01 Oct 2022

# Efficient Data Collection in IoT Networks using Trajectory Encoded with Geometric Shapes

Xiaofei Cao

Sanjay Kumar Madria
*Missouri University of Science and Technology*, madrias@mst.edu



## Recommended Citation

X. Cao and S. K. Madria, "Efficient Data Collection in IoT Networks using Trajectory Encoded with Geometric Shapes," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 4, pp. 799 - 813, Institute of Electrical and Electronics Engineers, Oct 2022.
The definitive version is available at https://doi.org/10.1109/TSUSC.2020.3044292

# Efficient Data Collection in IoT Networks Using Trajectory Encoded With Geometric Shapes

Xiaofei Cao and Sanjay K Madria, *Senior Member, IEEE*

**Abstract—**The mobile edge computing (MEC) paradigm changes the role of edge devices from data producers and requesters to data consumers and processors. MEC mitigates the bandwidth limitation between the edge server and the remote cloud by directly processing the large amount of data locally generated by the network of the internet of things (IoT) at the edge. An efficient data-gathering scheme is crucial for providing quality of service (QoS) within MEC. To reduce redundant data transmission, this paper proposes a data collection scheme that only gathers the necessary data from IoT devices (like wireless sensors) along a trajectory. Instead of using and transmitting location information (which may leak the location anonymity), a virtual coordinate system called "distance vector of hops to anchors" (DV-Hop) is used. The proposed trajectory encoding algorithm uses ellipse and hyperbola constraints to encode the position of interest (POI) and the trajectory route to the POI. Sensors make routing decisions only based on the geometric constraints and the DV-Hop information, both of which are stored in their memory. Also, the proposed scheme can work in heterogeneous networks (with different radio ranges) where each sensor can calculate the average one-hop distance within the POI dynamically. The proposed DV-Hop updating algorithm enables the users to collect data in an IoT network with mobile nodes. The experiments show that in heterogeneous IoT networks, the proposed data collection scheme outperforms two other state-of-the-art topology-based routing protocols, called ring routing, and nested ring. The results also show that the proposed scheme has better latency, reliability, coverage, energy usage, and provide location privacy compared to state-of-the-art schemes.

**Index Terms—**Trajectory, routing, DV-Hop, GPS-free, hyperbola, ellipse, IoT, edge computing, sensor, encoding

## 1 Introduction

The Internet of Things (IoT) facilitates fast access, process, and utilization of the big data created by the 'things' surrounding many applications such as disaster management, battlefield monitoring, and moving object tracking. However, for some IoT devices like wireless sensors, the limited energy, and high recharging cost require them to save energy as much as possible during their duty cycles. Bandwidth limitation is another challenge for IoT networks. In recent years, the number of IoT devices, the data rate, and the enormous data produced by these large numbers of things are increasing faster than the growth of wireless bandwidth. Also, for heterogeneous IoT networks with mobile nodes, there is a need of maintaining the correct and stable data collection routes without leaking the users' locations information. Last but not the least, different applications should be able to share the same IoT network efficiently. Therefore, the users of different applications can fetch data from sensors regardless of their types, precise locations, and identities.

In recent years, researchers have turned their focus on edge computing [1] and fog computing [2] to support IoT networks. The edge/fog networks interacting with local wireless sensor networks (WSNs) provide services with higher reliability in collecting, caching, and exploiting sensing data locally. Also, the latency of data collection in mobile edge networks (MEN) is reduced because edge nodes can collect and process data faster than the remote cloud. However, with the participation of third party mobile edge devices, the location anonymity problem draws some researchers' attention again. For example, consider a celebrity athlete who may try to get pollution levels along his/her running trajectory from crowd-sourcing and/or existing environmental sensor networks. An adversary could easily infer the user's locations like the hiking trail and home address, and predict the mobility based on the history of using the location-aware services [3]. Therefore, as such this individual user would like to keep the location anonymity against the potential security risks.

In MEN, sensors and other IoT devices contribute to the most volume of sensing data. In most of the WSN applications, sensors report their sensing data periodically. However, periodically sensing has a long delay which is not desirable for applications requiring real-time low-distorted data. In recent works like [4], data sensing only happens when the sensor or mobile user receives a data request packet. Therefore, unnecessary data sensing and reporting outside of the position of interest are reduced. Another way to reduce the data delay is broadcasting, which is widely used as the fastest way to disseminate real-time data to the whole network. The drawback of broadcasting is its high energy consumption due to massive rebroadcasting. Therefore, the counter-based broadcasting scheme [5] and their adaptive versions [6], [7], [8] are proposed to minimize the redundant rebroadcasting to save energy and mitigate the broadcast storm effects [5]. However, even the state-of-the-art adaptions of counter-based broadcasting

• *The authors are with the Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409 USA. E-mail: {xc9pd, madrias}@mst.edu.*

*Manuscript received 21 Apr. 2020; revised 25 Oct. 2020; accepted 27 Nov. 2020. Date of publication 14 Dec. 2020; date of current version 8 Dec. 2022. (Corresponding author: Sanjay K Madria.) Recommended for acceptance by Daniel Grosu, Jiannong Cao, and Marco Brocanelli. Digital Object Identifier no. 10.1109/TSUSC.2020.3044292*

cannot reduce more than 60 percent rebroadcasting. Also, the broadcast can not control the data flow precisely which is not acceptable for some military applications like tracking enemy objects, while not being detected by the targets.

Some trajectory-based routing protocols, which route packets through wireless sensor nodes that reside more or less on the designed trajectory, have the potential to fetch the data from specific areas with the minimum overhead of redundant forwarding. However, most of the trajectory routing protocols like [9] and [10] require all the sensor nodes to have the GPS to decode the encoded routing trajectory, which is not practical for low-cost WSNs. Although the cubic Bezier curve used in [9] provides a good compression ratio for the position of interest (POI), it still cannot be adapted in WSNs without GPS modules. A virtual coordinate system is an option for IoT consisting of WSNs without GPS. It can use local connectivity information such as the number of neighbors of each node and the perimeter nodes' locations as in [11]. It can also use the anchor nodes and the vector of minimum hop distance (DV-Hop) to the anchor nodes to estimate the distance between nodes. However, the state-of-the-art DV-Hop based location estimation [12] requires lots of memory resources and computational power, therefore, is not suitable for low power wireless sensor networks.

To address the shortcomings of the existing works, this paper proposes a spatial data collection scheme that has both low latency and less overhead of redundant broadcasting. Instead of using the exact nodes' location information from GPS as in [9], [10], [13], The proposed algorithm uses a vector of the minimal distance of hops (DV-Hops) to all the anchor nodes selected by the secure fog server as a dictionary or virtual coordinate. The area of the position of interest can be represented as a list of hop constraints to the anchor nodes. The routing message only contains two basic geometric shapes; hyperbola and ellipse segments in the proposed scheme. Each shape is encoded with simple hop constraints (e.g., size of the ellipse and the hyperbola and the start and end of the segment). The sensor nodes could avoid complex geometric computing, which makes it suitable for WSNs that have low-power and low-computing resources. In addition, the proposed scheme provides location anonymity by avoiding using and transmission of the GPS location information. To decode the POI of the client, the adversary has to have the encoded message as well as the location of the anchor nodes, which are stored in the secure fog server. The broadcast storm issue is addressed by integrating the counter-based broadcasting mechanism. The performance evaluation shows that the proposed scheme reduces the redundant rebroadcast in a small real-world WSN. In simulation experiments, it compresses the data about eight times and reduces more than half of the latency in the data requesting and collection process compared to directly broadcasting the list of node identifications that reside within the POI.

The reliability of the proposed scheme also beats the state-of-the-art geospatial routing protocols like ring routing and nested routing [14], [15], which route messages in a circular trajectory. The energy consumption of data requests within our scheme is also reduced compared to the state-of-the-art counter-based broadcasting schemes [7], [8].

The paper is organized as follows: Section 2 discusses related works about routing, broadcasting, and virtual coordinate. Section 3 first describes the system overview and assumptions. Then it elaborates the method of encoding trajectory with hyperbola and ellipse. Last, it shows some examples of encoding and routing with the encoded trajectory message. Section 4 gives an adapted data collection protocol for low-power WSN in where sensors sleep periodically. Section 5 explains the experiment setup and demonstrates the performance improvement with detailed elaborations. Section 6 concludes the paper with future work ideas.

## 2 Related Works

### 2.1 Counter Based Broadcasting

Broadcasting is the fastest way to flood a message to cover the whole WSN. However, limited bandwidth causes a delay in broadcasting a sequence of messages into the network. After a node receives a given packet, the counter-based broadcasting schemes [5], [6], [7], [8] require a node to wait for a short period to listen to its neighbors and count how many times the given packet has been rebroadcast. If the broadcast count of the given packet reaches the predefined threshold, it will drop the packet. Thus, only a few of the nodes in the network will rebroadcast the given packet which saves bandwidth and thus, alleviates the congestion.

### 2.2 Grid-Based Routing

Hierarchical grid-based routing is an energy-efficient method for routing of data packets [16]. With the mobile sink and predefined virtual grid, packets could bypass the congestion area of the grid and route to the mobile sink by fetching the updated mobile sink's location from the cell-center. The grid-based routing protocol can be classified into two categories, the query-based protocol (i.e., PANEL [17], Grid-Based Coordinated Routing GMCAR [18], etc.) and the event-based protocol EAGER [19]. For the query-based routing protocol, sensor nodes only sending the sensing data on request, while event-based protocol sensors will report events based on the pre-configuration of the event definition. Comparing to the event-based routing protocol where each sensor report sensing data periodically, the query-based routing protocol greatly reduces the overhead of unnecessary sensing and routing, which saves both energy and bandwidth. However, efficiently disseminating the data request packets in a WSN without GPS is a challenge for the query-based routing protocol. Some works like [16] switch alternately grid-head states to overcome the energy and bandwidth overhead of flooding control packets. The work [20] uses the location information of the cell-header and their neighbors to forward the query towards the target cell and flood message only in the target cell. Though these works reduce the broadcast overhead, the grid-based routing still needs the GPS information and extra energy to maintain the grid topology.

### 2.3 Ring Routing and Nested Routing

Ring routing and nested routing [14], [15] are proposed to solve the problem of routing packets to a mobile sink. The idea behind ring routing and nested routing is to store the current mobile sink's location in a ring or nested ring structure. The data source needs to query the nodes in the ring/nested ring structure to fetch the updated location of

the mobile sink before routing the packets. Then the data source routes the packets to the mobile sink using the updated sink location. Ring routing and nested routing achieve good delay and energy performance because searching the ring structure is easier than searching the whole network.

### 2.4 Trajectory Based Routing and Virtual Coordinate

Trajectory based routing [9], [10] is a paradigm that only the nodes near the given routing trajectory will forward the packets. It includes trajectory generating and encoding and the routing decision rules for each sensor node. It has the following challenges: First, the trajectory encoding algorithm should able to compress the trajectory as the encoded message will be included in the routing packets. Second, each compressed message should be able to route through the trajectory to the sink reliably. Third, the overhead of routing caused by redundant rebroadcast should be minimized. However, for a WSN, the additional challenge is to route through a trajectory without using any GPS-based location information.

