Real-time Vehicle Trajectory Clustering Using Geomagnetic Sensor Measurements

Guoqiang Mao
School of Transportation
Southeast University, Nanjing 210096, China
Email: g.mao@ieee.org

Zhaoying Hou, Corresponding Author
School of Telecommunications Engineering
Xidian University, Xi'an 710071, China
Email: 23011210459@stu.xidian.edu.cn

Maolin Peng
School of Telecommunications Engineering
Xidian University, Xi'an 710071, China
Email: 24011210845@stu.xidian.edu.cn

Tianxuan Fu
School of Telecommunications Engineering
Xidian University, Xi'an 710071, China
Email: futianxuan@stu.xidian.edu.cn

Word Count: 4592 words + 2 table(s) × 250 = 5092 words

Submission Date: July 29, 2025

ABSTRACT

Real-time vehicle tracking is important for traffic management in intelligent transportation systems (ITS). Geomagnetic sensors become a promising candidate for traffic measurements due to its robustness in adverse weather conditions, cost-effectiveness, and compact size. However, weak measurement signals due to small car size, noise and interference may cause missed or false detections, reducing tracking accuracy. This issue is further compounded by the wireless transmission losses caused when transmitting the measurement data through low-power wireless transmission technology such as LoRa. In lieu of the aforementioned challenges, this paper proposes a clustering algorithm for real-time vehicle tracking using ubiquitously deployed geomagnetic sensors. We first propose using directionality constraints, which can be inferred from geomagnetic sensor measurements, and dynamic energy management to handle trajectory fragmentation. On that basis, Bayesian optimization is employed for parameter tuning to enhance the robustness of trajectory estimation. The proposed algorithm is implemented using the real-world geomagnetic sensor network deployed on the Dongxiangxi Expressway in Jiangxi, China. Experimental results validate the superior performance of the proposed method to traditional methods like Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Kalman Filter, achieving an FMI score of 0.8683. The proposed method offers a cost-effective and robust solution for vehicle tracking in complex traffic environments, promoting geomagnetic sensor applications in ITS.

Keywords: Vehicle Tracking, Trajectory Estimation, Data Clustering, Geomagnetic Sensors

INTRODUCTION

Real-time vehicle tracking is important for traffic management, which may significantly enhance traffic flow, safety, and road efficiency across highways and urban road networks (*1*). In recent years, it has been widely applied in fields like real-time roadside vehicle tracking and monitoring (*2*).

Roadside perception systems commonly rely on radar and camera-based sensors to collect real-time traffic data for vehicle tracking (*3*). However, deploying these sensors for large-scale highway applications presents significant challenges, including high implementation costs and degraded performance under adverse environmental conditions. Cameras, widely used in traffic monitoring, employ computer vision algorithms but face accuracy limitations in low-light conditions and occlusion scenarios (*4*). Similarly, radar excels at penetrating fog and rain to detect objects but struggles with target separation in dense urban environments due to the multi-path effect (*5*). Therefore, there is an urgent need to develop novel sensing technologies or sensors to enable cost-effective and accurate vehicle tracking.

Geomagnetic sensors have gained prominence for vehicle tracking in ITS due to their robust performance in adverse weather, low cost, compact size, and ease of installation (*6*). These attributes make them well-suited for scalable traffic monitoring solutions. For instance, a geomagnetic sensor network deployed along lane markings (*7*), combined with a differential double-window vehicle detection algorithm (which identifies vehicles by comparing data from adjacent sensor time windows), achieves high-accuracy lane-level vehicle detection. Additionally, a distributed sensor network integrating Total Field Matching (TFM, a localization technique based on magnetic field pattern comparison) with a Kalman Filter enables precise vehicle localization (*8*). However, these approachs assume idealized conditions without environmental interference, limiting its applicability in complex scenarios. To address single-sensor limitations, multi-sensor fusion has been explored, such as a tunnel vehicle tracking system that combines millimeter-wave radar and geomagnetic sensors through asynchronous data fusion to enhance robustness in tunnel environments (*9*).

Despite technological advances, critical challenges persist in vehicular magnetic sensing systems, including missed detections from signature overlap in dense traffic and environmentally-induced false positives from metallic interference (*10*). To address signal overlap in dense traffic, a filtering framework has been introduced, improving detection for most vehicles. However, it remains ineffective for small vehicles with weak magnetic signatures, leading to frequent missed detection (*11*). Interference from adjacent lanes can also cause false signals, resulting in “ghost target” trajectories. While static methods using fixed thresholds and regions of interest (ROIs) struggle to adapt to dynamic traffic (*12*), increasing misclassification risk. In LoRa-based systems, although time-window compensation helps with short-term communication loss, trajectory updates are still unreliable under long interruptions or multiple-vehicle conditions, causing errors in tracking and reconstruction (*13*). The weak magnetic signals generated by small vehicles due to their lower metal mass, combined with environmental noise and interference from large vehicles in adjacent lanes, lead to frequent missed detections and false detections by geomagnetic sensors. Additionally, when using low-power wireless technologies (such as LoRa) for data transmission, communication signals are prone to interruption, which may cause tracking algorithms to prematurely terminate vehicle trajectories due to missing measurements, leading to the loss of tracking targets. Furthermore, interference from vehicles in adjacent lanes can cause false detections, generating measurements unrelated to actual vehicles and consequently producing entirely false "ghost target" trajectories.

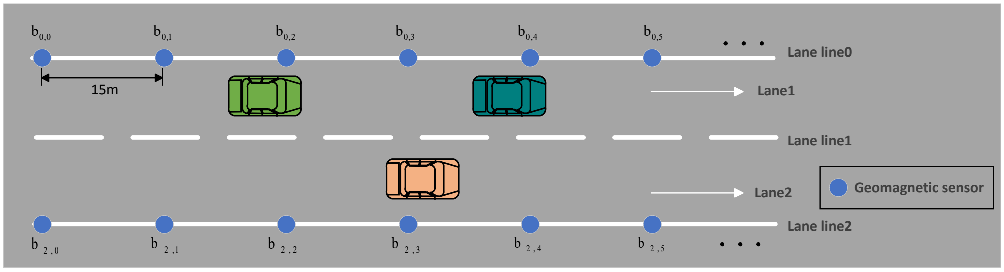


Figure 1 Schematic diagram of geomagnetic sensor deployment in dual lane scenario.

