Sensors and Materials, Vol. 31, No. 2 (2019) 365–376
MYU Tokyo

S & M 1777

Design and Implementation of an Internet-of-Things Roadside Parking System Based on Raspberry Pi 3 and Bluetooth Low Energy Mesh Sensor Network

Ing-Chau Chang,[1] Peng-Jun Hsu,[1] Yuan-Sheng Lin,[1] Yu-Sheng Wu,[1]
Pin-Lin Chen,[1] Xian-Feng Lin,[1] and Chin-En Yen[2*]

[1]Department of Computer Science and Information Engineering, National Changhua University of Education, No. 2, Shi-Da Road, Changhua City 500, Taiwan, ROC
[2]Department of Early Childhood Development and Education, Chaoyang University of Technology, 168, Jifeng E. Rd., Wufeng District, Taichung 41349, Taiwan, ROC

(Received May 25, 2018; accepted October 11, 2018)

Keywords: Internet of Things (IoT), roadside parking system, Raspberry Pi 3, Bluetooth Low Energy (BLE) mesh sensor network

In this paper, we propose an Internet of Things (IoT) parking management solution for roadside parking spaces. This solution includes software and sensor hardware installed at each parking space. Because the smartphone is prevalent nowadays, we also design a smartphone application to enable users to easily access information regarding parking spaces near the user's destination and devise the best path to it anytime, anywhere. We choose Apple iOS as the programming platform on the smartphone and Raspberry Pi 3 as the sensor hardware, which uses an infrared distance sensor to determine whether the parking space has been occupied by a vehicle or not. We also adopt Bluetooth Low Energy (BLE) technology to connect sensors of all roadside parking lots and build a BLE mesh sensor network to collect parking statuses. A routing protocol for the BLE mesh sensor network to efficiently forward parking information to the parking management server is proposed in this paper. Finally, we present the software and sensor hardware used for implementing this parking management system, the simulator to execute the proposed BLE mesh routing algorithm, and snapshots of the parking app.

1. Introduction

Currently, most parking management systems count the number of available parking spaces when a vehicle enters or exits the parking lot. Some advanced systems use cameras to detect whether or not a parking space is occupied. However, this approach is too expensive to be adopted for roadside parking spaces. Moreover, it requires wired networks to collect parking information. As listed in Table 1, we compare three parking apps and find several defects in them.[1–3] First, none can search or reserve available roadside parking spaces. Second, Tc Parking[2] and Parking Luck[3] do not record the parking history for future-usage analysis. Most

*Corresponding author: e-mail: ceyen@cyut.edu.tw
https://doi.org/10.18494/SAM.2019.2087

ISSN 0914-4935
https://myukk.org/

Table 1
Comparison of parking apps.

	Parking Lot App[1]	Tc Parking[2]	Parking Luck[3]	This System
Navigate to parking spaces	Yes	Yes	Yes	Yes
Log in via Facebook ID	No	No	No	Yes
Reserve parking space	No	No	No	Yes
Calculate parking fee	Yes	Yes	No	Yes
Record parking history	Yes	No	No	Yes
Collect parking space status through BLE sensor networks	No	No	No	Yes

important of all, these apps collect parking space statuses manually, instead of aggregating them automatically through wireless sensor networks. For better integration and higher security, we prefer to use the user ID and password of Facebook to log in to the parking app.

Recently, Bluetooth Low Energy (BLE) mesh sensor networks have been proposed to efficiently collect statuses of roadside parking spaces that are connected by BLE technology.[4] Because the conventional Bluetooth networks only support the star topology, many issues, e.g., how to discover neighboring nodes and find the best route to transmit packets to the destination node, should be solved for building the BLE mesh sensor networks. Although related work has yielded some advances,[5–9] new problems have been introduced, as will be discussed in Sect. 2, on this kind of mesh topology.

In this paper, we propose an Internet of Things (IoT) parking management solution for roadside parking spaces. Its contributions are as follows.

1. It supports the use of Facebook user ID and password to log in to this app.
2. It adopts Raspberry Pi 3 as the sensor hardware, which uses the infrared distance sensor, i.e., Sharp GP2Y0A41SK0F, to determine whether or not the parking space has been occupied by a vehicle. This information is further transferred back to the parking server using the BLE module of Pi 3.
3. Because traditional BLE techniques cannot fully support the mesh network topology at this time, we propose and implement our BLE routing algorithm for the present topology to collect statuses of roadside parking spaces in this app.
4. This app can, for example, search and reserve roadside parking spaces, navigate the vehicle to the reserved space, calculate the parking fee, and record the parking history.

