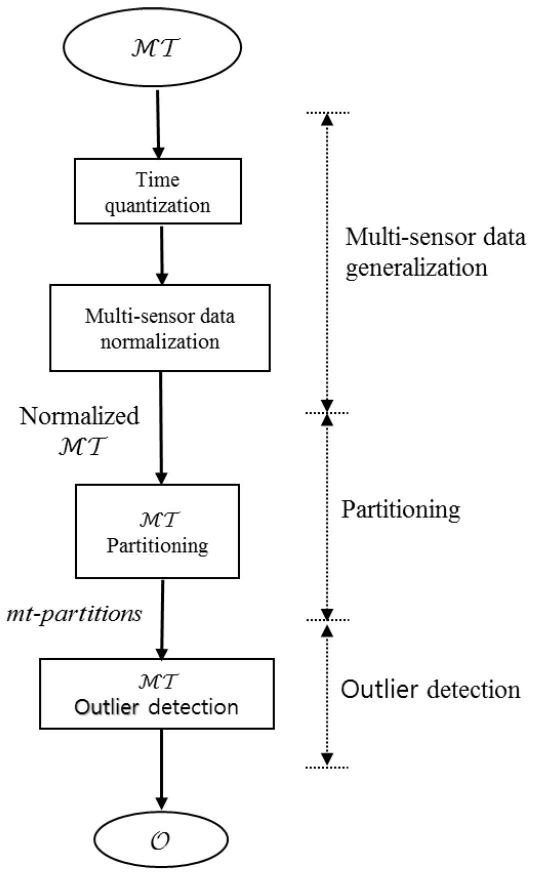

$\mathcal{MT}$
Time quantization
Multi-sensor data normalization
Normalized $\mathcal{MT}$
$\mathcal{MT}$ Partitioning
mt-partitions
$\mathcal{MT}$ Outlier detection
$\mathcal{O}$
Multi-sensor data generalization
Partitioning
Outlier detection

FIGURE 5. The overall procedure of *MTOD* algorithm.

TABLE 1. The notation for the moving tings outlier detection.

Symbol	Definition
w_s	The maximum weight from w_1 to w_l
$Len(M_i)$	The length of *mt-partition* M_i
$dist(M_i, M_j)$	The distance between M_i and M_j (see section II-A)
$CMT(M_i, d_t)$	The set of moving things is close to M_i

IV. MOVING THINGS OUTLIER DETECTION

A. DEFINITION OF MOVING THINGS OUTLIER DETECTION

Moving things outlier detection is defined mainly using distance. More specifically, an outlying *mt-partition* is identified based on the number of close moving things, which is determined by the distance from neighboring moving things. Before proceeding, we summarize the necessary notation in Table 1.

Definition 1: A moving thing is close to an *mt-partition* M_i means the distance between the moving things, and M_i is less or equal than d_t.

Here, d_t is the parameter given by a user.

Different sensor data will correspond to different weight; that's because each sensor data may have different data types or attributes in the applications.

We now define an outlying *mt-partition* in Definition 2.

Definition 2: An *mt-partition* M_i is outlying if Ineq. (7) is true. $|\ \mathcal{MT}\ |$ indicates the total number of moving things. Here, p is a parameter given by a user.

$$|CMT(M_i, d_t)| \leq \lceil (1-p)\,|\mathcal{MT}| \rceil \tag{7}$$

We then define an outlier in Definition 3. Intuitively, a moving thing becomes an outlier if the moving thing contains non-negligible (designated by f) outlying moving things segments. By definition 3, a moving thing with just a slight deviation is not included in the detection result.

Definition 3: A moving thing mt_i is an outlier if Ineq. (8) is true. Here, f is a parameter given by a user.

$$O_{fra}(mt_i) = \frac{\sum len(M_i)}{len(mt_i)} \geq f \tag{8}$$

Where M_i is the outlying segment.

B. MULTI-SENSOR DATA GENERALIZATION

1) TIME QUANTIZATION

The moving things consist of location information data and multi-sensor data, where the location information is expressed in the form of a trajectory, which is generated by GPS, and the recording time is usually every 1 second. The multi-sensor data is expressed in the form of a wave and the recording time is often every 20-50 milliseconds. In most cases, GPS data and multi-sensor data do not match precisely in the time level. The main reasons for the time mismatching have shown in Figure 6.

The first one is the different time density between GPS data and multi-sensor data in Figure 6 A). The second one is the different start time between GPS data and multi-sensor data in Figure 6 B). The third one is time missing in the multi-sensor in Figure 6 C).

Because of the above three reasons, the time mismatching between GPS data and multi-sensor data is caused. So we have to quantify the multi-sensor data to achieve time matching. In this paper, we use *linear interpolation* method to implement time quantization between GPS and multi-sensor.