In lieu of the aforementioned challenges, we propose a clustering-based tracking algorithm that leverages ubiquitously deployed geomagnetic sensors. We first propose a method that utilizes directionality constraints, which are inferred from geomagnetic sensor measurements, together with dynamic energy management to handle trajectory fragmentation. On that basis, Bayesian optimization is employed for parameter tuning to enhance the robustness of trajectory estimation. Finally, we verify the effectiveness of our proposed scheme through the real-world geomagnetic sensor network deployed on the Dongxiangxi Expressway in Jiangxi, China.

The remainder of this paper is summarized as follows. Section 2 describes the system model. Section 3 introduces a real-time trajectory clustering algorithm with directionality constraints and dynamic energy management, alongside a Bayesian optimization framework for parameter tuning. Section 4 analyzes the algorithm's performance and implementation in real-world highway scenarios. Section 5 presents the conclusions.

SYSTEM MODEL

We consider a typical one-way, two-lane highway in China. In our study, geomagnetic sensors are deployed along the lane boundary lines on both sides of the road, evenly installed at intervals of $L_a = 15$ m, as shown in Figure 1. The positions of the k-th pair of geomagnetic sensors on lane lines 0 and 2 are denoted as $b_{0,k}$ and $b_{2,k}$, respectively.

When a vehicle passes over a geomagnetic sensor, the sensor is triggered and reports its measurements to the fusion center. Each reported measurement is treated as an individual sample in the clustering algorithm, denoted as s_j (where j indexes the arrival order of samples in time), and is represented as a four-dimensional feature vector:

$$s_j = [time_j, sensorID_j, Mpeak_j, lane_j] \tag{1}$$

$time_j$ denotes the timestamp when the geomagnetic sensor was triggered; $sensorID_j$ corresponds to the position of the geomagnetic sensor; $Mpeak_j$ represents the peak magnetic disturbance value caused by the vehicle passage, capturing the strongest moment of geomagnetic fluctuation; and $lane_j$ indicates the lane line number to which the geomagnetic sensor belongs. A sample such as $[1702631668170,\ 17,\ 625,\ 2]$ corresponds to a measurement reported by a geomagnetic sensor located at position $b_{2,17}$, which is the 17th geomagnetic sensor installed along lane line 2, indicating that the target is assumed to come from lane 2. The value 1702631668170 indicates the timestamp in milliseconds when the sensor was triggered, and 625 is the peak magnetic disturbance caused by the vehicle passage, recorded in *LSB*. Here, least significant bit (*LSB*) refers to the unit of digital output produced by the 16-bit analog-to-digital converter (ADC) embedded in the VCM5883L geomagnetic sensor used in our system, with a conversion ratio of 3000*LSB* per Gauss.

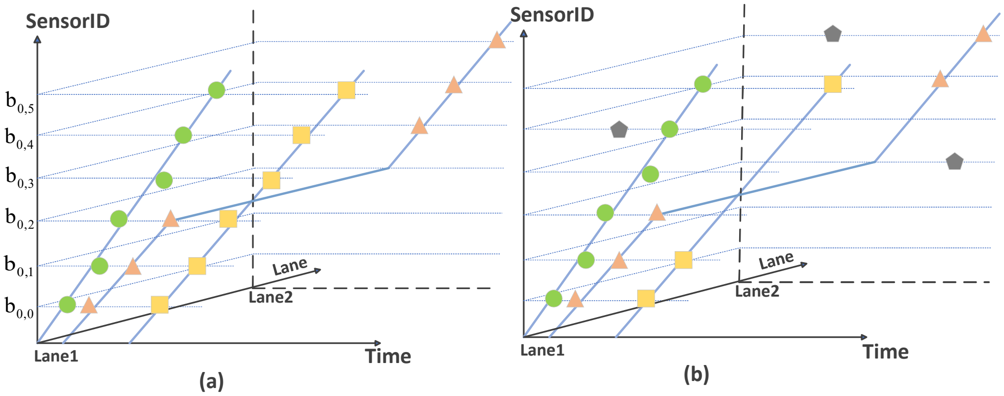


Figure 2 Missed and false detections in a multi-geomagnetic sensor system. Different marker shapes denote sample from geomagnetic sensors for various vehicles. (a) Ideal detections. (b) Detection errors, including single misses (orange triangles), consecutive misses (yellow squares), and false detections (black pentagons).

An example of missed and false detections in the multi-geomagnetic sensor vehicle tracking system is shown in Figure 2. To illustrate this case, we selected three dimensions from the original four-dimensional samples: *time*, *sensorID*, and *lane*. Panel (a) of Figure 2 presents an ideal detection scenario, where the trajectory marked by orange triangles indicates a vehicle changing from lane 1 to lane 2, while green circles and yellow squares represent vehicles traveling straight in lane 1. A problematic scenario involving both missed and false detections is illustrated in panel (b) of Figure 2. Missed detections, including single misses (e.g., the trajectory represented by orange triangles) and consecutive misses (e.g., the trajectory represented by yellow squares), lead to incomplete trajectory data. These missed detection cause large temporal and spatial gaps in the measurement sequence, along with the loss of key motion information, making it difficult for the algorithm to correctly associate measurements belonging to the same vehicle and increasing the risk of trajectory mismatches or omissions. At the same time, false detections (e.g., black pentagons) may occur due to interference from adjacent lanes or environmental noise, further complicating the matching between geomagnetic sensor measurements and actual vehicle trajectories. This issue becomes more pronounced in dense traffic scenarios, where such false detection are more likely to generate spurious trajectories that do not correspond to any real vehicle.

These challenges clearly show the need for a more robust tracking framework. Therefore, the objective of this paper is to develop a system capable of accurate vehicle tracking even in the presence of missed and false detections.

THE PROPOSED ALGORITHM

Trajectory Clustering Algorithm Integrating Directionality and Energy Management

Clustering-based approaches are widely adopted in data stream analysis due to their ability to efficiently discover arbitrary-shaped patterns and adapt to dynamic input without requiring predefined labels (*14*). In the context of vehicle tracking using geomagnetic sensor data, clustering is particularly suitable because it naturally supports online grouping of measurements into vehicle trajectories, even under streaming and noisy conditions. Typical density-based clustering methods assign new samples to existing clusters based on spatial proximity, usually measured by the distance to the cluster center, and use local density to distinguish between meaningful groups (*true targets*) and *noise*. However, these approaches often rely solely on geometric features and may fail when measurements are sparse or disrupted by missed and false detections, which are common in real-world traffic.