To route through a trajectory with virtual coordinates using DV-Hop rather than GPS, the virtual coordinates should be able to reflect the sensors' real location precisely. Intuitively, increasing the number of anchor nodes will improve the precision of the virtual coordinates. This has also been proven by DV-Hop based localization algorithms such as in [21], [22], and [23]. Some previous works like [24] and [23] use the non-dominated sorting (NSGA-II) algorithm to improve positioning accuracy for complex network topologies with many anchor nodes. Their results demonstrate DV-Hop based localization algorithm could achieve good localization accuracy. The challenge in this work, however, is to reduce computation and memory usage which are limited in sensor nodes. The naive combination of greedily checking the distance to the routing trajectory and the use of the virtual coordinate system with many anchor nodes requires computational resources and is also error-prone due to the use of the estimated location. According to the DV-Hop based localization algorithms, in the worst case, the error rate can be as large as 45 percent of the range of the radio [12], [24] and [23]. which could lead to routing failure.

### 2.5 Location Privacy in WSN

WSNs are vulnerable to be attacked by an eavesdropper. To oppose the eavesdropper and protect the location information of the source, sink, and gateways, different approaches are proposed. To protect the source's location privacy, the paper [25] proposed a dynamic clustering algorithm and dynamic shortest path scheme that change the network topology dynamically. Thus, the adversary won't be able to locate the routing path and can't find the source by backtracking. However, the dynamic clustering algorithm consumes excessive energy which is a drawback for many real-world applications. A more energy-efficient approach [26] proposed a hybrid source location privacy protection scheme that combined phantom source node strategy and ring routing. The phantom source node could mislead the adversary to the wrong location and the ring routing saves energy while has a good routing performance.

To protect the sink nodes' location privacy, the work [27] proposed a routing scheme that preserves the sink's location anonymity and guarantees to route within a delay threshold. The data packets route to the sink's location through several ray routes. Only one ray route is the true that goes through the sink location. The sink collects the data from random nearby intermediate nodes which store the data from the sender. However, the proposed ray routing requires the sensor nodes to maintain a table of their nearby neighbor nodes and their location information.

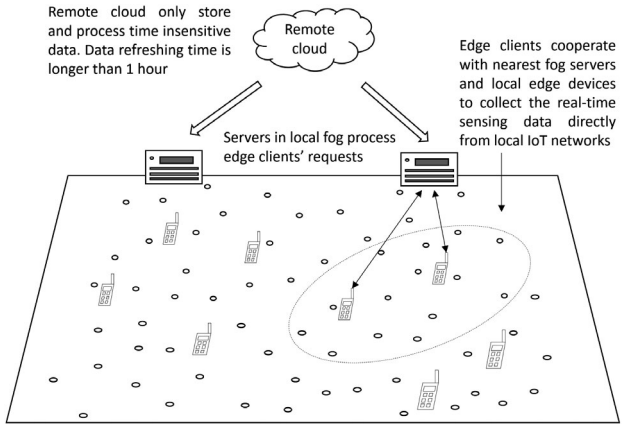



Fig. 1. Data collection in local edge network.

## 3 Proposed Data Collection Scheme

The proposed data collection scheme enables the fog server to directly collect only the necessary data for the edge clients from nearby IoT networks by sending a data request message. The data request packet from mobile edge devices like cellphones that are near the client's position of interest should be able to reach the targeted IoT devices (usually the wireless sensor motes) with minimum overhead and latency. In heterogeneous IoT networks, different wireless devices with different radio standards cannot directly communicate with each other.

### 3.1 System Overview

The proposed data collection scheme is based on an edge-computing paradigm where the edge devices consume data within the geometric constraints specified by the client. It uses mobile edge devices as the gateway that coordinate with the wireless sensor network and downstream fog servers. The data collection tasks are offloaded to the local edge nodes near the POI, and thus reduces the latency. The data processing tasks are offloaded to the local fog server.

The system overview is shown in Fig. 1. The cloud data centers only collect, process, and store the time-insensitive data. The fog servers collect, process, and generate the compressed geospatial constraints of user requests which will be discussed in the next section. Then, the data collection task is offloaded to the mobile edge devices near the POI. With the help of the serial port listening and writing app [28], the edge device becomes a gateway to the IoT sensor network by connecting a sensor mote to its serial port.

### 3.2 Assumptions

In IoT networks, many applications prefer to have the geospatial information tagged with the sensing data. For example, firefighters want to know the chemical leak status

based on the temperature and infrared sensors' data tagged with the location information. Due to their low cost and energy limitation, most of the sensor motes do not contain GPS modules themselves. To enable these sensors to provide geospatial information without GPS, the proposed algorithm uses a vector of the minimal distance of hops to the anchor nodes (DV-Hop) as the virtual coordinates.

The proposed data collection scheme has the following assumptions:

First, the anchor nodes, which are selected by the fog server, know their locations. They have a one-byte long ID number, which is enough for a local fog network. Also, the mobile edge devices, which different from anchor nodes, collect data from a WSN directly. Nodes in the WSN (maybe mobile also) may have different radio ranges and different sensing preferences.

Second, the fog server, which encodes the geospatial data request, has the location information of the local anchor nodes and has enough computing resources. In the WSN, near each sensor node, there are at least three randomly deployed anchor nodes that will flood their identification to the others in the area within a limited number of hops ($H_{max}$) from them. While flooding, all the sensor nodes will create and update a vector of the minimal distance of hop (DV-Hop) to their nearby (within $H_{max}$ hops) anchor nodes.

Third, the anchor nodes also have the DV-Hop of their nearby (within $H_{max}$ hops) anchor nodes and will eventually transmit this information to the fog server. After the network has been initialized, if any nodes move, they need to update their DV-Hop by querying all their new neighbors' DV-Hop. For the new DV-Hop of the moving nodes, the new DV-Hop entry is set to be one plus the minimum hop count entry of all the new neighbors' DV-Hops.

Fourth, this work assumes the user only wants to collect the necessary data where the definition of necessary data can vary based on the applications. For example, in the battlefield monitoring application, we have mentioned in the introduction, unnecessary radio broadcasting not only has the risk of being eavesdropped on but also exposes to the tactical intent. For an environmental monitoring application, we may want to monitor different locations with different frequencies. For this paper, in short, the sensor data reside in the trajectory that the user requests as the necessary data. We haven't restricted how the users should define their personalized trajectory.

Fifth, the trajectory, which includes the routing path and POI, can have the shape of any type of continuous curve. It may have different widths in different segments and can overlap with itself. The trajectory is unidirectional. So the overlapping trajectory is seen as one curve and the intersecting trajectory is seen as a branch. To avoid looping, it is assumed that every node in the trajectory re-broadcast the same routing packet only once. A routing packet size should be fewer than 127 bytes (the limit of IEEE 802.15.4 packet size).

### 3.3 DV-Hop Based Geospatial Encoding Algorithm

The topology of the IoT network is dynamically changing, as the network is heterogeneous and some nodes are mobile. So, the fixed routing table, which proactive routing protocol uses, is not suitable for routing and collecting data in IoT networks. Also, the reactive or cluster-based routing protocols are not efficient due to the overhead of updating the routing table or maintaining the cluster topology. They also need to be tuned for each specific application. Thus, it is hard to mix the use of different applications. The proposed trajectory encoding algorithm uses a similar idea in a software-defined network (SDN), which is to decouple the data plane (network layer) and the control plane of the IoT network. In the proposed work, the routing controllers are the fog servers shown in Fig. 3.1. The routing rules are encapsulated in each routing packet. Here, the routing rules are encoded by the controller application which resides in the control plane. Each sensor in a data plane is also seen as a mini router that decides the routing action (i.e., re-broadcast or drop packets) based on the matching rules in the routing packets.

To route without using the GPS data, the proposed algorithm uses DV-Hop as the virtual coordinate and store the DV-Hop table in each IoT device in the network. The controlling information in the routing packets is the geometric shape constraints discussed in the following paragraphs.

The idea of the proposed encoding algorithm is to use a set of geometric shapes to represent the position of interest and the trajectory from a gateway to the POI. With the assumption as in Section 3.2, the trajectory and POI drawn by the clients can be seen as a set of discrete pixels in a 2d euclidean space. Each pixel has two parameters: the x- and y-coordinates from the predefined origin point. The unit of the coordinate, $\mu$, is chosen based on the application requirement. So each 1 $\mu$ by 1 $\mu$ area in this WSN is a pixel that can be represented by a tuple $(x_{coor}, y_{coor})$ which is called the Trajectory Area Set (TAS).

Intuitively, a line can be used to connect two nodes with the shortest path length and an arc can be used to connect two lines with different directions. Then, any trajectory can be seen as the assembly of these two shapes or their more generalized form: the Hyperbola segment and an arc segment.

For example, in Fig. 2a, the shortest path from node $S_1$ to node $S_2$ is a straight line. Using anchor nodes $A_1$ and $A_2$, a hyperbola can be defined as: $h_1 - h_2 = 0$ which is a line passing through $S_1$ and $S_2$. Then, a third anchor node $A_3$ with hop $h_3$ is required to cover both $S_1$ and $S_2$. Finally, it gives a segment of the line that starts and ends from $S_1$ and $S_2$. Here, the anchor nodes $A_1, A_2$ determine the line's direction, and $A_3$'s location and hop constraints determine the line's starting and ending points. As a straight line is a special case of a hyperbola, a hyperbola can be used to approximate the line. The left arc of a hyperbola shown in Fig. 2b can be represented with constraints $h_1 - h_2 = 2 \times a, (a \in Integer)$ . Then, with the anchor node $A_3$ and its hop count $h_3$, it generates a segment of the hyperbola. When $a = 0$, the segment is a straight line.

Another example in Fig. 3 shows how to use the arc of a circle to connect two lines with different directions [29]. As a circle is a special case of an ellipse with overlapped foci, the ellipse segment could replace the circle segment.

The objective now is to use the virtual coordinate to represent these two shapes: the hyperbola segment and the ellipse arc. As the fog server knows the location and the DV-Hop of each anchor node, the average hop distance $d_{avg}$ in $\mu$ unit between each pair of anchor nodes can be

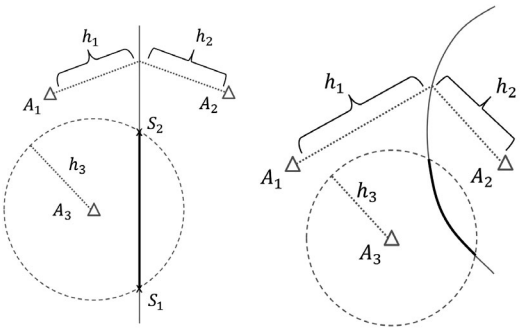



Fig. 2. Example of a segment of line, and a hyperbola trajectory represented with hop constraints.