We magnify the time mismatching part of Figure 6 A). For the time mismatching between the sensor data and the GPS data, we use the linear interpolation method to calculate the value of the sensor data in time matching points, which is shown in Figure 7.

In the sensor data, we can denote that the value of time match point X is sd_{t_i} and the amount of the next time on the left and right sides of X is $sd_{t_j} sd_{t_{j+1}}$ and.

The linear interpolation method can be expressed as Eq. (9).

$$sd_{t_i} = \frac{t_{j+1} - t_i}{t_{j+1} - t_j} * sd_{t_{j-1}} + \frac{t_i - t_j}{t_{j+1} - t_j} * sd_{t_j} \tag{9}$$

We can use the above method to calculate all the time match points on the sensor to quantify the sensor data from GPS.

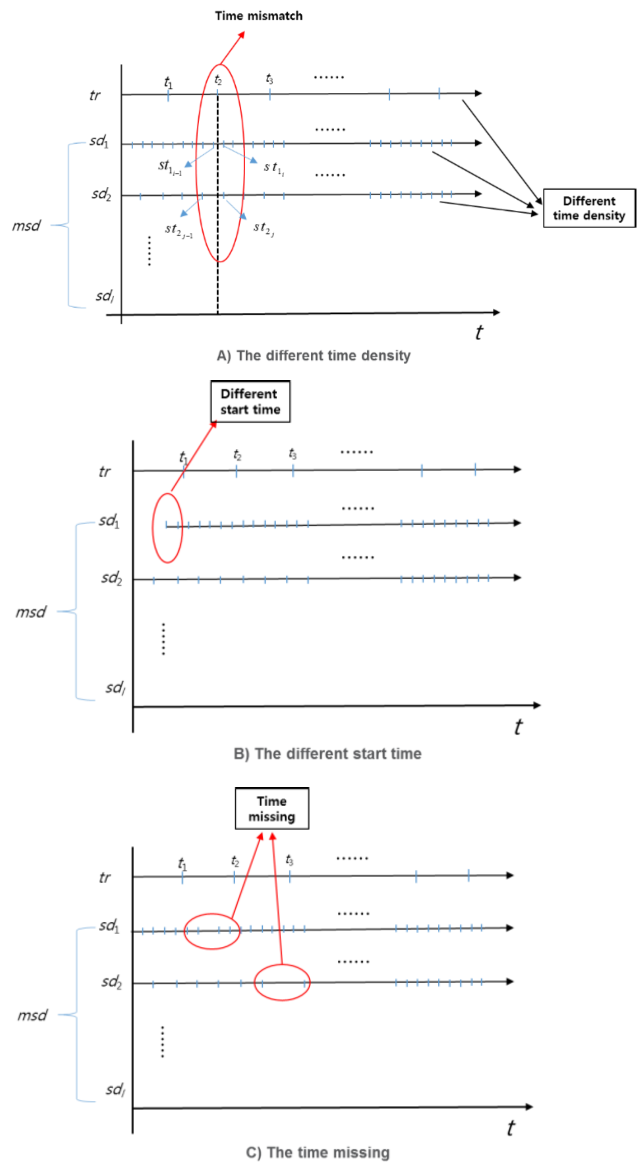


FIGURE 6. The examples of time mismatching between location information and multi-sensor data.

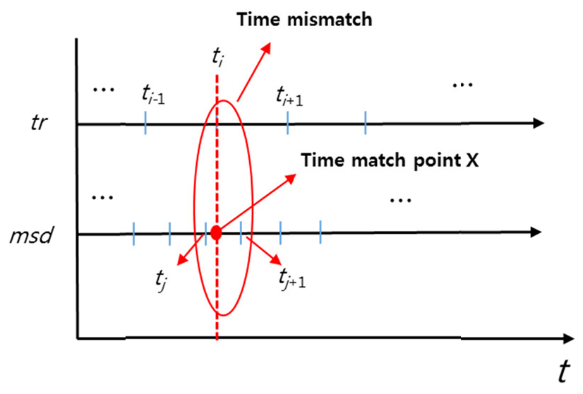


FIGURE 7. The example of time matching point X.

2) SENSOR DATA NORMALIZATION

On the internet of moving things, different multi-sensor may be used, and the data for each sensor may be a different standard. In order to standardize multi-sensor data to the same application platform, we must normalize them. We have noticed that the absolute data of some sensors are less than 0, positive or negative, and the reserved digits of decimal numbers are not the same. We have to pre-process these data to make them more accurate, complete, and consistent.

We will adopt data transformation to transform or unify the data into a form suitable for data mining. Here we take a normalized method to process data that aims at scaling attribute data to a specific interval. In other words, normalized data attempts to assign equal weight to all attributes, which is essential for distance-based classification and clustering (For example, our *MTOD* algorithm). There are three common normalization methods: min-max normalization, z-score normalization and decimal scaling normalization.