The rest of the paper is organized as follows. In Sect. 2, we discuss the operation of related routing protocols on BLE mesh sensor networks. In Sect. 3, details of the proposed system architecture, BLE routing algorithm, and functions of the parking app are explained. The software and hardware used for implementing this parking management system, the simulator to execute the proposed BLE mesh routing algorithm, and snapshots of the parking app are presented in Sect. 4. Finally, in Sect. 5, we conclude this paper.

2. Related Routing Protocols on BLE Mesh Sensor Networks

Below, we briefly discuss the operations of related routing protocols on BLE networks. The first one is MultiHop Transfer Service (MHTS),[6] which is based on Bluetooth 4.0 star

topology. It includes the on-demand route discovery phase and the packet forwarding one. If the packet source does not know the route to the destination, it will begin the route discovery process by sending an advertisement message to its neighbors, which further repeat this process toward the destination until the advertisement message reaches a node that has a known path to the destination. Hence, each node on the forwarding route of the advertisement message will update its routing table to record its neighbor to the destination as its next hop. After that, the source node conveys all data together through the link layer connection between adjacent nodes until the destination has received all data. As a result, the memory size of each BLE node limits the total data length transmitted by MHTS. Furthermore, MHTS suffers from long end-to-end latencies owing to its per-hop forwarding process.

The second protocol is the BLE Mesh Network (BMN).[7] It is based on Bluetooth 4.1 and the Directed Acyclic Graph (DAG) for routing and includes the construction, maintenance, and optimization stages. If a node intends to join the BLE mesh sensor network, it begins the construction stage by transferring a DAG Information Solicitation (DIS) message to its neighbors and waiting for the DAG Information Object (DIO) messages returned from them. Thus, it can identify its parent and alternative node, which are associated through their corresponding Rank values, toward the root of DAG. Finally, every node records its parent, alternative parent, and all children in a table. Hence, the routing table of the DAG root records the path to each destination node. When a source node wants to send a packet to a destination node, it starts the maintenance stage to find a path to the destination in its routing table. If not, it repeatedly forwards this packet to its parent until the parent has found a path to the destination or the DAG root has received the packet. Hence, this packet can be forwarded to the destination. Finally, in the optimization stage, attempts to balance the distance of each node to the DAG root are made. In summary, defects such as the single-node failure and traffic congestion near the root may degrade the performance of this protocol.

The on-demand scatternet formation and routing protocol is proposed to connect several piconets as a scatternet using Bluetooth 4.1.[8] This protocol includes two stages: forming the scatternet and discovering the route. In the first stage, the master and slave nodes build a table to record its slaves and master, respectively. For connecting two piconets, the node acts as the master in one piconet but as the slave in another. After a new node creates a connection to its nearby node, it may behave as the master or the slave, depending on its function as an advertiser or a scanner during the connection formation. In the second stage, if the source node intends to send a packet to a destination node, it first transmits a route request to check whether the destination has been recorded in the slave table of its master or not. If not, the master of the source node forwards this route request to each slave found by the breadth-first search. Each slave, which has also joined another piconet, repeats this process until the destination has received the route request. In this way, the source node finds all paths to the destination and chooses the shortest one to convey the packet. Consequently, this protocol consumes many network capabilities to discover all paths exhaustively.

In our proposed BLE routing protocol for the mesh topology, we first define the metric distance of each node X with respect to its neighbor A. This value depends on the metric distance from B to X and the received signal strength indicator from A to B. Then, each node

alternates its state to scan its neighbors and to broadcast its information to them to discover the gateway (GW) node and establish the BLE mesh sensor network. Whenever a node finds a new path to the GW, this protocol recalculates the new metric distance of this node to the GW and updates its routing table if needed. Hence, it achieves shorter end-to-end latencies and consumes less network resources than the aforementioned related protocols. Details of this proposed routing protocol will be mentioned in Sect. 3.2.