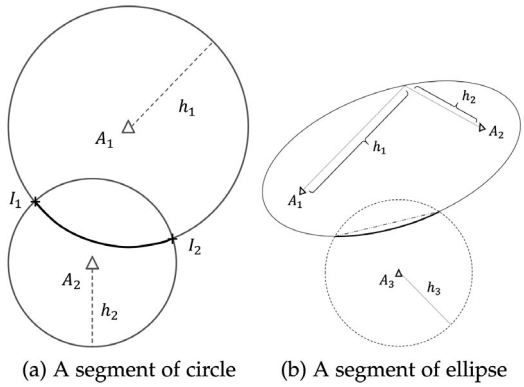



Fig. 3. Example of a segment of circle, and a ellipse segment.

calculated. Each pair of the anchor nodes can be seen as the foci of a set of hyperbolas and a set of ellipses. The desired hyperbola and an ellipse can be chosen by setting the $a$ value (in hop counts) of the Cartesian equation of the hyperbola and an ellipse shown in Equations (1) and (2) where $x_o, y_o$ are the coordinate of the center of the ellipse and the hyperbola. At last, keep the hyperbola and ellipse segments which overlap most of the trajectory for data collection. The segment is the intersection of the hyperbola and an ellipse with a control circle that uses an anchor node as its center and given hops count as its radius which is shown in Equation (3) where $x_c, y_c$ are the coordinate of the center of the circle. Then, the two following simple shapes: the hyperbola segment and the ellipse segment can be encoded with the hop constraints.

$$\frac{(x-x_o)^2}{a^2} - \frac{(y-y_o)^2}{b^2} = 1 \tag{1}$$

$$\frac{(x-x_o)^2}{a^2} + \frac{(y-y_o)^2}{b^2} = 1 \tag{2}$$

$$\frac{(x-x_c)^2}{r^2} + \frac{(y-y_c)^2}{r^2} = 1. \tag{3}$$

The computation complexity is another challenge in encoding a geospatial area using shapes. As discussed, a hyperbola or an ellipse segment is determined with three different anchor nodes (two as foci and one as the center of the control circle) and two constraints of hop-distance (one as the $a$ value, and the other as the radius of the control circle). For a WSN with anchor nodes $N_{Anchor}$ and hops limitation $H_{max}$ for each anchor node, the shapes (hyperbola and ellipse) which constitute the trajectory are chosen from $N_{shapes} = N_{Anchor}^3 H_{max}^2$ different possible shape constraints. Testing of all the combinations of the shapes has $O(N_{shapes}!)$ time complexity, which is not practical. Thus, this paper proposes a greedy algorithm that considers both the number of newly covered pixels and the effective coverage ratio ($ECR$). $ECR$, defined in Equation (4), is the ratio of the overlapping area of TAS and a shape over the mathematical area of the shape.

$$ECR = \frac{Area_{covered}}{Area_{shape}}. \tag{4}$$

The trajectory encoding algorithm is divided into the three steps discussed next.

#### 3.3.1 Find all Possible Shapes From $N_{shapes}$ and Their Area $Area_{shape}$

The possible shapes are defined as the shapes that could constitute a portion of the trajectory. In other words, the possible shapes must overlap with a portion of the data collection trajectory. The objective of this step is to generate $N_{shapes}$ shape constraints where $N_{shapes} = N_{Anchor}^3 H_{max}^2$. Each possible shape constraint can be represented as follows:

$$C_{shape} = [A_1, A_2, A_3, a, h_3]. \tag{5}$$

Here, $A_1, A_2$ are the foci of the shape hyperbola or the ellipse, $A_3$ is the center of the control circle, $a$ is the parameter in the Cartesian Equations (1) and (2), and $h_3$ is the radius of the control circle. The average one-hop distance $d_{avg}$ is used to estimate the overlapping area of each shape as shown in the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TSUSC.2020.3044292. Then the algorithm uses the following filtering criteria to filter each shape constraint (hyperbola and ellipse segments) and discard shapes that don't overlap with the TAS.

First, check if the control circle overlaps with the TAS. As the control circle will intercept the shape segment and finalize the shape, it is the most important criteria to determine if a shape overlaps with the TAS. The proposed algorithm only keeps the constraints that overlap with the TAS by comparing the distance of the center of the control circle and the convex hull of the TAS which is constructed using Chan's algorithm [30].

The second is to check if the hyperbola or an ellipse overlaps with the TAS using a filter algorithm and discard any constraint that does not overlap with the convex hull of the TAS. For the hyperbola, it uses the rectangle which is perpendicular with the line between the foci with length equal to $2H_{max}$, width equal to $H_{max}$, and the start point is the midpoint between the foci, to approximate the hyperbola. For an ellipse, it uses circles with radius $a$ and the foci as the center to approximate the ellipses constraints where $a$ is half of the long axis length of the ellipse.

Third, check if the hyperbola or ellipse overlaps with the control circle. In this step, the filter algorithm not only needs to discard the shape constraints with zero or trivial area but also needs to store the area of the shape for each constraint and send them to the GPU along with all the possible shape constraints and the TAS for further calculation.

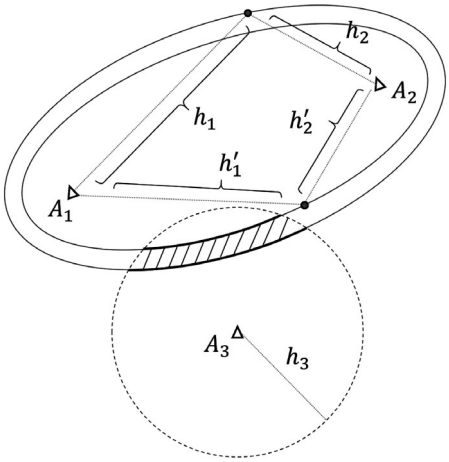



Fig. 4. A ellipse segment's area estimation example.

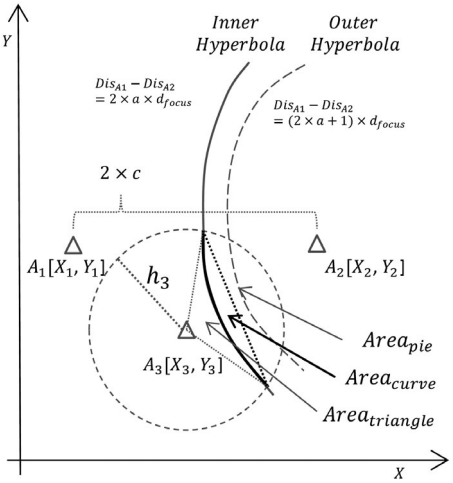



Fig. 5. A hyperbola segment's area estimation example.

### 3.3.2 Calculating the Area of the Shape Segments

To ensure the accuracy of the effective coverage ratio (ECR) as shown in Equation (4), the concise computation of the shape segments' area is essential. This paper defines the area of an ellipse segment $Area_{Segment}$ (in Eq. (6)) as the difference of the area of the outer ellipse overlapping with the control circle $Area_{outeroverlap}$ and the area of the inner ellipse overlapping with the control circle $Area_{innerOverlap}$(the shaded area shown in Fig. 4) where the control circle has the center $A_3$ and the radius $h_3$. The outer ellipse and inner ellipse share the same foci $A_1$ and $A_2$. Also, for any point on the outer ellipse, the distance from it to the foci ($h_1$ and $h_2$) meets $h_1 + h_2 = 2 \times a$ where $a$ is a predefined value. The distance from any point on the inner ellipse to the foci ($h_1'$ and $h_2'$) meets $h_1' + h_2' = 2 \times a'$.

$$Area_{Segment} = Area_{outerOverlap} - Area_{innerOverlap}. \tag{6}$$

The objective now is to calculate the overlapping area of the inner ellipse and the control circle, and the overlapping area of the outer ellipse and the circle. To do that, the first step is to transfer the ellipse to standard Cartesian format as in Equation (2) by rotating the foci to be horizontally aligned and shift the center of the ellipse to the origin point of the predefined coordinate.

The second step is to find the intersection point of the ellipse and the circle by solving the simultaneous equations of the Cartesian equation of the ellipse (Eq. (2)) and the Cartesian equation of the circle (Eq. (3) where $x_c$ and $y_c$ are the x and y coordinate of the circle center). The simultaneous equations can be transformed to be a quadratic equation (Eq. (7)) which has zero to four real number solutions for $y$. Each real solution indicates one intersection point of the ellipse and the circle.

$$\begin{cases} x = \frac{a\sqrt{b^2 - y^2}}{b} \\ y^4 + S_1 y^3 + S_2 y^2 + S_3 y + S_4 = 0 \\ m = b^2 - a^2 \\ n = a^2 b^2 + b^2 c_x^2 + b^2 c_y^2 - b^2 r^2 \\ S_1 = -4b^2 c_y / m \\ S_2 = (2mn + 4b^4 c_y^2 + 4b^2 a^2 c_x^2)/m^2 \\ S_3 = -4b^2 c_y n / m^2 \\ S_4 = (n^2 - 4b^2 c_x^2 a^2 b^2)/m^2 \end{cases}. \tag{7}$$

The third step is to simplify and adapt the algorithm in [31] for computing the overlapping area of an ellipse and the circle. The detailed procedures and algorithms are described in the Appendix, available in the online supplemental material.

A hyperbola's segment constraints are defined as the overlapping area of a hyperbola with one hop width and a control circle with a given radius. The hyperbola is defined with two anchor nodes which are its foci. The control circle is defined with one anchor node which is its center. To find all possible hyperbola segments, The proposed algorithm uses three layers nested for-loop to iterate through the anchor nodes $A_1, A_2, A_3$ as shown in Equation (5). For each shape, it iterates hops from $h_{start}$ to $h_{end}$. It also calculates the average one-hop distance between the foci of the hyperbola as $d_{focus}$ and the average one-hop distance between all three anchor nodes as $d_{avg}$ which is used to estimate the overlapping area of the hyperbola segment and the control circle discussed in Fig. 5.

For example, in Fig. 5, assume that the x-coordinate of the anchor nodes is $X_{ID}$ where "ID" is the anchor node's identification number and the y-coordinate of the anchor nodes is $Y_{ID}$. Assume that the euclidean distance between $A_1$ and $A_2$ is $2 \times c$. The overlapping area of the hyperbola of $A_1$ and $A_2$ and the control circle of $A_3$ is defined as $Area_{hyper} = Area_{innerHyper} - Area_{hyperbola}$. The inner hyperbola meets $Dis_{A1} - Dis_{A2} = 2 \times a \times d_{focus}$. Here $Dis_{A_{ID}}$ is the distance between any point in the hyperbola to the focus with given 'ID'; 'a' is a positive integer less than $H_{max}$. The overlapping area of a hyperbola and the control circle can be calculated as follows:

$$Area_{hyper} = Area_{pie} + Area_{triangle} - Area_{curve}.$$

As shown in Fig. 5, the $Area_{curve}$ is the area between the line connected to the two intersection points and the hyperbola curve between the two intersection points. This area is the result of the definite integral of the hyperbola function minus the line function. The detailed algorithms are given in the Appendix, available in the online supplemental material.