For sensor data normalization, we use the Z-score method, which is the number of standard deviations from the mean a data point is, as shown in Eq. (10).

$$Z = \frac{x-\mu}{\sigma} \tag{10}$$

where x is a raw score and μ is the mean of the population and σ is the standard deviation of the population.

After normalization, the value range falls within a small common interval $[-1, 1]$, thus avoiding the dependence on the unit of measure selection. Therefore, at the data level, the normalized multi-sensor data has the same Weights.

At the level of moving things, multi-sensor will have different weights depending on the focus, and we will select one of the most important sensor as the base sensor, based on the principle of maximum weight.

$$w_s = \max(w1, \ldots, w_l) \tag{11}$$

Using Eq. (11), we can know that the sensor will be chosen as the base sensor when the weight of this sensor is equal *to* w_s.

C. THE DISTANCE OF MOVING THINGS

We will denote the distance of the moving thing as *mt-dist*. The value of the *mt-dist* is equal to the weighted sum of the location distance (*loc-dist*) between two information trajectory segments *t-partitions* and the multi-sensor distance (*ms-dist*) between the two sensor data segments *s-partitions*. The *mt-dist* between two moving things segments is the primary tool for moving things outlier detection.

1) LOC-DIST

The location information of the moving things exists in the form of a trajectory, and the multi-sensor data can be used as the attribute of this trajectory, so the location distance between the moving things can be calculated using the distance between the trajectories mentioned in Section II A. The location distance can be expressed by the following Eq. (12).

$$loc-dist(T_i, T_j) = w_{pe} * d_{pe}(T_i, T_j) + w_{pa} * d_{pa}(T_i, T_j) + w_\alpha * d_\alpha(T_i, T_j) \tag{12}$$

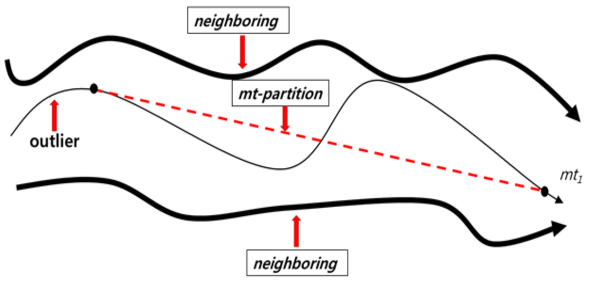


FIGURE 8. An example of missing possible outliers.

Where T_i and T_j denote the two information trajectory segments *t-partitions*.

2) MS-DIST

We partitioned the moving things into several moving things segments, as shown in Figure 4. The multi-sensor distance uses the multi-sensor data as mentioned above to calculate the absolute value of the difference in the corresponding moving things multi-sensor point.

The multi-sensor distance between the two moving things segments can be expressed by the following Eq. (13).

$$ms - dist\left(S_i, S_j\right) = w_{pe} * d_{pe}\left(S_i, S_j\right) + w_{pa} * d_{pa}\left(S_i, S_j\right) + w_\alpha * d_\alpha\left(S_i, S_j\right) \quad (13)$$

where S_i and S_j denote the two multi-sensor data segments *s-partitions*, and the weight of the chosen sensor is equal to w_s.

In summary, the moving things distance *mt*-distance can be defined as follows:

$$mt - dist(M_i, M_j) = w_t * loc - dist(T_i, T_j) + w_s * ms - dist(S_i, S_j) \quad (14)$$

where the weights t_w and w_s are determined depending on the applications.

Using definition 1, if the Ineq. (15) is true, we consider that *mt-partitions* M_i and M_j are close.

$$mt - dist\left(M_i, M_j\right) \leq d_t \quad (15)$$

D. DISCUSSION OF MOVING THINGS PARTITIONING

We now discuss the desiderata of moving things partitioning in our algorithm. In principle, any partitioning strategy, such as line simplification, can be exploited. However, careless partitioning (especially, in a long length) could miss possible outliers.

As mentioned above, we need to partition the moving things. The line segment after the partition, we call it *mt-partition*. The rule for partition is that *mt-partition* can't be too long and can't be too short also. Obviously, if the partition is too short, it is easy to increase the amount of calculation, and on the contrary, we may ignore possible outliers as shown in Figure 8.

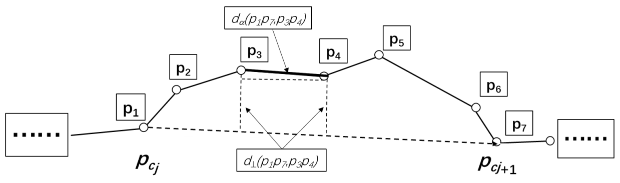


FIGURE 9. Formulation of the MDL method.