3. System Design

3.1 System architecture

As shown in Fig. 1, this system has two application programming interface (API) servers. The first one provides public parking APIs such as requesting and reserving the parking space for mobile apps. The second one affords parking sensor APIs for parking space sensors to report their statuses to two databases, MongoDB[10] for storing static information of parking spaces and Remote Dictionary Server[11] (Redis) for recording their real-time statuses, using the (key:value) pair, in a hash table. To reduce the complexity of server applications and difficulties in implementing mobile apps for this system, we choose the JSON[12] format to upload information. In addition, the user can log in to this system by using his/her user ID/password or binding it to his/her Facebook account. Then, with the help of the granted token, which is carried in the HTTP header, in the log in process, the user can access data stored in the servers. The aim of this system is focused on building a BLE mesh sensor network to improve the energy efficiency of parking sensors. Through this network, the sensor of each parking space first transfers its parking status to the nearest GW, which then aggregates parking information and forwards it to the server providing the parking sensor APIs through Wi-Fi. Finally, this parking information is recorded in the aforementioned two databases.

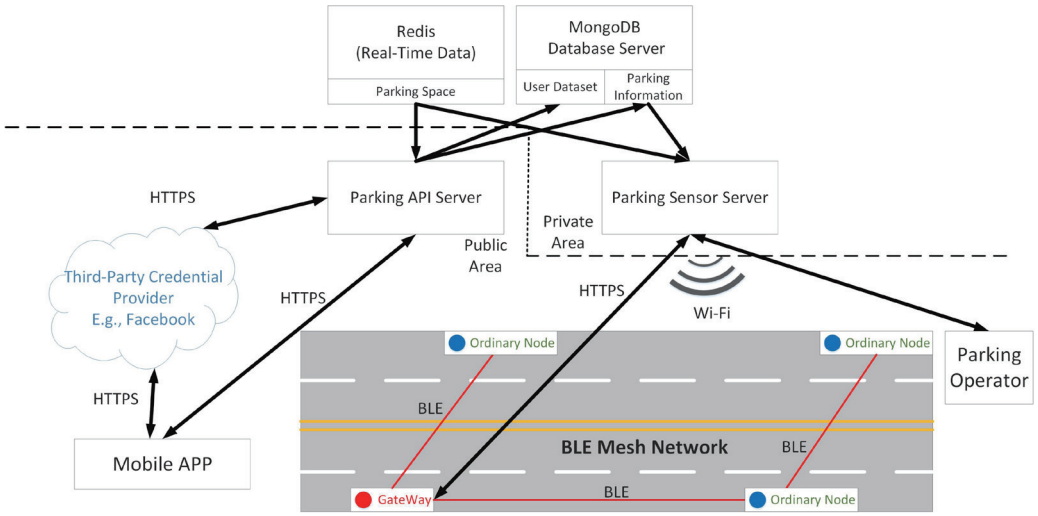

Fig. 1. (Color online) System architecture of our IoT parking management system.

3.2 BLE routing algorithm for the mesh topology

For this system, we modified approaches described in Refs. 5 to 9 to implement the routing algorithm for BLE mesh sensor networks. All sensor nodes periodically broadcast their statuses to neighbors and receive information from them. The GW node is a sensor node that collects parking information from nearby sensor nodes and forwards this information to the parking server through its built-in Wi-Fi interface. We define the metric distance of each node X as Eq. (1), where the metric distance from A to X, i.e., MD_A^X, and that from B to X, i.e., MD_B^X, are 16-bit values in the range of 0 to 65535. The received signal strength indicator from A to B, i.e., $RSSI_A^B$, is a negative value in dBm, which means that the closer the value is to 0, the stronger the received signal is.

$$MD_A^X = -0.68 \times (RSSI_A^B - 20) + MD_B^X \tag{1}$$

As soon as the sensor node X joins the BLE mesh sensor network, it uses the proposed routing algorithm, shown in Fig. 2, to find the GW.

1. If X has not found the GW, which means it does not know the route to the GW, its metric distance is set as 65535. However, that of the GW is 0.