To improve clustering robustness under such conditions, we incorporate directionality to capture movement trends, apply dynamic energy management to handle cluster lifecycle, and design a scoring-based data association mechanism that balances spatial distance and directional similarity. This combination enables our method to reduce fragmentation and avoid mismatches in trajectory tracking based on geomagnetic sensor streams.

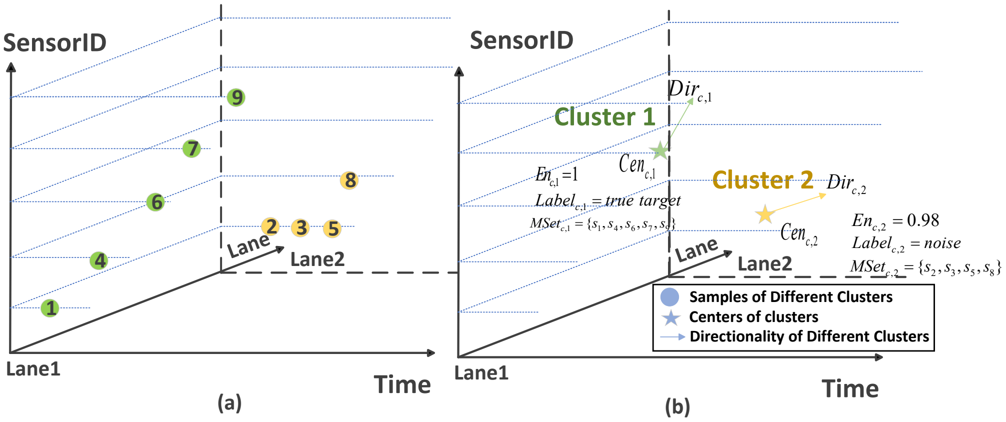


Figure 3 The relationship between samples, clusters, and their cluster graph. (a) Clustered samples, where numbers indicate the temporal order of samples and colors represent cluster assignments. (b) Cluster graph structure.

We use a structure called the cluster graph to store and manage the key attributes of the i-th cluster C_i, as illustrated in Figure 3, defined as:

$$\text{Graph}_i = \{Cen_{c,i},\ \text{Dir}_{c,i},\ \text{En}_{c,i},\ \text{Label}_{c,i},\ \text{Mset}_{c,i}\} \tag{2}$$

each cluster graph contains the state information of a vehicle trajectory, $Cen_{c,i}$ denotes the center of i-th cluster C_i, which physically represents an approximate position of the target associated with this cluster in the four-dimensional sample space. Rather than being a strict geometric center. It is computed as a weighted combination of the cluster's historical position and the newly associated sample, as defined in Equation 7.

$Dir_{c,i}$ is the directionality vector, capturing the recent movement trend of the target within the same feature space as the sample vectors defined in Equation 1. It is computed as a weighted combination of the cluster's historical directionality and the newly associated sample, as defined in Equation 4.

$En_{c,i}$, representing the lifetime of the i-th cluster, is constrained within the range $(-\infty,\ 1]$. We adopt a simple time-based decay model, in which the energy is updated upon each state update, as described in Equation 11. A cluster is considered active if $En_{c,i} > 0$, meaning the target is still being tracked; otherwise, it is treated as inactive and removed from the tracking process.

$Label_{c,i}$ is the category label of the i-th cluster, taking a value of either *true target* or *noise*.

$Mset_{c,i}$ is an associated measurement set containing all samples assigned to the i-th cluster. The initialization procedures for all these attributes are described in Section 3.1.2, while their update mechanisms are detailed in Section 3.1.4.

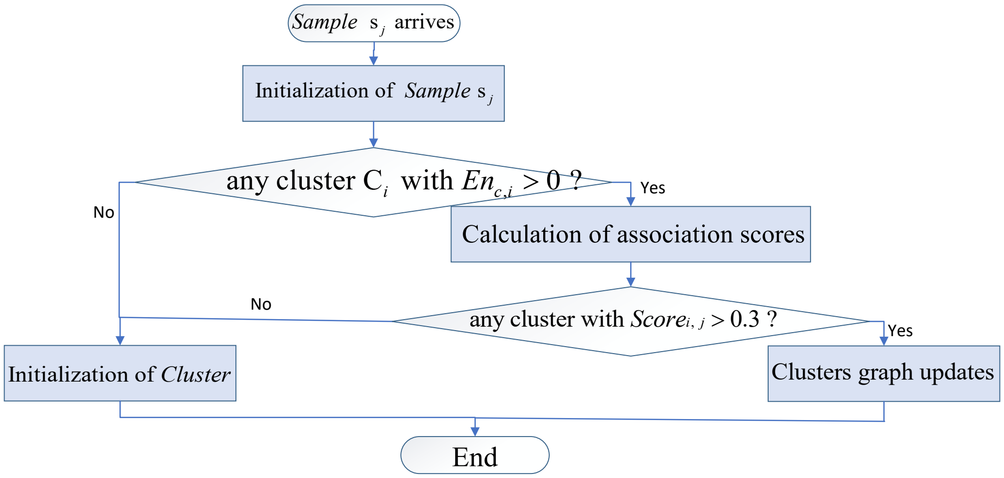


Figure 4 Workflow of the trajectory clustering algorithm.

The overall workflow of the algorithm is illustrated in Figure 4. When a new sample s_j arrives at time $time_j$, the sample is initialized, as described in Section 3.1.1. Next, the algorithm checks whether any existing cluster with a positive lifetime value, i.e., $En_{c,i} > 0$, is available. If no such cluster exists, the sample is used to form a new cluster (see Section 3.1.2), and the current processing round terminates, awaiting the next incoming sample. If at least one such active cluster exists, the algorithm proceeds to compute association scores (detailed in Section 3.1.3). Based on the score results, the sample either initializes a new cluster (Section 3.1.2) or is associated with an existing cluster, with corresponding parameter updates (Section 3.1.4). This process repeats as each new sample arrives. The detailed description of each stage is provided in the following subsections.

Initialization of Sample

To initialize the j-th sample s_j, we assign its center $Cen_{s,j}$ as the complete four-dimensional feature vector $[time_j,\ sensorID_j,\ Mpeak_j,\ lane_j]$, representing the sample's coordinates in the observation space.