In this paper, we adopt the MDL method for the location information and multi-sensor data of the moving things.

We use the MDL method to partitioning the location information trajectory firstly. The MDL cost consists of two components: $L(H)$ and $L(D|H)$. Here, H means the hypothesis and D the data. The two parts are informally stated as follows [35]: "$L(H)$ is the length, in bits, of the description of the hypothesis; and $L(D|H)$ is the length, in bits, of the description of the data when encoded with the help of the hypothesis." The best hypothesis H to explain D is the one that minimizes the sum of $L(H)$ and $L(D|H)$.

The MDL method fits very well with our problem. A set of *t-partitions* corresponds to H, and a location information trajectory corresponds to D. Most importantly, $L(H)$ measures conciseness and $L(D|H)$ preciseness. Thus, finding the optimal partitioning translates to finding the best hypothesis using the MDL method. One advantage of this method is that it does not require any additional parameters as opposed to line simplification.

Figure 9 shows our formulation of $L(H)$ and $L(D|H)$. We formulate $L(H)$ by Eq. (16). $L(H)$ represents the sum of the length of a *t-partition*. On the other hand, we formulate $L(D|H)$ by Eq. (17). $L(D|H)$ represents the sum of the difference between a trajectory and a *t-partition*. For each *t-partition* $p_{cj}p_{cj+1}$, we add up the difference between $p_{cj}p_{cj+1}$ and P_kP_{k+1} ($c_j < k < c_{j+1} - 1$). To measure the difference, the sum of d_{pe} and d_α is used, but d_{pa} is not since a trajectory always encloses its *t-partitions*.

$$L(H) = \sum_{j=1}^{par_i - 1} \log_2(len(p_{c_j}p_{c_{j+1}})) \quad (16)$$

$$L(H) = \sum_{j=1}^{par_i - 1} \sum_{k=c_j}^{c_{j+1}-1} \{\log_2(d_\perp(p_{c_j}p_{c_{j+1}}, p_k, p_{k+1})) + \log_2(d_\alpha(p_{c_j}p_{c_{j+1}}, p_k, p_{k+1}))\} \quad (17)$$

When we find out the trajectory partitions using the MDL method, we can find the times of the partition points and then we can find the sensor data partitions according to the corresponding partition times. We can find out all the moving things partitions *mt-partitions* finally.

E. THE MTOD ALGORITHM

Table 2 shows our moving things outlier detection algorithm *MTOD*. This algorithm consists of three phases: multi-sensor data generalization, partition, and detection. In the multi-sensor data generalization phase, the algorithm quantizes

TABLE 2. The moving things outlier detection algorithm *MTOD*.

```
Algorithm: MTOD
Input: A set of moving things $MT=\{mt_1,\dots,mt_n\}$,
       parameters for trajectory: $w_t$, $d_t$
                 for multi-sensor: $w_1,\dots,w_l$
                 for outlier: $p$, $f$
Output: A set of outliers $O=\{o_1,\dots,o_m\}$
        with outlying $mt-partition$
ALGORITHM:
   /* I.MULTI-SENSOR GENERALIZATION PHASE */
01:Quantize multi-sensor data by the time in moving things
02:Normalize the multi-sensor data in moving things
03:Choose a base sensor with the maximum weight $w_s$
   /*II.PARTITIONING PHASE*/
04: for each $mt_i \in MT$ do
05:     Partition $mt_i$ at a base unit by MDL
   /*III. DETECTION PHASE*/
   /*$\Lambda$ denotes the set of moving things segments*/
06: for each $M_i \in \Lambda$ do
07:     count | $CMT(M_i, d_t)$ |
        by computing $mt-dist(M_i, M_j)$
08:     if | $CMT(M_i, d_t)$ | $\leq (1-p)$ | $MT$ | then
09:         Mark $M_i$ as outlying
10: for each $mt_i \in MT$ do
11:     if $O_{fra}(mt_i) \geq f$ then
12:         Output $mt_i$ with its outlying $mt-partition$;
```

multi-sensor data using linear interpolation method with GPS data firstly and then normalizes sensor data using Z-score method, at last, choose a base sensor with the maximum weight (lines 1-3). Second, the algorithm partitions moving things at a base unit by the MDL method (lines 4-5). Third, the algorithm detects outlying *mt-partition* and outlier (lines 6-11). The algorithm output the outlier with its outlying *mt-partition* finally (line 12).

V. EXPERIMENT EVALUATION

A. EXPERIMENTAL SETTING

We use real moving things data set: the BDD100K data set. The BDD100K data set is a kind of a large-scale, diverse driving video database. It has the most extensive and most varied open driving video dataset so far, and all the information from the data were collected by a real driving platform for computer vision research.