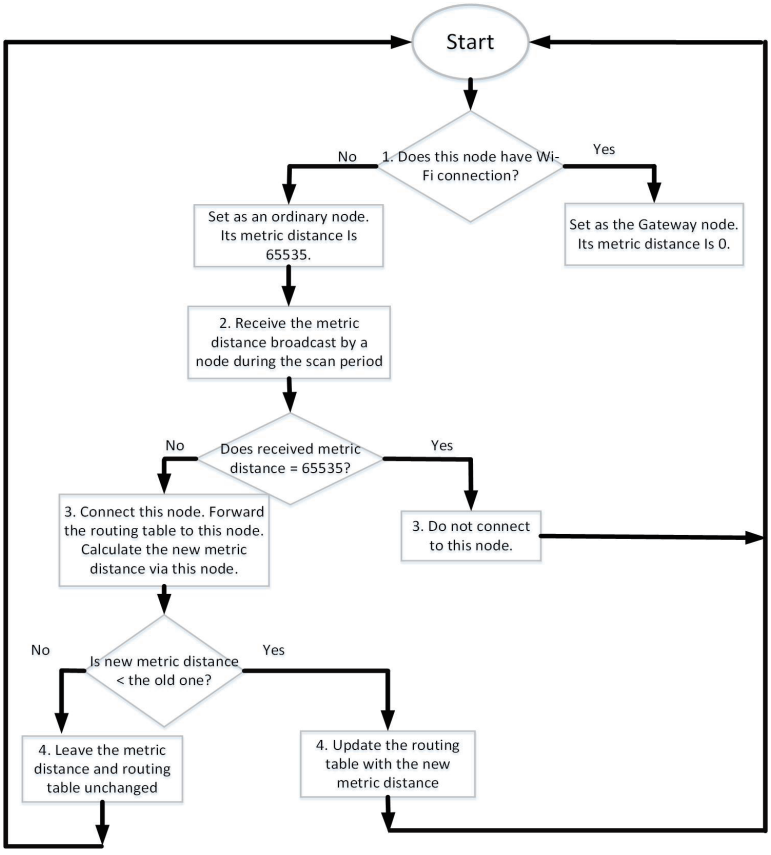

Fig. 2. Flow diagram of the proposed routing algorithm for BLE mesh sensor networks.

2. In this protocol, each node periodically alternates its state. It first broadcasts its information to neighboring nodes and then scans neighboring nodes. At this step, X starts to scan neighboring nodes and receives their broadcast metric distances. As shown in Fig. 3, the GW and nodes A, B, and C broadcast their metric distances to their neighbors.
3. If the metric distance of a neighboring node is 65535, X does not connect to this neighbor. Otherwise, these two nodes create a BLE connection between them where one node is in the scanning state and the other one is in the broadcasting state. As shown in Fig. 4, nodes A and C each establish a BLE connection to the GW. Then both of them forward its routing table to the GW. In this way, the routing table of the GW owns the most complete routing information from all nodes in its mesh network. The routing table of each sensor node records the GW address, the MAC address of the preceding node, and the total metric distance of the route to the GW.
4. After that, because node B is not within the scan range of GW, it scans node C, which already knows the route to the GW, and connects to C. It calculates and records its metric distance to the GW through node C. Furthermore, C replicates the routing information of B in its routing table. As shown in Fig. 5, node B calculates its route to the GW via C with a metric distance of 18. Finally, the GW knows the route for each sensor node to forward sensed data back to it.
5. Whenever B finds A and creates a BLE connection with it, B recalculates its new metric distance to the GW through A. If the new metric distance of B is less than the old one, B will dynamically update its routing table with this value, which means that it has discovered a better route to the GW in the BLE mesh sensor network. As shown in Fig. 6, when node B recognizes that the metric distance, i.e., 15, via A to the GW is smaller than that, i.e., 18, via C, it will update its route to A with the metric distance of 15.

3.3 Functions of the parking app

There are eight major functions in this parking app.

1. Login: the user uses his/her Facebook ID/password to log into this parking system. This app will transmit the FB token returned by Facebook to the parking server, which will further

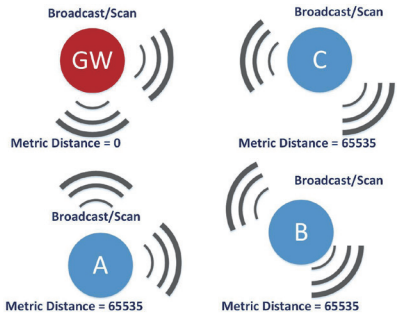

Fig. 3. (Color online) Each node alternately broadcasts its metric distance and scans its neighbors.