Initialization of Cluster

When the j-th sample s_j needs to be initialized as a new cluster C_i, we assign it a corresponding cluster graph structure at time t_j, as defined in Equation 2. Specifically, the cluster center $Cen_{c,i}$ is initialized using the spatial position of the incoming sample, such that $Cen_{c,i} = Cen_{s,j}$. The directionality vector is set as $Dir_{c,i} = Cen_{c,i} + [600,\ 1,\ 0,\ 0]$, which reflects an empirical assumption based on typical highway traffic. It corresponds to a vehicle advancing by one sensor every 600 milliseconds (corresponding to a speed of approximately 90 km/h), without deviating from its lane or changing its magnetic value. The initialization provides a directional seed to support early-stage association, it can be adjusted for different road conditions, and its impact is quickly reduced by continuous updates with newly associated measurements. The cluster lifetime is initialized as $Enc_{c,i} = 1$ to reflect that the target has just been matched with a new sample, indicating recent and reliable activity. The label $Label_{c,i}$ is initially set to *noise* due to the limited number of samples, as defined in Equation 10. Finally, the measurement set is initialized as $Mset_{c,i} = s_j$.

Calculation of Association Scores

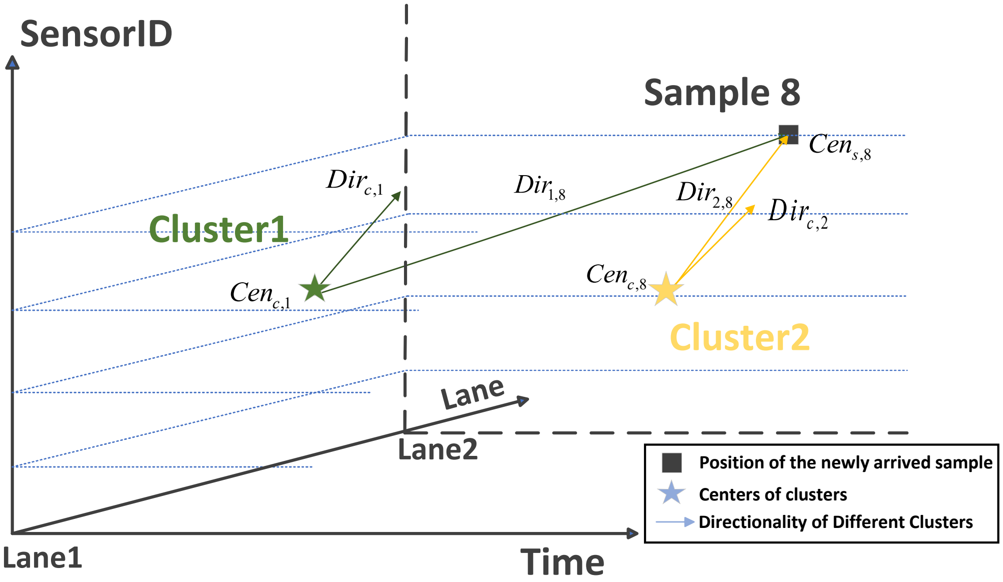


Figure 5 Computation of association scores between clusters and sample.

We use an association score $Score_{i,j}$ to evaluate the benefit of assigning the newly arrived s_j to each currently active cluster C_i $(Enc_{c,i} > 0)$ at time t_j, this score is calculated as defined in Equation 3, with the goal of maximizing the overall assignment effectiveness. Subsequent computations and variable values in this section are based on the state at the arrival time t_j of s_j.

$$Score_{i,j} = \alpha \cdot Dist_{i,j} + \beta \cdot CosSim_{i,j} \tag{3}$$

where α and β are weights that balance the influence of $Dist_{i,j}$ and $CosSim_{i,j}$. $Dist_{i,j}$ is the weighted Euclidean distance between the center of the cluster $Cen_{c,i}$ and the sample's position $Cen_{s,j}$ across four-dimensional space:

$$Dist_{i,j} = \sqrt{\sum_{k=1}^{4} w_k \left(Cen_{c,i}^{(k)} - Cen_{s,j}^{(k)} \right)^2} \tag{4}$$

where w_k represents the importance weight assigned to the k-th dimension of the feature space. The weights w_1, w_2, w_3, w_4, α and β are determined through optimization in Section 3.2. The directional similarity $CosSim_{i,j}$ between the cluster center $Cen_{c,i}$ and the sample's position $Cen_{s,j}$ is illustrated in Figure 5, is expressed as

$$\mathrm{CosSim}_{i,j} = \frac{\mathrm{Dir}_{c,i} \cdot \mathrm{Dir}_{i,j}}{\|\mathrm{Dir}_{c,i}\| \, \|\mathrm{Dir}_{i,j}\|} \tag{5}$$

where $Dir_{c,i}$ represents the directionality of C_i, and $Dir_{i,j}$ denotes the direction vector between the i-th cluster C_i and the j-th sample s_j, defined as

$$Dir_{i,j} = Cen_{s,j} - Dir_{c,i} \tag{6}$$

The sample is associated with the cluster that obtains the highest score, provided that the score exceeds 0.3. This threshold is a commonly used heuristic designed to filter out low-similarity clusters and avoid unreliable associations. We find that this value maintains matching accuracy while providing sufficient tolerance, making it suitable for most real-world scenarios with noise and uncertainty.

Cluster Graph Update

When sample s_j is associated with i-th cluster C_i, the cluster's state at time t_j is updated as follows.

The center $Cen_{c,i}(t_j)$ is updated by integrating the center position $Cen_{c,i}(t_{j-1})$ and the sample's position of the newly associated sample $Cen_{s,j}(t_j)$:

$$Cen_{c,i}(t_j) = \gamma \cdot Cen_{c,i}(t_{j-1}) + \delta \cdot Cen_{s,j}(t_j) \tag{7}$$

where $\gamma + \delta = 1$, and t_{j-1} denotes the arrival time of the previous $(j-1)$-th sample s_{j-1}. By bringing the cluster center $Cen_{c,i}(t_j)$ closer to the actual position of the target, the accuracy of association score calculations between newly arrived samples and all active clusters can be improved.

Similarly, the cluster's directionality vector $Dir_{c,i}(t_j)$ is updated as follows:

$$Dir(t_j) = \varepsilon \cdot Dir_{c,i}(t_{j-1}) + \theta \cdot Dir_{i,j}(t_j) \tag{8}$$

where $\varepsilon + \theta = 1$. The values of γ , δ , ε and θ are optimized in Section 3.2.