### 3.3.3 Compute the Effective Coverage Ratio ($ECR$) and Elect the Best Shapes

To calculate the $Area_{covered}$, a brute force method is used by testing hop constraints pixel by pixel. Different from the

TABLE 1
Payload Data Structure

| Descriptions | Starting Bytes | Length in Bytes |
|---|---|---|
| Message ID | 0 | 4 |
| Parent Node address | 4 | 2 |
| Hop counts | 6 | 1 |
| relaxation parameter | 7 | 1 |
| constraints for relax | 9 | 6 |
| routing constraints | 14 | 100 |

first step which uses a lot of condition branches, the second step has few branches. Thus, GPU can accelerate its computing. The NVIDIA CUDA kernel that calculates the $ECR$ is designed as follows:

For the ellipse arc, if the distance from any pixel in the TAS to $A_1$ ($Dis_{A_1}$) and $A_2(Dis_{A_2})$ obeys $Dis_{A_1} + Dis_{A_2} \leq 2 \times d_{avg} \times a$ and any pixel in TAS to $A_3(Dis_{A_3})$ obeys $Dis_{A_3} \leq d_{avg} \times h_3$, then that pixel is covered by the arc shape. For the hyperbola, if the distance from any pixel in the TAS to $A_1$ ($Dis_{A_1}$) and to $A_2$ ($Dis_{A_2}$) obeys $Dis_{A_1} - DisA_2 = 2 \times a \times d_{focus}$ and the distance from any pixel in the TAS to $A_3(Dis_{A_3})$ obeys $Dis_{A_3} \leq h_3 \times d_{avg}$, then that pixel is covered by the hyperbola shape (note that here all the notations used are the same as in Equation (5) and Fig. 5). The GPU algorithm considers both the total number of pixels in TAS covered by the element shapes and the Effective Coverage Ratio ($ECR$). In each iteration, the proposed greedy algorithm only selects the shape which provides the maximum value of the Greedy Factor (GF) which is the number of pixels multiplied by cubic $ECR$ as shown in Equation (8)

$$GF = Area_{covered} \times ECR^3. \tag{8}$$

So each iteration will eliminate some pixels of TAS, which also exists in the best shape in Section 3.3.3. Then, the newly covered pixels is calculated for each possible shape with the updated TAS. Also, instead of calculating a new $ECR$ for each shape, the algorithm reuses the $ECR$ calculated in the first iteration. The shape with the maximum GF is chosen as the best shape. This procedure is repeated until the size of the updated TAS is less than $1 - Th$ of the original size, where $Th$ is the predefined coverage threshold. This procedure is discussed in Fig. 22, which generates the final sequence of shapes that cover most of the TAS. If the trajectory encoding message exceeds the packet size limitation, shown in Table 1, the POI needs to be divided into two, and create two separate trajectories with two different gateway nodes and encode them separately. It is shown in Fig. 1.

### 3.4 Routing Decision for Wireless Sensors

The message structure of TinyOS has an 11-byte header that includes the sender's address, the type, and the group data. The payload structure, defined in Table 1, is used for the implementation of DV-TE-R and adapted DV-TE-BR discussed in Section 4.

The counter-based routing decision is also used to mitigate the broadcast storm effect [5]. In the proposed counter-based routing decision, each wireless sensor uses two kinds of criteria to decide if it should forward the routing packet or not. The first is to check if it meets any constraint of the encoded trajectory ($flag_C$). The second is to check if the counter used to count the number of the nearby redundant re-broadcasting for the same packet reaches the threshold ($flag_T$).

As discussed in Section 3.3, the ellipse constraint has a size of four bytes: $A_1$, $A_2$, $A_3$, $a$, $h_3$. For any sensor node, setting $flag_C$ to be true means its DV-Hop entry of $A_1$, $A_2$, $A_3$ obeys Equation (9). The hyperbola constraint has a size of five bytes: $A_1$, $A_2$, $A_3$, $a$, $h_3$. For the same sensor node, $flag_C$ should be true if its DV-Hop entry of $A_1$, $A_2$, $A_3$ satisfies Equation (10). To distinguish the ellipse constrains from the hyperbola constraints, the ellipse constraint sets the $a$ value to be negative while the hyperbola constraint uses positive $a$ value.

$$\begin{cases} floor(DV_{Hop}[A_1] + DV_{Hop}[A_2])/2 == |a| \\ DV_{Hop}[A_3] \leq h_3 \end{cases} \tag{9}$$

$$\begin{cases} floor(DV_{Hop}[A_1] - DV_{Hop}[A_2])/2 == |a| \\ DV_{Hop}[A_3] \leq h_3 \end{cases}. \tag{10}$$

The counter-based routing decision is first proposed in [5]. Once a sensor node broadcasts a packet to its neighbors, the neighbors will be listening to the channel for a random amount of time before it forwards the packet. During the listening period, the sensors will count the number of times the same packet has been forwarded. If it exceeds the counter threshold, it will set the $flag_T$ to be false so only the sensors with $flag_C == true$ and $flag_T == true$ will forward the routing packets.

Another case is for a low-power listening WSN where sensors hibernate for most of their lifetime. The sensors will rebroadcast immediately if they find they satisfy Equations (9) and (10) ($flag_C == true$ ) and initialize a counter with value 0. Then, they will stop broadcasting when there is a timeout or when their rebroadcasting neighbors' number reaches the counter threshold ($flag_T == true$).

### 3.5 Sample Routing Result and Analysis

The proposed geospatial area encoding algorithm works for any shape and trajectory. Fig. 6 shows the encoding algorithm on some sample trajectories. For each of the routing trajectories, assume anchor node ID is one byte long and the number of hops is one byte long. As shown in Table 2, the length of the encoded trajectory is 23, 54, and 89 bytes when encoding without the use of ellipse constraints, and the length of the encoded trajectory is 21, 51, and 86 when encoding with the ellipse constraints. The red shapes, which represent the encoded trajectory, shown in Figs. 6a, 6b, and 6c are the cascaded circle arc and hyperbola shapes. Fig. 9 shows the encoding result of trajectory handwriting "A" and hand drawing park boundary when encoding with ellipse and hyperbola constraints. JPEG compression algorithm is lossy for images. The above JPEG example has a $64 \times 64$ resolution while has only about 800 bytes size. Assume that there are "n" anchor nodes in the local edge network, each anchor node floods at most "r" hops, and the TAS has "m" entries. Then the time complexity of finding the best ellipse arc and the best hyperbola segment is

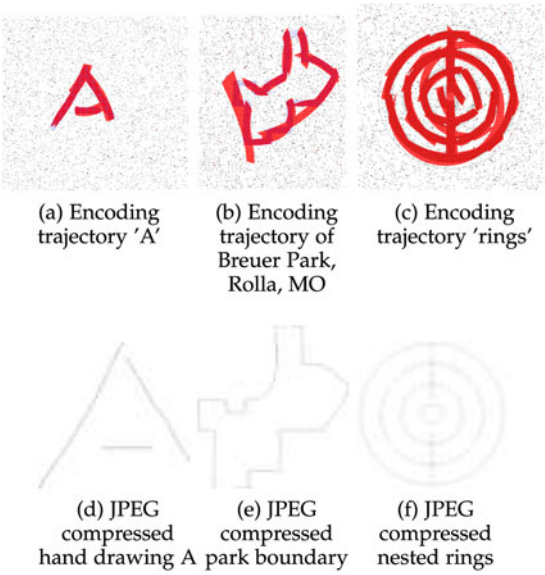



Fig. 6. Example encoding with the circle and the hyperbola constraints and the compressed trajectory using JPEG.

TABLE 2
Encoding Result for Sample Trajectories

| Trajectory type | A | outline | Circles |
|---|---|---|---|
| Number of circle arcs (without ellipse) | 3 | 12 | 17 |
| Number of hyperbolas (without ellipse) | 2 | 1 | 4 |
| Message length(byte) | 23 | 54 | 89 |
| Number of ellipse arcs (with ellipse) | 2 | 6 | 12 |
| Number of hyperbolas (with ellipse) | 2 | 4 | 5 |
| Message length(byte) | 21 | 51 | 86 |
| Compressed size with JPEG(byte) | 867 | 887 | 934 |

$O(mn^3r^3)$. Another experiment is conducted with anchor nodes from 20 to 80, and the TAS is from 10,000 to 100,000. Fig. 7 shows when the number of anchor nodes increases, the CPU time of calculating the area of all possible shapes has a polynomial growth. Fig. 8b shows that the GPU time of finding the best shape also has polynomial growth when the number of anchor nodes increases. However, Fig. 8a indicates that the GPU time of finding the best shape has a linear relationship with the size of TAS.

## 4 Adapted DV-Hop Based Data Collection Scheme for Low-Power WSN

To ensure high QoS in a WSN, the DV-Hop based trajectory encoding and routing protocol (DV-TE-R) has been proposed. For low-power WSN, which can be deployed in a harsh environment, the density of the network topology can be heterogeneous. Somewhere in the region, the sensors may be sparsely deployed, or the routing path could be obstructed by some "holes" shown in Fig. 10. Now the user wants to forward a packet from node $S_i$ to $S_o$ through an arc with center $A$ and radius $h$ hops with 1 hop width. Although the DV-hop of both the nodes $S_i$ and $S_o$ is $h$, these two nodes are not directly connected because of an obstacle between them. If using local broadcasting, for example in Fig. 10, to fix the routing path, at least 4 hops extra broadcast is required which is a huge overhead. Therefore, the routing protocol adopts a bridge on the

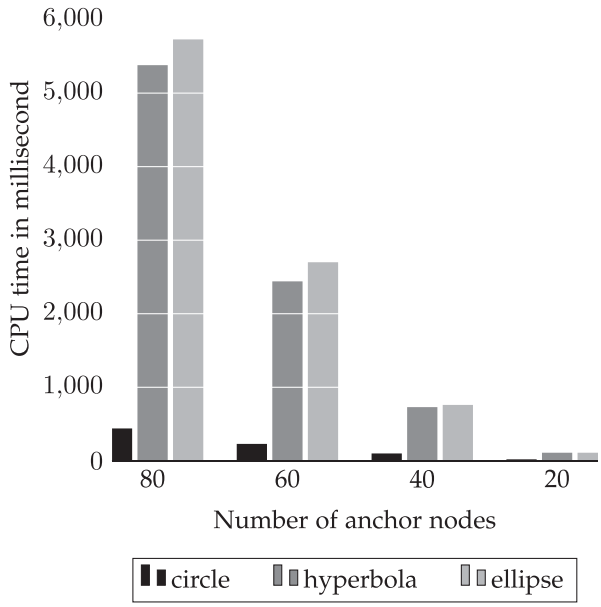



Fig. 7. CPU time calculating area of all possible shapes.