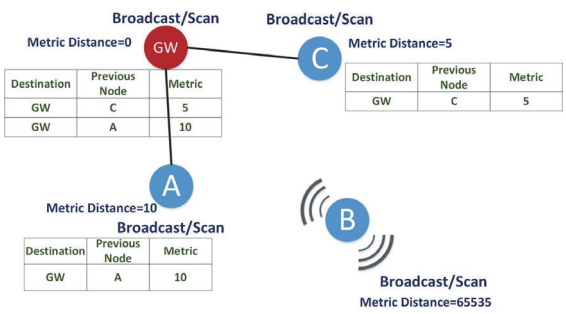

Fig. 4. (Color online) Nodes A and C establish a BLE connection and forward their routing tables to the GW.

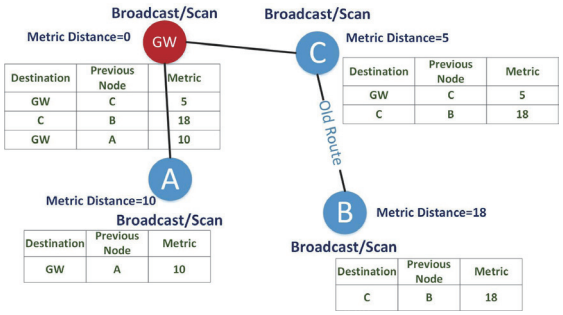

Fig. 5. (Color online) Node B calculates its route to the GW via C.

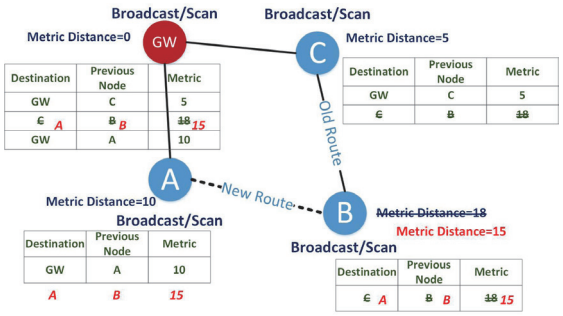

Fig. 6. (Color online) Node B updates its route to the GW via A.

verify the user credential with Facebook using this FB token. If the user has passed the credential check of Facebook, he/she will become a normal user of this system and receive a user token.

2. Map View: This function displays the electronic map and parking spaces.
3. Navigation: Apple Map navigates the user to the chosen parking space.
4. Vehicle View: This function lets the user check the vehicle location.
5. Parking Space Reservation: The user can reserve a parking space at a certain time. This app first issues a HTTP request, which contains the parking space ID, to the parking server. Then this server will command the LED of the reserved parking space to blink until a vehicle has parked in this parking space. The parking server also generates an accounting record for this reservation.
6. Accounting: Whenever the vehicle leaves the parking space, this system will calculate the parking fee automatically.
7. Parking history: The user can examine his/her parking history, including the parking space, time, and fee.
8. Add parking spaces: Before a user can add new parking spaces, he/she must first become a parking space provider by applying for the provider token from the parking server using his/her user token.

4. System Implementation

4.1 Simulations of the BLE mesh routing algorithm

Because a large-scale simulation is difficult owing to the limited number of sensor nodes, we write a C# program to evaluate the proposed BLE mesh routing algorithm. Images of this program are shown in Figs. 7 and 8. The current mesh topology, status, and routing table of each sensor node are illustrated in the left, central, and right parts of these images, respectively. As mentioned above, whenever a node finds a shorter route with a smaller metric distance to the GW, it will dynamically update its route and routing table. The complete BLE mesh topology is shown in Fig. 8.

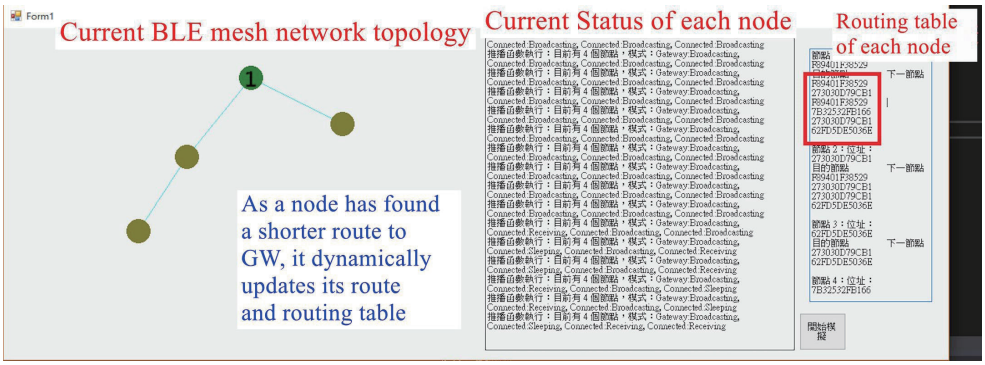

Fig. 7. (Color online) Image of sensor nodes updating their routes and routing tables.