The energy $En_{c,i}(t_j)$ is reset to 1 to show that the target was recently detected again.

The new sample s_j is added to the cluster's measurement set:

$$Mset_{c,i}(t_j) = Mset_{c,i}(t_{j-1}) \cup \{s_j\} \tag{9}$$

Lastly, the cluster's category label $Label_{c,i}$ is evaluated based on the size of its associated measurement set $Mset_{c,i}$. If the number of associated samples exceeds a predefined threshold θ_N, the label is updated to *true target*; otherwise, it remains as *noise*:

$$Label_{c,i} = \begin{cases} \textit{true target} & \text{if } |Mset_{c,i}(t_j)| \geq \theta_N \\ \textit{noise} & \text{otherwise} \end{cases} \tag{10}$$

in our implementation, $\theta_N = 5$, which reflects a minimum number of observations required to reliably distinguish *true target* from *noise*.

For all clusters $C_l, (l \neq i)$ not associated with the current sample, their lifetime $En_{c,l}(t_j)$ are reduced by a fixed decay unit decay in Equation 11, while all other attributes of these clusters remain unchanged.

$$\mathrm{En}_{c,l}(t_j) = \mathrm{En}_{c,l}(t_{j-1}) - decay \tag{11}$$

The parameter $decay \in (0,1)$ controls how quickly a cluster is removed when no new samples are associated. We set $decay = 0.02$, meaning a cluster will expire after about 3.5 seconds (around six sensor gaps) without updates. This value was tested on real highway data and worked well across different speeds. The simple linear rule is easy to implement, and its conservative setting allows for short gaps caused by occasional missed detections without losing valid clusters.

Hyperparameter Optimization Framework Based on Bayesian Optimization

The performance of the aforementioned trajectory clustering algorithm depends heavily on a set of key hyperparameters. Traditional optimization methods like grid search and random search explore the hyperparameter space by sampling candidate hyperparameter without leveraging prior evaluations, resulting in poor sample efficiency (*15*). In these approaches, each sampled hyperparameter vector is evaluated by inputting it into the objective function, which returns a scalar performance metric. Black-box optimizers (*16*) are also typically evaluation-intensive and not designed to minimize them effectively. In contrast, Bayesian optimization enables efficient global optimization of black-box functions by coupling a surrogate model with an adaptive acquisition strategy, which guides the sampling process and thereby reduces the number of required evaluations.

For our real-time clustering algorithm, the objective function is defined as $FMI = F(x)$, where $x = \{\alpha, \beta, w_1, w_2, w_3, w_4, \gamma, \delta, \varepsilon, \theta\}$ (see Section 3.1), and Fowlkes-Mallows Index (FMI) is the clustering performance metric (see Section 4.2). We choose FMI as the optimization objective because it effectively balances precision and recall, providing a robust measure of the similarity between the predicted clustering and the ground truth. Its sensitivity to both correct and incorrect cluster assignments makes it particularly well-suited for evaluating the consistency and reliability of clustering structures in real-world vehicle trajectory data. To achieve accurate and robust vehicle clustering with minimal evaluations, we implement a Bayesian optimization framework that iteratively builds a probabilistic surrogate model and uses an acquisition function to guide the selection of the next most promising hyperparameter setting.

Objective Modeling with Gaussian Process

We adopt a Gaussian Process (GP) as the surrogate model to approximate the true objective function $F(x)$. GP is a non-parametric Bayesian model that provides both mean estimates and uncertainty measures, which makes it particularly suitable for sample-efficient exploration in black-box optimization settings (*17*). It is defined by mean function $m(x)$ and kernel function $k(x_i, x_j)$. Assuming zero mean $m(x) = 0$, we use the Squared Exponential kernel (*17*):

$$k(x_i, x_j) = \exp\left(-\frac{1}{2}\|x_i - x_j\|^2\right) \tag{12}$$

In this function, x_i and x_j represent the i-th and j-th hyperparameter vector. This kernel encodes the assumption that the objective function is smooth with respect to its hyperparameters, so small variations in input lead to gradual changes in FMI.

We begin by evaluating the objective at m initial hyperparameter vectors, forming the hyperparameter training set $D_{1:m} = \{(x_n, f_n)\}_{n=1}^{m}$, where each x_n denotes a specific hyperparameter vector, and $f_n = F(x_n)$ represents the corresponding objective value, i.e., the clustering performance (*FMI*) evaluated at the n-th hyperparameter vector. To model the objective function using a Gaussian Process, we define the prior distribution over the function values as $f_{1:m} \sim \mathcal{N}(0, \mathbf{K})$, where $\mathbf{K}$ is the kernel matrix:

$$\mathbf{K} = \left[k(x_i, x_j)\right]_{i,j=1}^{m} \tag{13}$$

For a new candidate hyperparameter vector x_{m+1}, the posterior distribution yields (*18*):

$$F(x_{m+1}) \mid f_{1:m} \sim \mathcal{N}\left(\mu(x_{m+1}), \sigma^2(x_{m+1})\right) \tag{14}$$

with

$$\mu(x_{m+1}) = \mathbf{k}\mathbf{K}^{-1} f_{1:m}^{\top}, \tag{15}$$

$$\sigma^2(x_{m+1}) = k(x_{m+1}, x_{m+1}) - \mathbf{k}\mathbf{K}^{-1}\mathbf{k}^{\top} \tag{16}$$

where $\mathbf{k} = [k(x_{m+1}, x_1), ..., k(x_{m+1}, x_m)]^{\top}$. The posterior mean $\mu(x_{m+1})$ and variance $\sigma^2(x_{m+1})$ are used to construct the acquisition function, which then determines the next candidate point to evaluate.

Efficient Point Selection via Expected Improvement

We use the Expected Improvement (EI) criterion to select the next candidate point. For a candidate hyperparameter x_{m+1}, the EI is defined as $EI(x_{m+1}) = \mathbb{E}[\max(f(x_{m+1}) - f^*, 0)]$, where $f^* = \max(f_{1:m})$ is the best FMI value observed so far. This expectation captures the potential gain of evaluating $F(x_{m+1})$ over the current best value f^*, considering only cases where $F(x_{m+1}) > f^*$. The EI has a closed-form expression (Equation 17) (*18*), which is maximized (Equation 19) to select the next candidate.

$$EI(x_{m+1}) = (\mu(x_{m+1}) - f^*)\,\Phi(Z) + \sigma(x_{m+1})\phi(Z) \tag{17}$$

$$Z = \frac{\mu(x_{m+1}) - f^*}{\sigma(x_{m+1})} \tag{18}$$

here $\Phi(\cdot)$ and $\phi(\cdot)$ denote the cumulative distribution function and probability density function of the standard normal distribution, respectively. The term Z represents the standardization of the improvement relative to predicted uncertainty, enabling us to quantify the significance of the potential improvement in the context of the predicted uncertainty.