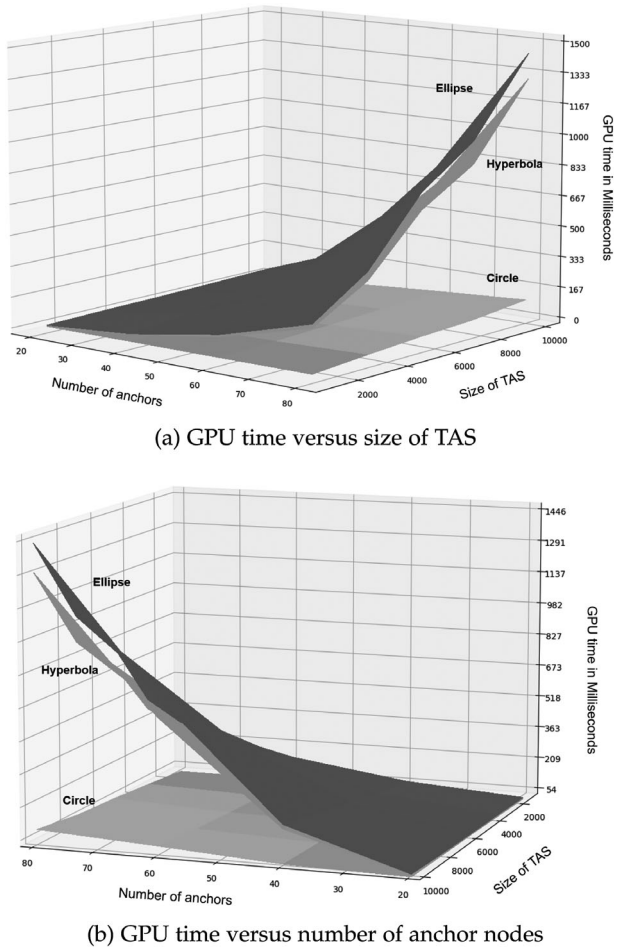



Fig. 8. GPU time finding the best shape.

edge adaption (DV-TE-BR) that could connect a broken routing path with minimum overhead.

After a forwarder node has forwarded the routing packet and has not overheard any rebroadcast from its neighboring

(a) Hand drawing 'A' (b) Hand drawing boundary of Breuer Park,Rolla,MO

Fig. 9. Example Encoding of ellipse and hyperbola.

nodes nor acknowledgment from the sinks, it will start iterating its valid constraints, relax those by one hop, note all the changes, and rebroadcast the packets again. If it receives the rebroadcast from its neighbors, it will stop iterating and go to sleep immediately. If a node receives the relaxed-constraint packet, it will tighten the constraint by one, and repeat the previous procedure until recovering the original constraint. Note that both $h_1$ and $h_2$ are relaxed for the arc constraint, and both $a$ and $h_3$ are relaxed for hyperbola constraint.

For example, like Fig. 10, the nodes $B_1, B_2, B_3, B_4$ have an increased DV-hop entry of $A$ from $h+1$ to $h+2$. Thus, the sensor node $S_i$ needs to relax the hop constraints by one. Then, the node $B_1$ needs to relax the hop constraint by one more. For $B_2$, it will hold the constraints as of $B_1$. The constraint is tightened by one for $B_3$, and so does $B_4$. Finally, the route is fixed after $B_4$ forwards the packet to $S_o$.

## 5 Experiments and Results

The experiments are conducted with a simulation tool TOSSIM and a real sensor network test-bed. The simulation uses a randomly created large-scale wireless sensor network as the working area which contains both heterogeneous and homogeneous topology. It uses real-world trajectory data (taxi-trajectory [32]) for sensor data collection. The experiments also include the performance evaluation of the proposed data collection algorithm in WSN with mobile nodes.

The structure of our experiments is listed as follows: First, an experiment to validate the effeteness of the proposed trajectory-based routing protocol in a real wireless sensor network with 28 nodes. Second, an experiment to evaluate the proposed data collection scheme in a large static WSN with a mobile sink using the TOSSIM simulator. The simulation uses the real-world taxi trajectory dataset. Third, an experiment to test routing in the shortest path to the mobile sink

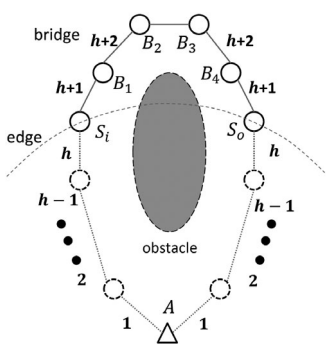



Fig. 10. Example of the bridge on the edge adaption.

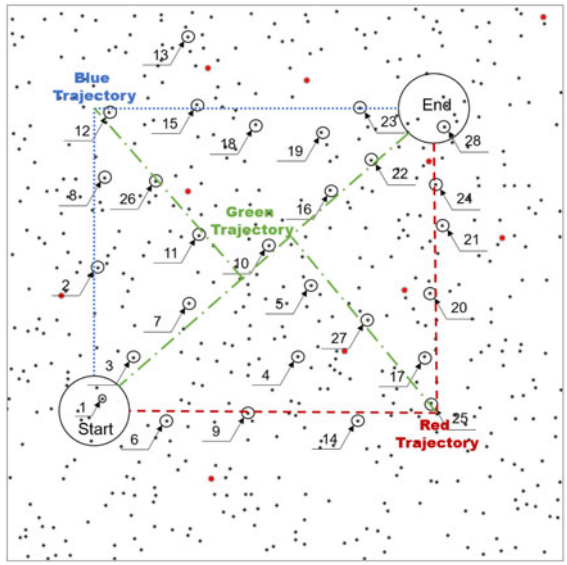



Fig. 11. Experimental WSN and routing trajectory.

and compared the delay and energy consumption comparing to ring routing and nested routing. Fourth, a Python program to evaluate the proposed data collection scheme in a WSN with mobile sensor nodes. It shows the data collection area scatters with the movement of sensors.

### 5.1 Performance Using a WSN Test-Bed

This real-world experiment contains a small wireless sensor network, which is a mix of 16 TelosB sensors [33] and 12 MicaZ sensors [34], with minimal radio power configuration, and hard-coded DV-Hop information of 10 virtual anchor nodes. The experiments test the performance of broadcasting the data collection requests toward the location of interest through a given trajectory. The performance metrics were mainly focused on the coverage ratio and latency. The baseline comparison was with the counter-based broadcasting scheme discussed earlier.

The 28 wireless sensors are coded with the proposed DV-Hop based broadcasting protocol, and the counter based broadcasting scheme. The sensors are labeled and hard-coded with the DV-Hop of 10 virtual anchor nodes. The values of the DV-Hops are generated with a simulation tool that simulates a WSN with 500 nodes and 10 anchor nodes. It selects 28 nodes from the 500 nodes in the WSN. Three trajectories are used in this experiment as shown in Fig. 11. The blue trajectory (dot line) has a "Γ" shape and is located near the top left corner of the WSN. The red trajectory (dash line) has a "⌐" shape and is located on the bottom right corner of the WSN. The green trajectory (dash-dot line) has a shape that looks like an "$X$" and is located in the middle of the WSN.

Next, the wireless sensor network deployed follows the same topology as in the simulation tool based on their labels. Node 1 is the start node and node 28 is the sink node for all three trajectories. In the center of the WSN, the green trajectory is a multi-casting route that forwards packets from node 1 to nodes 28, 25, and 12 at the same time.

Then, use the simulation tool to generate the routing messages covering the colored trajectory, and sending the routing messages through the gateway sensor (source node) connecting the laptop near node 1.

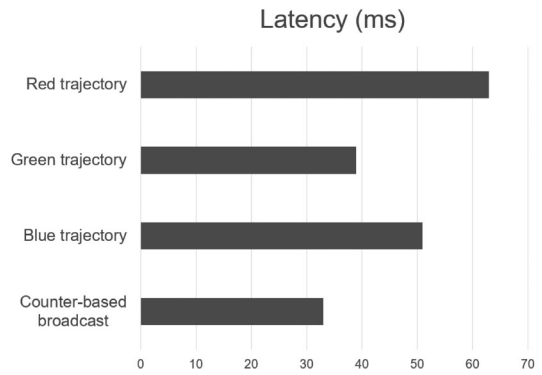



Fig. 12. Latency of multi-hop routing when disseminating data collection message with proposed data forwarding approach and counter-based broadcast.

Once the sensors rebroadcast the received data request messages, they will blink their LED light until enough neighbors (larger than the counter threshold) rebroadcast or when it timeouts. Also, the predefined destination nodes (POI) will route the sensing data back to the gateway sensors through its parent's node. The time elapsed between the time the source sending the data collection packets and the time the source received back the sensing data is considered to be the latency.

The experiment shows that the proposed approach can successfully disseminate messages to the desired path by only modifying the constraints in the routing message. Only the sensors on the trajectory that meets the constraints will be activated to rebroadcast the message. In Fig. 12, the multi-hop latency is similar to the broadcasting approach for the green trajectory. The reason is that the routing decision of the proposed DV-Hop based data forwarding approach is made by comparing the routing constraints and the DV-Hop stored in the sensor, which is trivial. Most portion of the delay is caused by the predefined random channel listening period which is similar to the counter-based broadcasting approach (It is used to mitigate the broadcasting storm effect). The proposed approach has better latency when routing through the green trajectory. The reason is that the green trajectory has fewer hops from the start node to the sink node. The broadcasting approach has better latency compared to the proposed data forwarding approach because it can always route messages to the destination through the minimum hop counts. However, as a trade-off, the broadcast approach can only flood through all

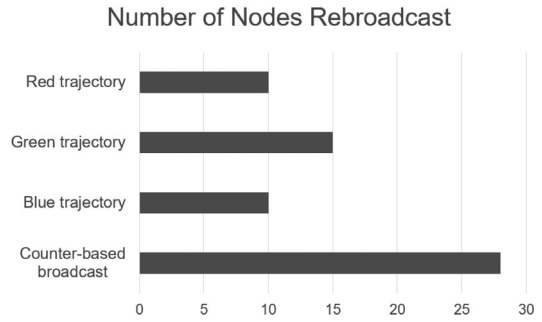



Fig. 13. Total number of nodes rebroadcasting when disseminating data collection message with proposed data forwarding approach through trajectory and counter-based broadcast.

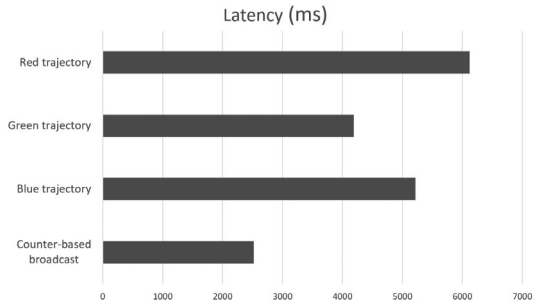



Fig. 14. Latency of multi-hop routing when disseminating data collection messages with proposed data forwarding approach and counter-based broadcast in LPL WSN.

the nodes while the proposed DV-Hop based data forwarding approach not only can route on the trajectory but also reduce the overhead caused by the redundant rebroadcasting. Fig. 13 shows that the baseline(counter-based broadcasting) approach will flood the whole network and will cause all the nodes to rebroadcast even in the small wireless sensor network.

Another experiment is executed with low power listening(LPL) [35] configured WSN. In LPL mode, the sensors fell asleep frequently and only wake up for a small period to listen to the channel. The proposed approach sets the sleeping time as 600 ms and wake up time as 10 ms in one cycle. It is challenging to do routing in an LPL WSN where the network topology is unstable as most of the neighboring sensors fell asleep. The adaption of the routing approach for LPL WSN, discussed in Section 4, doesn't need to rely on the routing table. However, unlike the existing LPL broadcasting scheme, the proposed approach lets the current broadcasting nodes to keep broadcasting until the number of rebroadcasting neighbors reaches the predefined counter threshold or it times out. Fig. 14 shows that although the counter based broadcasting has a lower delay due to flooding, the proposed data forwarding protocol still has an acceptable multi-hop routing delay.