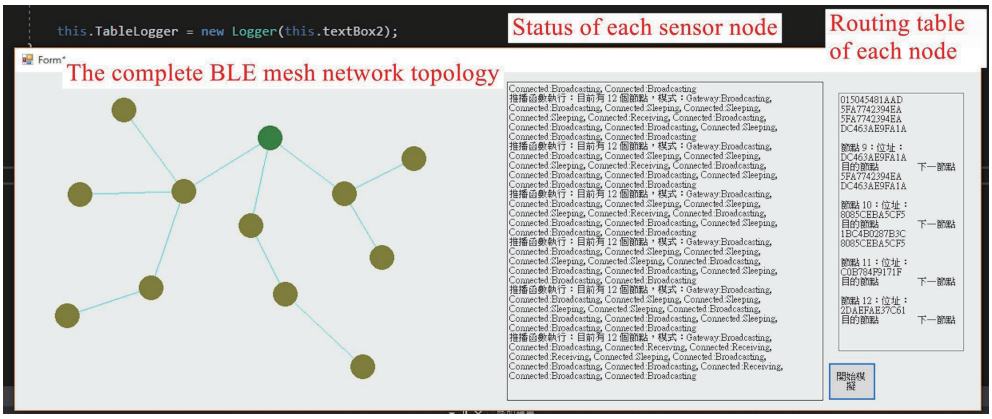

Fig. 8. (Color online) Image of the complete BLE mesh sensor network topology.

4.2 Software and hardware

Parking space hardware: Raspberry Pi 3, infrared sensor, motherboard.
Integrated development environments: VisualStudio, Xcode,[13] Sublime,[14] Docker.[15]
Software language: Swift,[16] Javascript.
Database: MongoDB, Redis.

In this system, we use Sharp GP2Y0A41SK0F as the infrared distance sensor to determine the distance between it and the object in front from the output voltage of its pin. In general, the higher the voltage it outputs, the shorter the distance that separates the sensor from the object. For example, the distances are 4, 10, and 30 cm when the output voltages are 3.1, 1.25, and 0.3 V, respectively. Raspberry Pi 3 adopts a 10-bit analog-to-digital converter (ADC), i.e., MCP3008,[17] to convert the output voltage of Sharp GP2Y0A41SK0F to a digital value of voltage (ADC) in the range of 0 to 1023. With the ADC value, we use Eq. (2) to calculate the distance in centimeters. In this way, this system can determine whether or not a vehicle has occupied the parking space.

$$Distance = 5461 / (ADC - 17) - 2 \quad (2)$$

4.3 Operations of this app

This parking system divides the Taiwan area into multiple 1.11 × 1.11 km^2 grids. Using the app, the client issues a HTTP GET message to request information on the current grid. Then the user clicks the Facebook Login button and uses his/her Facebook ID/password to log into Facebook and the parking server. After the user has passed the credential check of Facebook, he/she will become a normal user of this system and the app will show the current user location on the map. As shown in Fig. 9, the green and red pins indicate available parking and reserved or occupied spaces, respectively, in the BLE mesh sensor network. If the user wants to reserve an available parking space, he/she must click the corresponding green pin to display information on this parking space. After choosing the date of the reservation and clicking the Reserve button shown in Fig. 10, this green pin will change to red and this page will list detailed reservation data, as shown in Fig. 11. The user can cancel this reservation or start navigation to this reserved parking space by clicking the Cancel or Navigate button, respectively. The navigation page is shown in Fig. 12. The user can query his/her parking history regarding locations and parking fees of past reserved or parked spaces. Figure 13 shows a part of the BLE mesh sensor network in our implementation, where the red line represents the BLE link between the GW and the parking space. Figure 14 shows the parking space hardware, which consists of a Raspberry Pi 3, an infrared distance sensor, and a lamp post with an LED. If a vehicle has parked in this parking space, the infrared distance sensor will send a signal to Raspberry Pi

Fig. 9. (Color online) Main page of this app.