As suggested in the name, this dataset consists of 100,000 videos. Each video is about 40 seconds long, 720p, and 30 fps. The videos also come with GPS/IMU information recorded by cell-phones to show rough driving trajectories. All videos were collected from diverse locations in the United States. Our database covers different weather conditions, including sunny, overcast, and rainy, as well as varying times of the day, including daytime and nighttime. Table 3 below summarizes comparisons with previous datasets, which shows our dataset is much larger and more diverse.

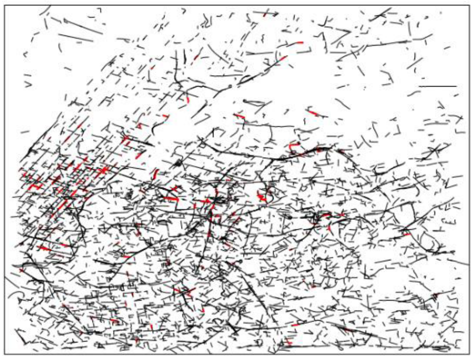


FIGURE 10. The result for BDD100K in the TRAOD algorithm.

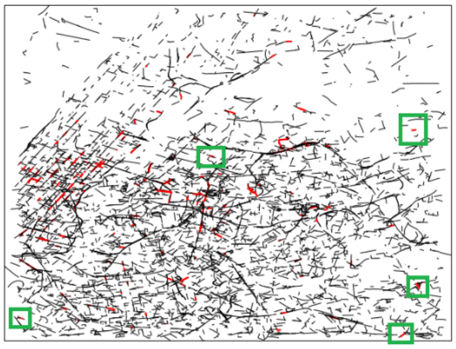


FIGURE 11. The result for BDD100K in the *MTOD* algorithm.

We extract the GPS data, accelerometer data, and gyroscope data (total six sensor data) from BDD100K for experiments. BDD100K has 100,000 moving things, and each moving thing has 40 points on average. We choose a small portion (10,000 moving things) of the data set in our experiment.

We put more weights on the angular outliers in experiments, so w_α is set to be ten times larger than w_{pe} or w_{pa} in the BDD100K data set.

We conduct all experiments on an Intel(R) Core(TM) i7-7500U-2.70GHz PC with 8 GBytes of main memory. We implement our algorithm and visual inspection tool in C++ using Microsoft Visual Studio 2019.

B. RESULTS FOR BDD100K DATA SET

Figure 10 and Figure 11 show the result, respectively, for a small portion (10,000) of the BDD100K data set in our proposed *MTOD* algorithm and the TRAOD algorithm. The parameters are set as follows: $d_t = 10, p = 0.95,$ and $f = 0.2$. Here, the red lines represent the anomaly moving things, and thin black lines represent normal moving things.

Compared with Figure 10 (113 outliers), we can see that there is a more generalized anomaly in Figure 11 (118 outliers), the area enclosed by the green rectangle. This is because we consider not only the location information but also the multi-sensor data. So more generalized anomaly will be detected by our proposed algorithm.

TABLE 3. The number of outliers detected between MTOD and TRAOD algorithm.

algorithm	2000	4000	6000	8000	10000
MTOD	33	50	75	97	118
TRAOD	32	48	71	92	113

TABLE 4. The accuracy of outliers detected between *MTOD* and TRAOD algorithm.

algorithm	2000	4000	6000	8000	10000
MTOD	0.81	0.79	0.83	0.85	0.84
TRAOD	0.80	0.76	0.79	0.82	0.82

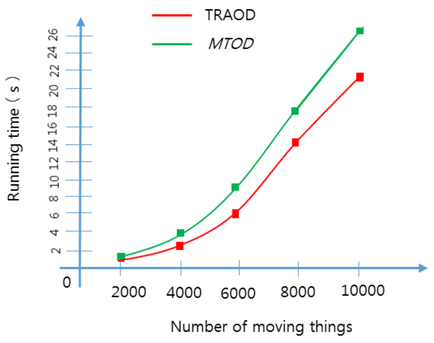


FIGURE 12. The comparison of running time between the TRAOD and *MTOD* algorithm.

C. THE COMPARISON OF RUNNING TIME AND ACCURACY BETWEEN MTOD AND TRAOD

Tables 3 and 4 compare the number of outliers, and the accuracy between *MTOD* and TRAOD algorithms in the case of randomly taking 2000, 4000, 6000, 8000, 10000 moving things in the BDD100K data set respectively. Compared with the TRAOD algorithm, our proposed algorithm can detect a more generalized anomaly whose multi-sensor data is abnormal, and our algorithm has higher accuracy than the TRAOD algorithm.