### 5.2 Simulation Results

The simulation experiment creates random heterogeneous WSNs and routes data collection messages on the real-world taxi trajectory [32]. Coverage performance is defined as follows. The correct coverage ratio is the number of nodes rebroadcasting while on the trajectory over the number of the nodes supposed to rebroadcast (total nodes on the taxi trajectory). It shows the effectiveness of the proposed protocol (higher ratio indicates high accuracy). The redundant rebroadcast ratio is the number of the nodes not in the trajectory while still rebroadcasting over the number of nodes supposed to rebroadcast. A higher redundant rebroadcast ratio indicates a higher overhead of bandwidth and energy consumption.

The experimental parameters are defined in Table 3. The trajectory and POI encoding are executed using a desktop, which acts as a fog server, with a Xeon E5-1620 v2 and an Nvidia RTX 2070 GPU. The POI is located within a WSN distributed in a 2800m by 1700m area. The density of the WSN is less than 0.5 percent where the density is defined as the average number of neighbors over a total number of sensor nodes. To save energy, all the wireless sensors work

TABLE 3
Parameters for the Experiments

| | |
|---|---|
| Area of deployment | 2800×1700 m |
| Number of sensor nodes | 5000 |
| Communication range | 40-100 m |
| Number of edge devices | 30 |
| Broadcasting hop limitation | 30 hops |
| LPL sleeping time | 600ms |
| LPL wake time | 10ms |
| Energy model | MicaZ |
| Coverage threshold | 90% |
| Number of anchor nodes | (20 - 100) |
| Anchor cover range | 20 hops |

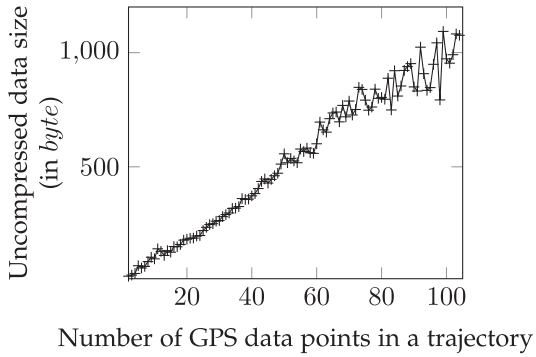



Fig. 16. Property of taxi trajectory dataset.

under low power listening mode, where each sensor node only wakes up for a few milliseconds to listen to the channel. The WSN is simulated using TOSSIM. The experiment also uses powerTOSSIM-Z to estimate the energy consumption of the activated sensor nodes. There are 30 local edge nodes randomly deployed in the WSN field. The local edge nodes act as the gateway that will broadcast the encoded data requests' packets and collect the data from the WSN. The performance metric used includes the compression ratio, reliability, average delay in data reporting, and energy consumption.

As mentioned above, the experiments use real-world taxi-trajectory data [32] as the routing trajectory. Each line of the trajectory data contains the trajectory of a taxi trip in the city of Porto in Portugal. The trajectory is represented as a list of 8 bytes of GPS data (latitude and longitude) sampled every 15 seconds. The 2 GB data-set contains trajectories of different shapes, lengths, starting, and ending locations. To use these trajectories in the experiments, the first step is to pre-process the data-set by removing the abnormal trajectory and aligning the trajectories in the center of the WSN. To remove the outliers, trajectories that contain abnormal itinerary like where the taxi has abnormal speed, are discarded so that the data-set only contains valid GPS data points for each itinerary. The third step is to reconstruct the itinerary from GPS data points and map it to a 2D trajectory which is the area in the fixed WSN field (TAS). To determine the rectangle field of the WSN and align with the taxi trajectory on it, the fourth step is to use Chan's algorithm [30] to generate the convex hull of the trajectory and then find the minimal rectangle that could wrap the trajectory as shown in Fig. 15. The largest length and width of the minimal wrapping rectangles of all the taxi trajectories are chosen as the outline parameter of the rectangle field of the WSN. Then all the trajectories are shifted and rotated to align to the center of the WSN field. The last step is to set the thickness of the trajectory line to be the average one-hop radio distance of the sensors. Fig. 15 shows two trajectory examples with 60 (left) and 189 GPS data points.

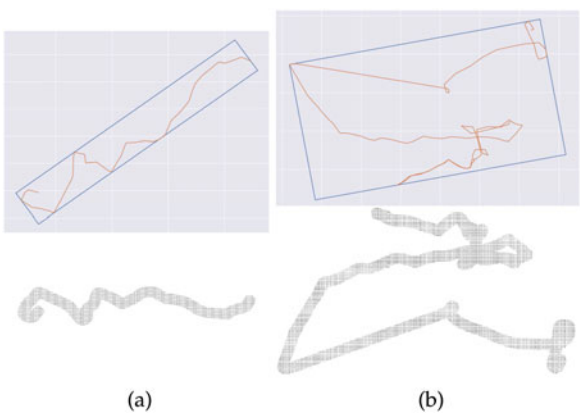



Fig. 15. Find convex hull and the minimal surrounding rectangle for taxi trajectory with 60 GPS data points(left) and 189 GPS data points(right) and the trajectories after pre-process.

The statistics of the dataset are shown in Fig. 16 where the $X$-axis shows the number of GPS data points of the trajectory, and $Y$-axis is the size of the set of node IDs that reside on the trajectory. The figure shows that the trajectory with more GPS data will cover more sensor nodes. In Fig. 20 and Fig. 21, where the $X$-axis is the number of GPS data in a trajectory and the $Y$-axis is the compression ratio, 500 trajectories were sampled, the number of GPS data points ranging from [2, 20],[21, 40],[41, 60],[61, 80], and [81, 105] and the number of anchor nodes in the set [20, 40, 60, 80, 100] which is (0.4, 0.8, 1.2, 1.6, 2 percent) of the total number of nodes in the network. The compression ratio ($CR$) is defined as the uncompressed-data size over the compressed data size using the proposed encoding algorithm. Here the uncompressed data size is the size of a list of sensor IDs that reside in the trajectory. The results show that when the number of anchor nodes increases, the $CR$ also increases. It is more likely to find the closest 'shapes' when the number of anchor nodes is large, which costs fewer iterations to cover the area. Note that when the number of GPS data points increases, the $CR$ is not increased when encoding with 20 or 40 anchor nodes. A possible reason is that when there are fewer anchor nodes, redundant shapes may be selected in the later iterations, which increases the size of the encoded message. For example, in Figs. 6b and 6c, more shapes that overlap with other shapes than that in the case of Fig. 6a.

When the mobile nodes change location, they will update the DV-Hop by querying the nearby sensor nodes. As discussed in Section 3.1, the new DV-Hop to anchors will be the minimum hop count of all current neighbors plus one. However, when the neighbors contain the anchor nodes, the new DV-Hop will use the anchor nodes' minimal DV-Hop as the only reference. A trajectory routing example for a sample trajectory with 66 GPS data in Fig. 17 shows the coverage area increases steadily when the network has 50 percent of the mobile nodes. Fig. 18 shows the result of how the correct

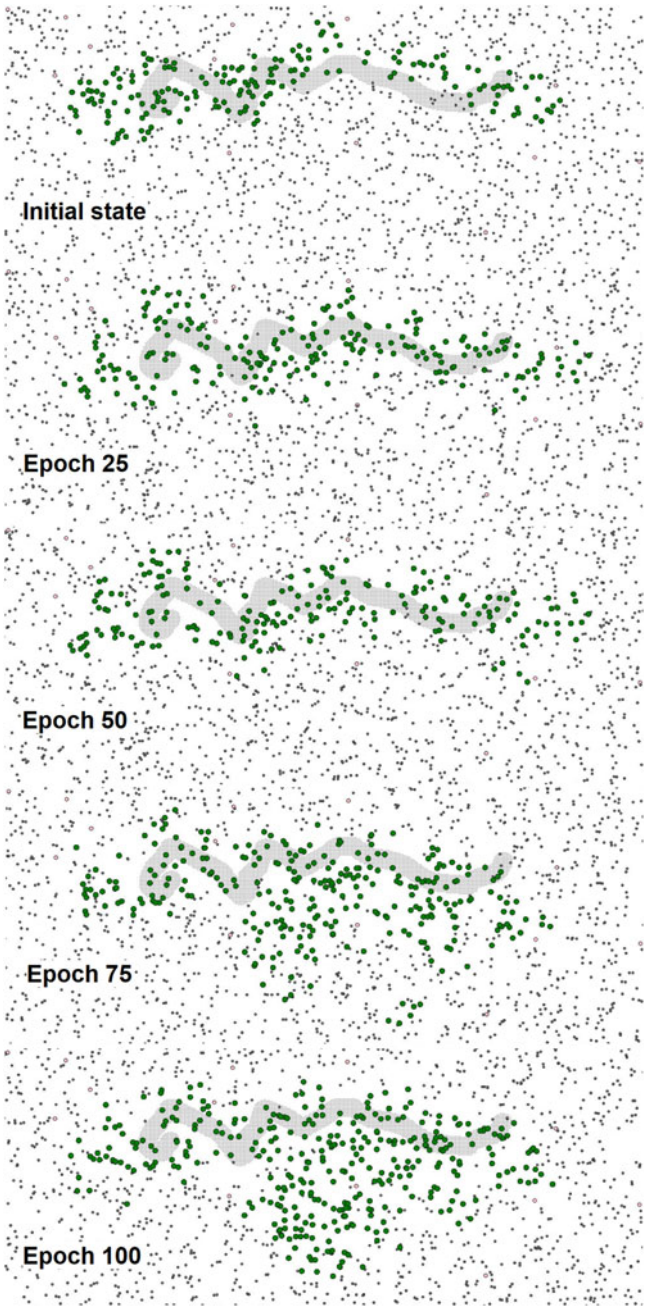



Fig. 17. With 50 percent mobile nodes for each epoch, the changing of coverage for taxi trajectory of Fig. 15a.

coverage ratio changes with 20, 50, and 80 percent mobile nodes in a given period (0-100 epoch). In each epoch, the randomly selected sensors will move 1.5 times of its radio distance. The correct coverage ratio will increase due to the increase in the total coverage area. However, the trade-off is the redundant rebroadcast ratio also increases as shown in Fig. 19.