The next hyperparameter is selected to maximize the expected improvement by solving

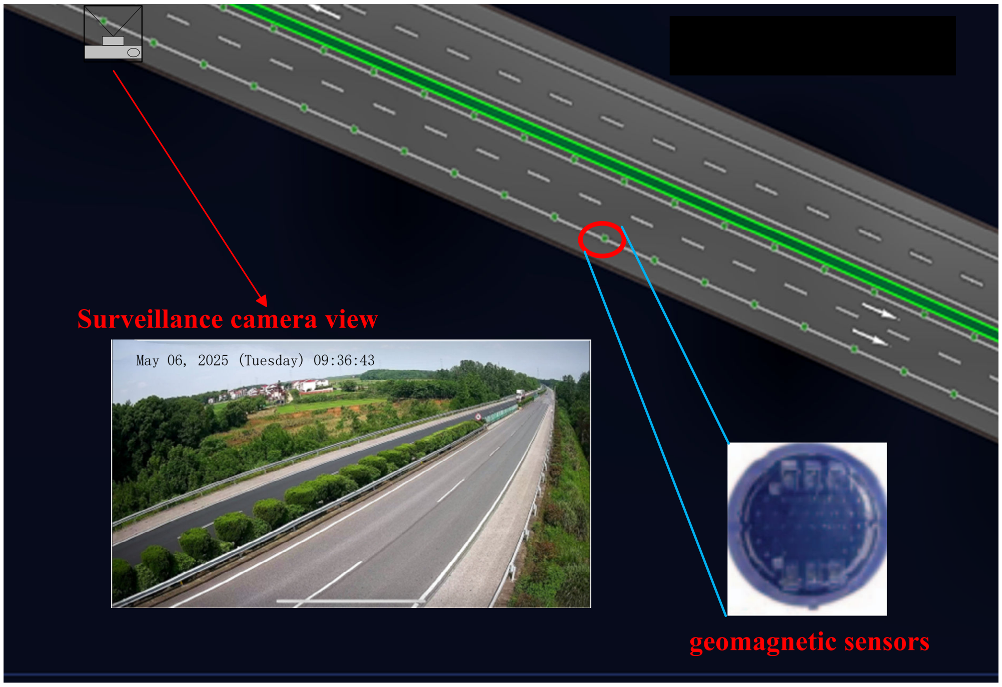
$$x_{m+1} = arg\ max\ EI(x_{m+1}) \tag{19}$$

The newly observed (x_{m+1}, f_{m+1}) is added to the hyperparameter training set $D_{1:m}$, and the GP model is updated by repeating the steps described in Section 3.2.1 to predict the mean (Equation 15) and variance (Equation 16) of the next candidate hyperparameter. These predicted values are then used in the EI (Equation 17) to compute the expected gain, and the next hyperparameter is determined by maximizing EI according to Equation 19. This process continues until the desired number of iterations is reached. Finally, the optimal hyperparameter vector x^* is identified as the one yielding the highest observed FMI.

EXPERIMENTS AND ANALYSIS

Experimental Setup

The proposed algorithm has been deployed on the Dongxiangxi Expressway in Jiangxi Province, China, as illustrated in Figure 6. The one-way, dual-lane expressway, spanning approximately 1080 meters, is equipped with 144 geomagnetic sensors to detect the real-time measurements of vehicles. Additionally, a surveillance camera records actual vehicle movements for qualitative validation of clustering results.



[redacted]

Figure 6 Monitoring View of Dongxiangxi Road Section.

A key challenge addressed in this study is handling non-ideal data. Table 1 shows several representative 10-minute sessions, where sensor non-reporting rates reached up to 50.7% on some days, with up to 5 sensors consecutively missing data. While only a few sessions are shown here, similar reporting conditions have been frequently observed on this road segment and remained relatively similar across different observations.

TABLE 1 Data Reporting Status

Date of Observation	**2024.12.15**	**2024.12.16**	**2025.2.27**
Non-Reporting Rate	50.7%	31.9%	33.3%
Longest Consecutive Non-Reporting Count	5	5	4

Evaluation Metrics

To comprehensively evaluate the performance of clustering, we adopt three classical metrics: the Rand Index (RI), Jaccard Coefficient (JC), and FMI (*19*). These evaluation methods, based on ground-truth labels, assess the consistency between the clustering results and true class assignments by comparing pairwise relationships among samples, so as to reflect how well the algorithm captures the true grouping patterns in the data.

Consider a measurement set $S = \{s_1, s_2, \ldots, s_m\}$ with m samples, where the reference clustering is $C = \{C_1, C_2, \ldots, C_k\}$, and the predicted clustering is $C^* = \{C_1^*, C_2^*, \ldots, C_s^*\}$. We categorize all sample pairs (s_i, s_j) where $i < j$ into four groups: a, the number of pairs correctly assigned to the same cluster in both C and C^*; b, pairs in the same reference cluster but separated in the prediction; c, pairs in different reference clusters but grouped in the prediction; and d, pairs correctly assigned to different clusters in both.

These values satisfy:

$$a + b + c + d = \frac{m(m-1)}{2} \tag{20}$$

The RI measures the proportion of pairs consistently classified in the clustering results.

$$RI = \frac{2(a+d)}{m(m-1)} \tag{21}$$

The JC calculates the proportion of correctly clustered pairs among pairs grouped in either clustering.

$$JC = \frac{a}{a+b+c} \tag{22}$$

The FMI finds the geometric mean of precision and recall for the clustering.

$$FMI = \sqrt{\frac{a}{a+b} \times \frac{a}{a+c}} \tag{23}$$

All metrics range from 0 to 1, with higher values indicating better clustering performance. While RI evaluates overall consistency, JC emphasizes cluster compactness, and FMI balances precision and recall, making it a key performance indicator for this task.