Figure 12 compares the running time of two different algorithms in the case of randomly taking 2000, 4000, 6000, 8000, 10000 moving things in the BDD100K data set. As we have seen, the running time of *MTOD* algorithm we proposed is longer than the TRAOD algorithm, because we not only consider the location information of the moving things but also the multi-sensor data. Although the running time of our *MTOD* algorithm is more than that of the TRAOD algorithm, only just a small increment, but there is a significant enhancement in the number of outliers and the accuracy of outliers detected. Experimental results prove that our *MTOD* algorithm is effective and accuracy.

D. EFFECTS OF PARAMETER VALUES

The weight of each sensor is determined by its "importance" in the moving things. Here, we mainly talk about the

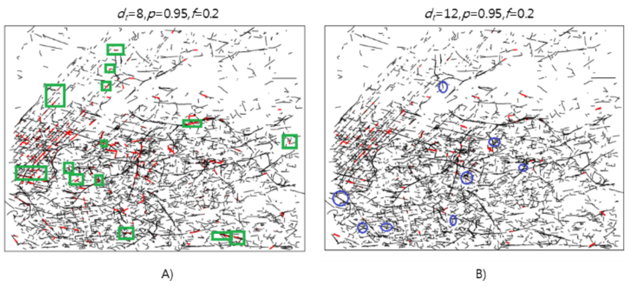


FIGURE 13. The comparison of moving things outliers when using different parameter d_t.

TABLE 5. The number of moving things outliers for BDD100K in MTOD depending on the values of the parameter d_t.

d_t(p=0.95)	8	9	10	11	12
MTOD	132	121	118	115	110

TABLE 6. The number of moving things outliers for BDD100K in *MTOD* depending on the values of the parameter p.

p(d_t=85)	0.92	0.93	0.94	0.95	0.96	0.97	0.98
MTOD	124	121	121	118	115	112	112

parameters d_t and p. Figure 13 compares the results for the BDD100K (10000 portions) data set when using different parameter d_t. Compared with Figure 11, a large number (14) of outliers are detected in Figure 13 A), which enclosed by the green rectangle. Whereas in Figure 13 B), there are less outliers (8) than the outliers in Figure 11, which enclosed by the blue ellipse.

Table 5 shows the number of moving things outliers for BDD100K when using different values of the parameter d_t in *MTOD* algorithm. The small portion is 10,000 moving things of the BDD100K data set. We can find that the number of moving things outliers increases as the value of d_t decreases.

Table 6 shows the number of moving things outliers for BDD100K when using different values of the parameter p in *the MTOD* algorithm. The small portion is 10,000 moving things of the BDD100K data set. We can find that the number of moving things outliers increases as the value of d_t decreases. While varying the value of p, the number of moving things is varied in a stair-like fashion.

VI. CONCLUSION

In this paper, we proposed an improved algorithm *MTOD* to detect the generalized anomaly, which considers not only the location trajectory but also the multi-sensor data. This algorithm includes a generalization-partition-detection three-step framework, which generalized multi-sensor data first and then partitioning moving things, finally detect the anomaly. The experimental results show that our *MTOD* algorithm can detect moving things anomaly efficiency and accurately.

This work is just the first step, and there are many challenging issues. First of all, according to our algorithm, each sensor data anomaly detection uses a different parameter, and involving too many parameters will affect the accuracy of the algorithm. The next step will be to delve into the relationship between these parameters and how to effectively reduce the parameters. Second, our algorithm is not implemented online in real-time. The next question to solve is how to apply our algorithm online. We are currently investigating the specific issues as a further study.

REFERENCES

[1] J. Van den Boom. The Internet of Moving Things. Pop-Up City. Amsterdam, The Netherlands. Accessed: Aug. 18, 2016. [Online]. Available: https://popupcity.net/the-internet-of-moving-things/

[2] R. Laxhammar and G. Falkman, "Online detection of anomalous subtrajectories: A sliding window approach based on conformal anomaly detection and local outlier factor," in *Proc. IFIP. AICT*, vol. 382, 2017, pp. 192–202. doi: 10.1007/978-3-642-33412-2_20.

[3] H. Cai, G. Yu, W. A. Yang, and K. Lu, "Mining frequent trajectory patterns of WIP in Internet of Things-based spatial-temporal database," *Int. J. Comput. Integr. Manuf.*, vol. 30, no. 12, pp. 1253–1271, 2017. doi: 10.1080/0951192X.2017.1307522.

[4] R. Chandramouli and K. P. Subbalakshmi, *Artificial Intelligence for the Internet of Moving Things*. Accessed: Jul. 31, 2018. [Online]. Available: https://innovationatwork.ieee.org/artificial-intelligence-for-the-internet-of-moving-things/

[5] Y. Zhao and F. Ferrari, "Topological effects on the mechanical properties of polymer knots," *Phys. A, Stat. Mech. Appl.*, vol. 486, pp. 44–64, Nov. 2017. doi: 10.1016/j.physa.2017.05.015.