For the reliability experiment. Since the proposed trajectory encoding algorithm uses basic shapes (ellipse arc and hyperbola segments) to approximate the trajectory, there is a possibility that the encoding will fail when no suitable shape is found. Here, an encoding failure is defined as when the encoding algorithm cannot find any combination of shapes that will cover a certain threshold ratio of *TAS*, which is defined as the predefined coverage ratio. For

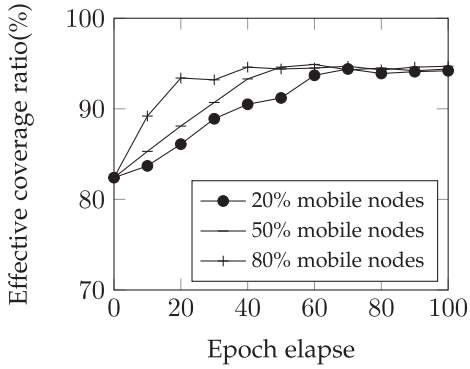



Fig. 18. Average correct coverage ratio.

example, when the threshold is set to 85 percent, the hops constraints represented shapes must overlap with more than 85 percent of the trajectory area. The lower the threshold, the higher is the encoding successful rate. However, a threshold lower than 85 percent is not recommended, as the routing reliability (the rate of successfully routing the data request message to the POI) will be affected due to the uncovered gaps between the routing paths. This experiment is to find the relationship between the number of anchor nodes, success rate (the percentage of encoding that does not fail), and coverage threshold. The result in Fig. 22 shows that as the number of anchor nodes increasing, the success rate of encoding will increase. However, increasing the coverage threshold will decrease the success rate of encoding. Note that When using ellipse constraints, the success rate is higher than using naive circle constraints when the number of anchor nodes is 20. The reason is the ellipse use three anchor nodes which provide more combination than the circle constraints with two anchor nodes.

The next experiment compares the reliability of the proposed encoding and routing protocol with the ring routing [14] and the nested routing [15], which both enable data routing to a moving sink (mobile edge device that can move in some applications) by relaying the data to a circular area where the nodes know the updated location of the mobile edge device. The area of a nested ring is in the middle of the sensing field, as shown in Fig. 6c. This experiment uses 50 anchor nodes for the proposed algorithm. The reliability of ring routing is defined as the success rate of generating a ring structure. The reliability of nested routing is the success rate of generating any one of the ring structures within its

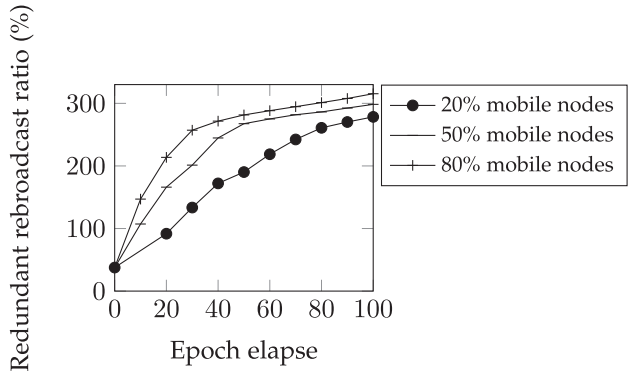



Fig. 19. Average redundant rebroadcast ratio.

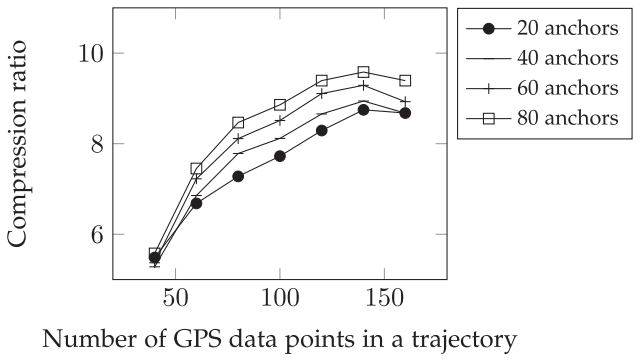



Fig. 20. Compression ratio of the encoding without using an ellipse.

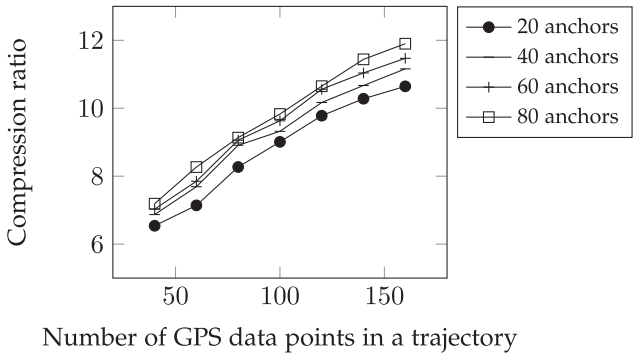



Fig. 21. The compression ratio of the proposed DV-Hop based trajectory encoding algorithm (with ellipse) for different trajectory sizes.

nested ring structure. The reliability of the proposed DV-Hop based trajectory encoding and routing (DV-TE-R) and its bridge on the edge (DV-TE-BR) adaption is the success rate of generating the encoded message of the trajectory and routing the data request packets to the nodes within *POI*. The result is shown in Fig. 23. When the average number of neighbors of each node is smaller than 8, which is 0.16 percent of the total number of nodes, both protocols have low reliability. The nested routing protocol has better reliability performance than ring routing because it has redundant rings. The bridge on the edge adaption improves the reliability of DV-TE-R by relaxing the hop constraints. When the average number of neighboring nodes is greater than 16, which is 0.32 percent of the total number of nodes, the reliability of DV-TE-BR is greater than 99 percent. It achieves the best reliability performance compared to ring routing and nested ring routing. In the rest of the experiments, by default, the following experiments use DV-TE-BR.

The next experiment assumes that mobile edge devices will move randomly with different speeds in the local IoT network with the configuration, as shown in Table 3. The average delay in data reporting is the time when the moving mobile edge devices receive the data minus the time when the source reports the data. The proposed DV-TE-BR fixes the current routing path by letting the mobile edge devices update their locations periodically. Thus, it cannot always guarantee the shortest reporting path as the ring routing and nested routing methods do. Although the ring routing and nested routing provide the current location of the mobile sink, fetching this information from the ring or nested ring for the source node causes the delay overhead of one round trip to the closest ring. Thus, the delays of ring

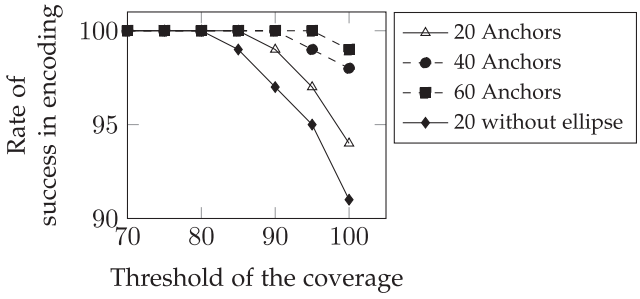



Fig. 22. Experiment of successful encoding rate with different number of anchor nodes and coverage threshold.

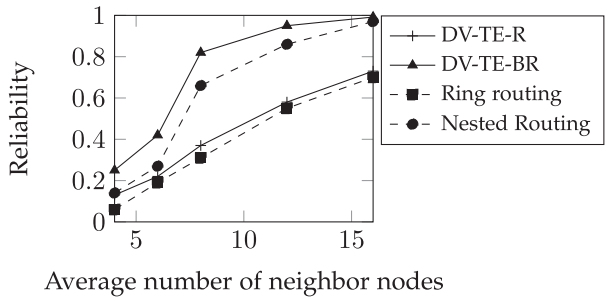



Fig. 23. The reliability of DV-TE-R, DV-TE-BR, Ring routing, and Nested routing with different number of neighbors.

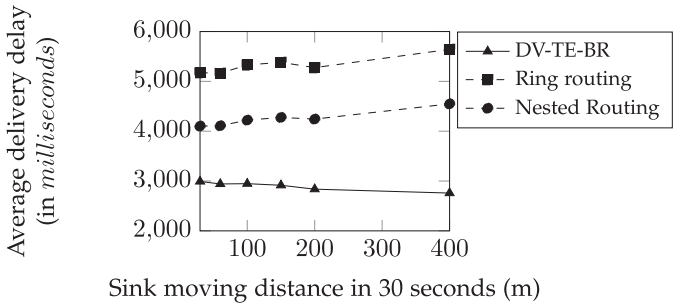



Fig. 24. The average delay in data reporting compared with state-of-the-art schemes.

routing and nested ring routing are still higher than the delay of DV-TE-BR. In addition, the counter-based routing strategy of DV-TE-BR reduces the waiting delay for the low-power listening WSN because the first awaked node could start routing, while ring routing and nested routing have to wait for specific routing nodes within its routing table. Nested routing has a better delay performance than ring routing because its average shortest distance from the source to the rings is shorter than the ring routing. The delay performance of data reporting is shown in Fig. 24.

The following latency and energy consumption experiments compare the proposed scheme with the state-of-the-art counter-based broadcasting algorithm. The experimental set-up, which uses the taxi trajectory data, is shown in Table 3. It uses TOSSIM to simulate the routing of data request messages and data reporting packets and visualize the results using Python. Fig. 26a shows a sample trajectory being encoded using hops constraints represented shapes, and Fig. 26b shows a sample output of the visualized routing result where the red dots are the sensors that forwarded a message and the green dots are sensors that received a message.

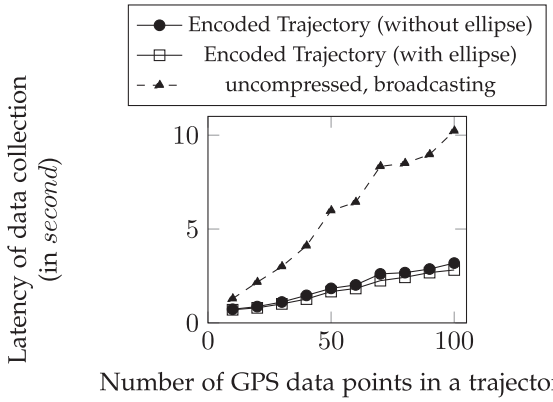



Fig. 25. Average delay from starting broadcast request till receiving all the data from the POI.

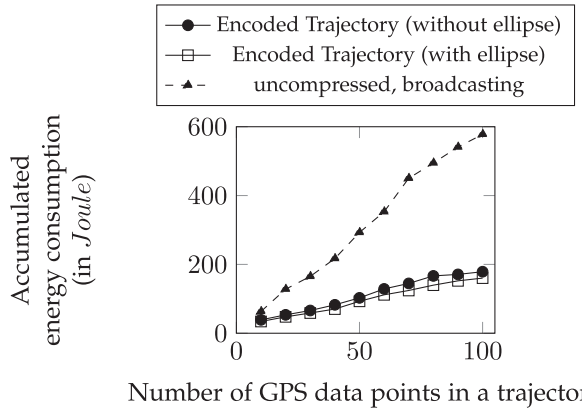



Fig. 27. Accumulated energy consumption in fetching the data from the POI.

The latency in collecting the requested data is an important factor in meeting the quality of service. Broadcasting is the fastest way to flood the data request into the whole network. The proposed data collection scheme also broadcasts the data request to the POI. However, as opposed to flooding approaches [7], only the nodes in a hop constraint defined trajectory can rebroadcast. Thus, energy consumption and bandwidth usage are minimized. Fig. 25 compares the latency of the local edge devices receiving all the sensing data of the POI from the local WSN, which are working under low power listening mode with a 660 ms sleeping and waking period. All local edge devices will broadcast the data request messages. The sensor nodes that receive the data request packets will be awake and send the data back to the nearest edge devices if they are at the POI. The experimental result shows that transmitting encoded data request messages could reduce latency because broadcasting a compressed message requires fewer packets than an uncompressed message which includes IDs of all the sensor nodes residing within the POI. The proposed scheme, which encodes trajectories with ellipse-circle constraint, achieves better latency performance than the trajectory encoding protocol with only circle-circle constraints due to its higher compression ratio.