Bayesian Optimization Experiments

We randomly selected 20 minutes of real-world measurements, and their ground-truth cluster labels were established via manual video verification to create a training set. The optimization objective was to maximize the FMI score (equivalent to minimizing negative FMI). Each optimization run consisted of 100 iterations and was repeated 10 times to account for randomness, with the best-performing parameter set being selected.

In Figure 7, the blue curve represents the best observed FMI score after each evaluation (i.e., the maximum observed value of the objective function), while the green curve shows the surrogate model's prediction of the maximum objective value. The optimization process unfolds in two phases. Initially, during the first 40 iterations, both curves decline sharply, reflecting Bayesian

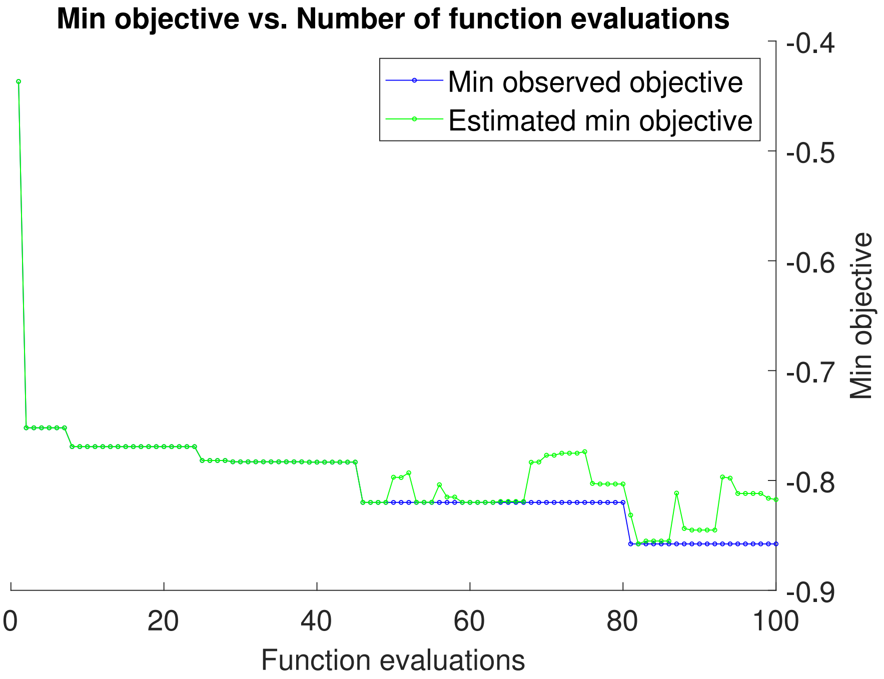


Figure 7 Bayesian Optimization iterative process.

optimization's effective identify promising parameter regions, as the GP surrogate model adapts to the objective function's shape. Subsequently, in the latter 60 iterations, the observed values stabilize around 0.8, while the surrogate model's estimates oscillate between 0.8 and 0.9, suggesting limited potential for further gains with exploration. Convergence is inferred when neither curve shows significant improvement, with the best FMI value reaching 0.8871.

Experimental Evaluation

To evaluate the effectiveness of the proposed clustering algorithm, we compared it with two representative baselines: a model-based tracker and a density-based clustering method. The first baseline is the Kalman Filter (KF) tracker, implemented with a constant velocity model and using the Hungarian algorithm for data association (*20*). This represents a standard approach for vehicle tracking on highways. The second baseline is DBSCAN (*21*), a well-known density-based clustering algorithm that handles noise but lacks any motion model, making it a suitable contrast for our method.

To ensure fairness, we optimized the key parameters of DBSCAN, including the neighborhood radius eps and the minimum number of core points $min\_samples$, as well as the full set of hyperparameters for our method using the same Bayesian Optimization framework on a common training set. The Kalman Filter adopts a constant velocity (CV) motion model. The state vector follows the four-dimensional structure defined in Equation 1, including time, geomagnetic sensor ID, magnetic peak value, and lane index. To ensure a fair comparison under imperfect data conditions involving missed and false detections, the process noise covariance matrix Q and the measurement noise covariance matrix R are set as diagonal matrices with moderately conservative values:

$$Q = \text{diag}(1 \times 10^{-3},\ 1,\ 0.05,\ 0.2) \tag{24}$$

$$R = \text{diag}(5 \times 10^{-3},\ 2,\ 0.1,\ 0.5) \tag{25}$$

These settings are designed to account for uncertainties commonly observed in geomagnetic sensor data.

The evaluation was conducted on a 25-minute real-world measurement with 256 vehicles. Our method and KF were executed in real time, processing data sequentially. The collected measurement set was then used offline for DBSCAN clustering.

TABLE 2 Clustering Performance Comparison Across Different Methods.

Method	**JC**	**FMI**	**RI**
Kalman Filter	0.7150	0.8121	0.9550
DBSCAN	0.6123	0.7245	0.9124
Ours	0.7814	0.8683	0.9871

Table 2 shows that our method outperforms both baselines across all metrics, achieving an FMI of 0.8683 compared to 0.7245 for DBSCAN. These results highlight the advantage of integrating directionality and Bayesian tuning in real-world conditions.

As shown in Figure 8, the proposed method generates clear trajectory clusters, where each color represents a unique vehicle. Clusters identified as *noise* by the algorithm are excluded from the visualization. For visualization purposes, we normalized time, sensor position, and lane index to present trajectories clearly in a unified coordinate space and avoid distortion from uneven value scales. The Figure 8 demonstrates that the method maintains robust tracking performance even in the presence of missing data. Notably, even when significant gaps exist within a trajectory, the algorithm can still correctly associate the fragmented segments, highlighting its strong resilience to incomplete observations.

CONCLUSION AND FUTURE WORK

In this work, we proposed a robust real-time vehicle tracking method using geomagnetic sensors. By incorporating directionality constraints and dynamic energy management into a clustering framework, the method improves trajectory accuracy and stability. A Bayesian optimization framework was employed to adaptively tune hyperparameters under varying traffic conditions. The algorithm was deployed on the Dongxiangxi Expressway in Jiangxi, China, and achieved superior performance compared to DBSCAN and Kalman Filter, with an FMI score of 0.8683. Future work will focus on extending the approach to multi-lane and congested traffic scenarios.