[6] D. Kumar, J. C. Bezdek, S. Rajasegarar, C. Leckie, and M. Palaniswami, "A visual-numeric approach to clustering and anomaly detection for trajectory data," *Vis. Comput.*, vol. 33, no. 3, pp. 265–281, Mar. 2017.

[7] S. Biswas and R. V. Babu, "Anomaly detection via short local trajectories," *Neurocomputing*, vol. 242, pp. 63–72, Jun. 2017. doi: 10.1016/j.neucom.2017.02.058.

[8] D. Saffer. *Smart Design*. Accessed: Oct. 24, 2013. [Online]. Available: https://smartdesignworldwide.com/ideas/internet-moving-things/

[9] X. Li, Z. Sun, D. Cao, and H. He, "Development of a new integrated local trajectory planning and tracking control framework for autonomous ground vehicles," *Mech. Syst. Signal Process.*, vol. 87, pp. 118–137, Mar. 2017. doi: 10.1016/j.ymssp.2015.10.021.

[10] Grant Imahara, Mouser Electronics, Mansfield, TX, USA. *Internet of Moving Things*. Accessed: Dec. 2, 2014. [Online]. Available: https://www.mouser.com/pdfDocs/EIT-Internet-of-Moving-Things.pdf

[11] E. Rastgoftar, M. R. A. Jannat, and B. Banijamali, "An integrated numerical method for simulation of drifted objects trajectory under real-world tsunami waves," *Appl. Ocean Res.*, vol. 73, pp. 1–16, Apr. 2018. doi: 10.1016/j.apor.2018.01.013.

[12] A. Gardi, R. Sabatini, and T. Kistan, "Multiobjective 4D trajectory optimization for integrated avionics and air traffic management systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 55, no. 1, pp. 170–181, Feb. 2019. doi: 10.1109/TAES.2018.2849238.

[13] D. R. MacAyeal and V. Barcilon, "Ice-shelf response to ice-stream discharge fluctuations: I. unconfined ice tongues," *J. Glaciology*, vol. 34, no. 116, pp. 121–127, 1988.

[14] R. Zhen, Y. Jin, Q. Hu, Z. Shao, and N. Nikitakos, "Maritime anomaly detection within coastal waters based on vessel trajectory clustering and Naïve Bayes classifier," *J. Navigat.*, vol. 70, no. 3, pp. 648–670, 2017. doi: 10.1017/s0373463316000850.

[15] R. Costa. *The Internet of Moving Things*. Accessed: Mar. 31, 2018. [Online]. Available: https://technologyandsociety.org/internet-moving-things/

[16] K. Reise, C. Buschbaum, B. H. Ttger, J. RickK, and M. Wegner, "Invasion trajectory of Pacific oysters in the northern Wadden sea," in *Marine Biology, International Journal on Life in Oceans and Coastal Waters*, vol. 164. Springer, Apr. 2017, p. 68. doi: 10.1007/s00227-017-3104-2.

[17] Y. Wang, K. Qin, Y. Chen, and P. Zhao, "Detecting anomalous trajectories and behavior patterns using hierarchical clustering from taxi GPS data," *ISPRS Int. J. Geo-Inf.*, vol. 7, no. 1, p. 25, 2018.

[18] H. Sommer, D. E. Jacob, R. A. Stern, D. Petts, D. P. Mattey, and D. G. Pearson, "Fluid-induced transition from banded kyanite- to bimineralic eclogite and implications for the evolution of cratons," *Geochimica Et Cosmochimica Acta*, vol. 207, pp. 19–42, Jun. 2017. doi: 10.1016/j.gca.2017.03.017.

[19] J. De Leeuw, J. Methven, and M. Blackburn, "Physical factors influencing regional precipitation variability attributed using an airmass trajectory method," *J. Climate*, vol. 30, no. 18, pp. 7359–7378, Sep. 2017. doi: 10.1175/JCLI-D-16-0547.1.

[20] K. Garri, F. Sailhan, S. Bouzefrane, and M. Uy, "Anomaly detection in RFID systems," *Int. J. Radio Freq. Identificat. Technol. Appl.*, vol. 3, no. 1, pp. 31–46, 2015. doi: 10.1504/IJRFITA.2011.039781.

[21] C. Scott, B. A. Wing, A. Bekker, N. J. Planavsky, P. Medvedev, S. M. Bates, M. Yun, and T. W. Lyons, "Pyrite multiple-sulfur isotope evidence for rapid expansion and contraction of the early Paleoproterozoic seawater sulfate reservoir," *Earth Planet. Sci. Lett.*, vol. 389, pp. 95–104, Mar. 2014. doi: 10.1016/j.epsl.2013.12.010.