The energy consumption experiment is simulated with powerTOSSIM-Z, which is an energy simulation tool for wireless sensors. It uses the micaZ energy model and can measure energy consumption at the packet level. The result in Fig. 27 shows that the proposed data collection scheme consumes less energy than the broadcasting approach. The proposed scheme, which encodes trajectories with ellipse-circle constraints, achieves better energy performance than the trajectory encoding protocol with only circle-circle constraints due to the higher compression ratio. The last experiment is to compare the average number of rebroadcasting nodes of the proposed DV-hop based trajectory encoding and routing scheme (DV-TE-BR) verses the state-of-the-art counter-based broadcasting [7] for each single data request packet. The result shows that the proposed DV-TE-BR scheme reduces the number of redundant rebroadcasting packets (142 versus 2491) by 94 percent and thus, saves bandwidth usage in the WSN.

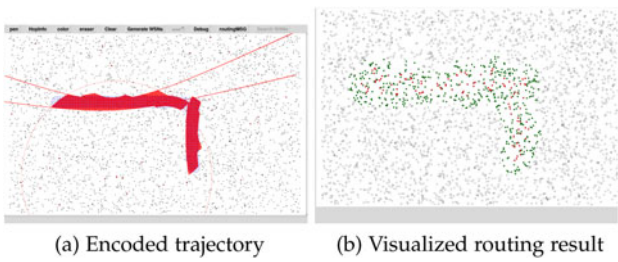



Fig. 26. Sample routing example.

## 6 Conclusion and Future Work

The proposed trajectory encoding and data collection algorithms for IoT applications have improved energy efficiency, reduced latency, and achieved reliable performance when fetching data from the POI in the local fog network without using GPS coordinates. In addition, with the use of virtual coordinates, location anonymity is achieved for the source, sink, and intermediate nodes in the routing path, as only the secure server in the local fog knows the anchor nodes' locations. Besides, the use of ellipse and hyperbola constraints increase the encoding accuracy and compression ratio. In the future, the plan is to solve the real-time event detection problem in multi-hop IoT network using the proposed approach and the conditional random field [36]. The plan is also explore to extend the proposed scheme for under water WSN.

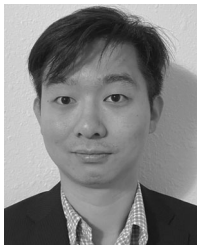
## Acknowledgments

This work was supported in part by a NSF Grant CNS 1461914 and DOE Grant P200A180051

## References

[1] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.
[2] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol.*, 2015, pp. 73–78.
[3] Y. Wang *et al.*, "Regularity and conformity: Location prediction using heterogeneous mobility data," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1275–1284.
[4] J. Wang, J. Tang, G. Xue, and D. Yang, "Towards energy-efficient task scheduling on smartphones in mobile crowd sensing systems," *Comput. Netw.*, vol. 115, pp. 100–109, 2017.
[5] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Netw.*, vol. 8, no. 2/3, pp. 153–167, 2002.

[6] C. Chen and C.-K. Hsu, "A distance-aware counter-based broadcast scheme for wireless ad hoc networks," in *Proc. IEEE Military Commun. Conf.*, 2005, pp. 1052–1058.
[7] J.-Y. Jung and D.-Y. Seo, "Counter-based broadcast scheme considering reachability, network density, and energy efficiency for wireless sensor networks," *Sensors*, vol. 18, no. 1, 2018, Art. no. 120.
[8] K.-P. Ng, C. Tsimenidis, and W. L. Woo, "C-Sync: Counter-based synchronization for duty-cycled wireless sensor networks," *Ad Hoc Netw.*, vol. 61, pp. 51–64, 2017.
[9] M. Yuksel, R. Pradhan, and S. Kalyanaraman, "An implementation framework for trajectory-based routing in ad hoc networks," *Ad Hoc Netw.*, vol. 4, no. 1, pp. 125–137, 2006.
[10] H. Labiod and N. Ababneh, "An efficient scalable trajectory based forwarding scheme for VANETs," in *Proc. 24th IEEE Int. Conf. Adv. Inf. Netw. Appl.*, 2010, pp. 600–606.
[11] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *Proc. 9th Annu. Int. Conf. Mobile Comput. Netw.*, 2003, pp. 96–108.
[12] S. Kumar and D. K. Lobiyal, "Novel DV-Hop localization algorithm for wireless sensor networks," *Telecommun. Syst.*, vol. 64, pp. 509–524, 2017.
[13] B. Nath and D. Niculescu, "Routing on a curve," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 155–160, 2003.
[14] C. Tunca, S. Isik, M. Y. Donmez, and C. Ersoy, "Ring routing: An energy-efficient routing protocol for wireless sensor networks with a mobile sink," *IEEE Trans. Mobile Comput.*, vol. 14, no. 9, pp. 1947–1960, Sep. 2015.
[15] R. Yarinezhad, "Reducing delay and prolonging the lifetime of wireless sensor network using efficient routing protocol based on mobile sink and virtual infrastructure," *Ad Hoc Netw.*, vol. 84, pp. 42–55, 2019.
[16] S. Sharma, D. Puthal, S. Tazeen, M. Prasad, and A. Y. Zomaya, "MSGR: A mode-switched grid-based sustainable routing protocol for wireless sensor networks," *IEEE Access*, vol. 5, pp. 19864–19875, 2017.
[17] L. Buttyán and P. Schaffer, "Position-based aggregator node election in wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 6, no. 1, 2010, Art. no. 679205.
[18] O. Banimelhem and S. Khasawneh, "GMCAR: Grid-based multipath with congestion avoidance routing protocol in wireless sensor networks," *Ad Hoc Netw.*, vol. 10, no. 7, pp. 1346–1361, 2012.
[19] Y.-P. Chi and H.-P. Chang, "An energy-aware grid-based routing scheme for wireless sensor networks," *Telecommun. Syst.*, vol. 54, no. 4, pp. 405–415, 2013.
[20] A. W. Khan, J. I. Bangash, A. Ahmed, and A. H. Abdullah, "QDVGDD: Query-driven virtual grid based data dissemination for wireless sensor networks using single mobile sink," *Wireless Netw.*, vol. 25, no. 1, pp. 241–253, 2019.
[21] K. Chen, Z.-H. Wang, M. Lin, and M. Yu, "An improved DV-Hop localization algorithm for wireless sensor networks," in *Proc. IET Int. Conf. Wireless Sensor Netw.*, 2010, pp. 255–259.
[22] S. Kumar and D. K. Lobiyal, "An advanced DV-Hop localization algorithm for wireless sensor networks," *Wireless Pers. Commun.*, vol. 71, no. 2, pp. 1365–1385, 2013.
[23] P. Wang, J. Huang, Z. Cui, L. Xie, and J. Chen, "A Gaussian error correction multi-objective positioning model with NSGA-II," *Concurrency Comput.: Pract. Experience*, vol. 32, no. 5, 2020, Art. no. e5464.
[24] X. Cai, P. Wang, L. Du, Z. Cui, W. Zhang, and J. Chen, "Multi-objective three-dimensional DV-Hop localization algorithm with NSGA-II," *IEEE Sensors J.*, vol. 19, no. 21, pp. 10003–10015, Nov. 2019.
[25] M. F. Al-Mistarihi, I. M. Tanash, F. S. Yaseen, and K. A. Darabkh, "Protecting source location privacy in a clustered wireless sensor networks against local eavesdroppers," *Mobile Netw. Appl.*, vol. 25, no. 1, pp. 42–54, 2020.
[26] H. Wang, G. Han, C. Zhu, S. Chan, and W. Zhang, "TCSLP: A trace cost based source location privacy protection scheme in WSNs for smart cities," *Future Gener. Comput. Syst.*, vol. 107, pp. 965–974, 2020.
[27] A. Liu, X. Liu, Z. Tang, L. T. Yang, and Z. Shao, "Preserving smart sink-location privacy with delay guaranteed routing scheme for WSNs," *ACM Trans. Embedded Comput. Syst.*, vol. 16, no. 3, pp. 1–25, 2017.
[28] X. Cao, "Serial port reader and writer for the Android gateway," 2018. [Online]. Available: https://github.com/cxfcdcpu/gateway
[29] X. Cao and S. Madria, "Efficient geospatial data collection in IoT networks for mobile edge computing," in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl.*, 2019, pp. 1–10.
[30] T. M. Chan, "Optimal output-sensitive convex hull algorithms in two and three dimensions," *Discrete Comput. Geometry*, vol. 16, no. 4, pp. 361–368, 1996.
[31] G. B. Hughes and M. Chraibi, "Calculating ellipse overlap areas," *Comput. Vis. Sci.*, vol. 15, no. 5, pp. 291–301, 2012.
[32] X. Min and G. Yuhong, "Machine learning and knowledge discovery in databases: European conference," in *Proc. Eur. Conf., ECML Part II*, 2015.
[33] TelosB Datasheet, "Crossbow Inc," 2010. [Online]. Available: www.willow.co.uk/TelosB_Datasheet.pdf
[34] MicaZ, "Crossbow Inc," 2006. [Online]. Available: www.willow.co.uk/MICAz_OEM_Edition_Datasheet.pdf
[35] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. 4th Int. Conf. Embedded Netw. Sensor Syst.*, 2006, pp. 307–320.
[36] H. Tseng, P.-C. Chang, G. Andrew, D. Jurafsky, and C. D. Manning, "A conditional random field word segmenter for sighan bakeoff 2005," in *Proc. 4th SIGHAN Workshop Chinese Lang. Process.*, 2005, pp. 168–171.



**Xiaofei Cao** received the BS degree in electrical engineering from Tianjin University, China, and the MS degree from the Electrical Engineering Department, Missouri University of Science and Technology, Rolla, Missouri. He is currently working toward the PhD degree from the Department of Computer Science, Missouri University of Science and Technology, Rolla, Missouri. His current research interests include energy efficient IoT networks, sensor data compression, and machine learning.

**Sanjay K Madria** (Senior Member, IEEE) is curators' distinguished professor with the Department of Computer Science, Missouri University of Science and Technology, Rolla, Missouri. He has published more than 275 Journal and conference papers in the areas of mobile and sensor computing, cloud and cyber security. He has won five IEEE best papers awards in conferences such as IEEE MDM and IEEE SRDS. He has been awarded JSPS (Japanese Society for Promotion of Science) visiting scientist fellowship, and ASEE (American Society of Engineering Education) fellowships. He was also awarded NRC Fellowships by National Academies. He received the faculty excellence research award six times from his university. He is also an ACM distinguished scientist, and served as ACM and IEEE distinguished speaker. He is an IEEE Golden Core Awardee for his distinguished service.