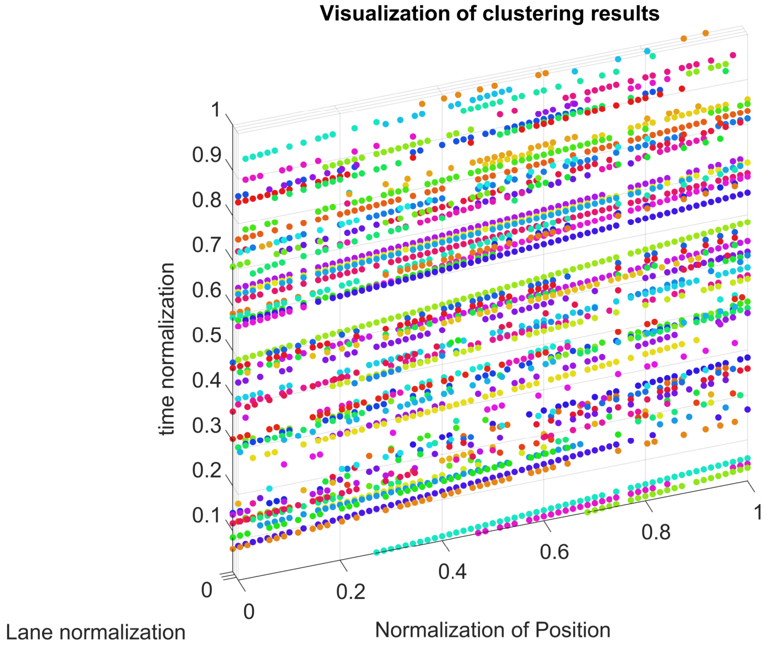


Figure 8 Cluster result chart.

REFERENCES

1. Azimjonov, J. and A. Özmen, A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Advanced Engineering Informatics*, Vol. 50, 2021, p. 101393.
2. Bai, Z., G. Wu, X. Qi, Y. Liu, K. Oguchi, and M. J. Barth, Infrastructure-Based Object Detection and Tracking for Cooperative Driving Automation: A Survey. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 1366–1373.
3. Bai, J., S. Li, H. Zhang, L. Huang, and P. Wang, Robust Target Detection and Tracking Algorithm Based on Roadside Radar and Camera. *Sensors*, Vol. 21, No. 4, 2021.
4. Velazquez-Pupo, R., A. Sierra-Romero, D. Torres-Roman, Y. V. Shkvarko, J. Santiago-Paz, D. Gómez-Gutiérrez, D. Robles-Valdez, F. Hermosillo-Reynoso, and M. Romero-Delgado, Vehicle Detection with Occlusion Handling, Tracking, and OC-SVM Classification: A High Performance Vision-Based System. *Sensors*, Vol. 18, No. 2, 2018.
5. Pan, Y., F. Huo, Z. Wang, S. Zhai, and Z. Geng, A Millimeter-Wave Radar Tunnel Obstacle Detection Method Based on Invalid Target Filtering. *Applied Sciences*, Vol. 13, No. 11, 2023.
6. Čulík, K., V. Štefancová, and K. Hrudkay, Application of Wireless Magnetic Sensors in the Urban Environment and Their Accuracy Verification. *Sensors*, Vol. 23, No. 12, 2023.
7. Wang, H., W. Quan, and W. Y. Ochieng, Smart road stud based two-lane traffic surveillance. *Journal of Intelligent Transportation Systems*, Vol. 24, No. 5, 2020, pp. 480–493.
8. Wang, J., J. Gao, S. Zhao, R. Zhu, Z. Jiang, Z. Chu, Z. Mao, and Y. Shen, From Model to Algorithms: Distributed Magnetic Sensor System for Vehicle Tracking. *IEEE Transactions on Industrial Informatics*, Vol. 19, No. 3, 2023, pp. 2963–2972.
9. Mao, G., T. Fu, and X. Ren, Asynchronous Data Fusion for Vehicle Tracking Using MMW

Radar and Magnetic Sensor in Tunnel. In *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, 2025, pp. 1–7.

10. Wang, K., H. Xiong, J. Zhang, H. Chen, D. Dou, and C.-Z. Xu, SenseMag: Enabling Low-Cost Traffic Monitoring Using Noninvasive Magnetic Sensing. *IEEE Internet of Things Journal*, Vol. 8, No. 22, 2021, pp. 16666–16679.
11. Zhang, Z., X. He, and H. Yuan, An Anti-Interference Traffic Speed Estimation System with Wireless Magnetic Sensor Network. *IEEE Transactions on Industrial Electronics*, Vol. 67, No. 11, 2019, pp. 9743–9752.
12. Wang, Q., J. Zheng, H. Xu, B. Xu, and Y. Zhu, Roadside magnetic sensor system for vehicle detection in urban environments. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 18, No. 8, 2017, pp. 2217–2226.
13. Yin, W., Y. Wang, H. Li, and Z. Wang, Distributed Magnetic Sensor-Based Vehicle Detection with LoRa Communication for Smart City Traffic Monitoring. *IEEE Internet of Things Journal*, Vol. 9, No. 13, 2022, pp. 11084–11094.
14. Liu, P. and M.-H. Meng, Online data-driven fuzzy clustering with applications to real-time robotic tracking. *IEEE Transactions on Fuzzy Systems*, Vol. 12, No. 4, 2004, pp. 516–523.
15. Bergstra, J. and Y. Bengio, Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, Vol. 13, No. null, 2012, p. 281–305.
16. Powell, M. J. D., A view of algorithms for optimization without derivatives. *Mathematics Today-Bulletin of the Institute of Mathematics and its Applications*, Vol. 43, No. 5, 2007, pp. 170–174.
17. Seeger, M., GAUSSIAN PROCESSES FOR MACHINE LEARNING. *International Journal of Neural Systems*, Vol. 14, No. 02, 2004, pp. 69–106, pMID: 15112367.
18. Frazier, P. I. and J. Wang, *Bayesian Optimization for Materials Design*, Springer International Publishing, Cham, pp. 45–75, 2016.
19. Brun, M., C. Sima, J. Hua, J. Lowey, B. Carroll, E. Suh, and E. R. Dougherty, Model-based evaluation of clustering validation measures. *Pattern Recognition*, Vol. 40, No. 3, 2007, pp. 807–824.
20. Bewley, A., Z. Ge, L. Ott, F. Ramos, and B. Upcroft, Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3464–3468.
21. Ester, M., H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, 1996, pp. 226–231.