[22] A. Sharifi, L. N. Murphy, A. Pourmand, A. C. Clement, E. A. Canuel, A. N. Beni, H. A. K. Lahijani, D. Delanghe, and H. Ahmady-Birgani, "Early-Holocene greening of the Afro-Asian dust belt changed sources of mineral dust in West Asia," *Earth Planet. Sci. Lett.*, vol. 481, pp. 30–40, Jan. 2018. doi: 10.1016/j.epsl.2017.10.001.

[23] B. Basso, S. Caron-Huot, and A. Sever, "Adjoint BFKL at finite coupling: A short-cut from the collinear limit," in *Journal of High Energy Physics*. Berlin, Germany: Springer, Jan. 2015, p. 27. doi: 10.1007/JHEP01(2015)027.

[24] M. Peichl, M. Gažovič, I. Vermeij, E. de Goede, O. Sonnentag, J. Limpens, and M. B. Nilsson, "Peatland vegetation composition and phenology drive the seasonal trajectory of maximum gross primary production," *Sci. Rep.*, vol. 8, no. 1, p. 8012, May 2018. doi: 10.1038/s41598-018-26147-4.

[25] S. Kim, S. Jeong, I. Woo, Y. Jang, R. Maciejewski, and D. S. Ebert, "Data flow analysis and visualization for spatiotemporal statistical data without trajectory information," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 3, pp. 1287–1300, Mar. 2018. doi: 10.1109/tvcg.2017.2666146.

[26] J. A. Morgan, "Interception in differential pursuit/evasion games," *J. Dyn. Game*, vol. 3, no. 4, pp. 335–354, 2017. doi: 10.3934/jdg.2016018.

[27] Y. Djenouri, A. Belhadi, J. C.-W. Lin, D. Djenouri, and A. Cano, "A survey on urban traffic anomalies detection algorithms," *IEEE Access*, vol. 7, pp. 12192–12205, 2019.

[28] T. Q. Huang, Y. Q. Yu, G. D. Guo, and X. L. Yu, "Trajectory outlier detection based on semi-supervised technology," *J. Comput. Res. Develop.*, vol. 48, no. 11, pp. 2074–2082, 2011. doi: 10.1007/s00466-010-0527-8.

[29] H. Jeung, M. L. Yiu, and C. S. Jensen, *Computing With Spatial Trajectories*. New York, NY, USA: Springer, 2011, pp. 143–177.

[30] J. G. Lee, J. Han, and K. Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proc. ACM SIGMOD Conf. Manage. Data*, Jun. 2007, pp. 593–604.

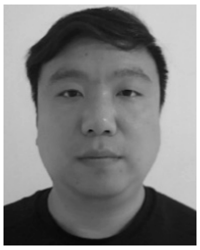
[31] J.-G. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 140–149.

[32] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-drive: Enhancing driving directions with taxi drivers' intelligence," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 220–232, Jan. 2013.

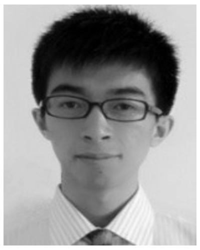
[33] R. Dasari, J. J. La Clair, and P. A. Kornienko, "Irreversible protein labeling by Paal-Knorr conjugation," *ChemBioChem*, vol. 18, no. 18, pp. 1792–1796, Sep. 2017. doi: 10.1002/cbic.201700210.

[34] L. Knorr-Held and G. Raßer, "Bayesian detection of clusters and discontinuities in disease Maps," *Biometrics*, vol. 56, no. 1, pp. 13–21, Mar. 2015. doi: 10.1111/j.0006-341X.2000.00013.x.

[35] X. Jian, Y. Jiang, C. Zeng, and T. Li, "Node anomaly detection for homogeneous distributed environments," *Expert Syst. Appl.*, vol. 42, no. 20, pp. 7012–7025, Nov. 2015. doi: 10.1016/j.eswa.2015.04.037.



JUNFENG TIAN is currently pursuing the Ph.D. degree with the Department of Computer and Information Engineering, Kunsan National University. His current research interests include spatio-temporal databases, data mining, and big data analysis.



WEI DING is currently pursuing the Ph.D. degree with the Department of Computer and Information Engineering, Kunsan National University. His current research interests include spatial–temporal databases, data mining, and geo-tagged data analytic.

CHUNRUI WU is currently pursuing the bachelor's degree in computer science and technology with Jiujiang University. Her current research interests include spatio–temporal databases, data mining, and deep learning.

KWANG WOO NAM received the Ph.D. degree in computer science from Chungbuk National University. After Ph.D., he studied location-based services and telematics at the Electronics and Telecommunications Research Institute of Korea. He is currently a Professor with the School of Computer, Information, and Communications Engineering, Kunsan National University, South Korea. His research interests include spatial and moving objects database, spatial big data, and GeoAI.