Fig. 10. (Color online) Parking space reservation page.

Fig. 11. (Color online) Details of reservation data of parking space.

Fig. 12. (Color online) Navigation page.

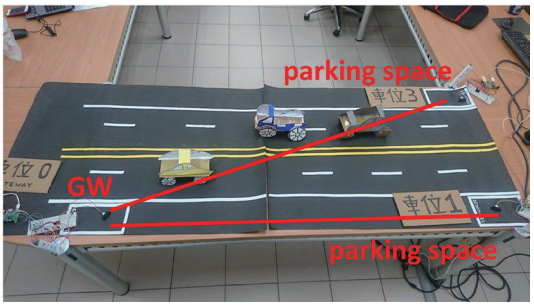

Fig. 13. (Color online) Part of the BLE mesh sensor network in our experiment.

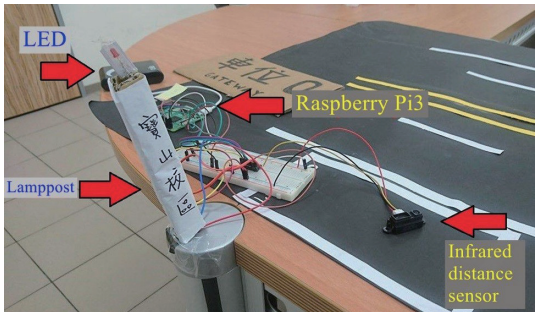

Fig. 14. (Color online) Hardware of a parking space.

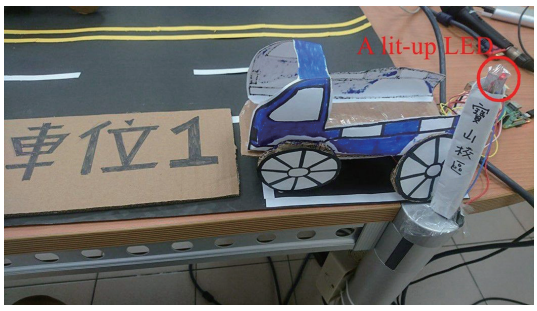

Fig. 15. (Color online) A lit LED indicates that a vehicle has parked in this parking space.

3, which further turns on the LED, as indicated in Fig. 15, and sends a message to the parking sensor server to update the status of this parking space in the proposed parking databases.

5. Conclusions

We proposed an IoT roadside parking system and a routing algorithm to build the BLE mesh sensor network. Then, we implemented a parking app, two servers to provide parking APIs and parking sensor APIs, and two databases, i.e., MongoDB and Redis, to manage public and real-time parking information. Each parking space was equipped with a Raspberry Pi 3, an infrared distance sensor, and an LED lamp post to detect and exhibit whether a vehicle has parked in it or not. In the future, we expect to develop a system using RFID or ETag to identify whether the vehicle parked in a reserved parking space is the one having reserved it. If yes, this system will begin to compute the parking fee of this vehicle. Otherwise, the vehicle driver will be notified by a warning alarm or flash.

Acknowledgments

This work was supported by the Ministry of Science and Technology (MOST), Taiwan, under Grant Number MOST 106-2221-E-018-008-.

References

1 Parking Lot App: https://play.google.com/store/apps/details?id=com.alfred.parkinglot&hl=zh_TW (accessed April 2018).
2 Tc parking: https://play.google.com/store/apps/details?id=holdingtop.TCePark&hl=zh_TW (accessed April 2018).
3 Parking Luck: https://play.google.com/store/apps/details?id=com.parkingluck&hl=zh_TW (accessed April 2018).
4 S. M. Darroudi and C. Gomez: Sensors **17** (2017) 1467. https://doi.org/10.3390/s17071467
5 M. Hatanen: Self-organizing Routing Protocol for Bluetooth Low Energy Sensor Networks: Thesis (Oulu University of Applied Sciences, 2012). http://urn.fi/URN:NBN:fi:amk-201203233719
6 K. Mikhaylov and J. Tervonen: Proc. 13th ITS Telecommunications (2013). https://doi.org/10.1109/ITST.2013.6685566
7 S. Sirur, P. Juturu, and H.P. Gupta: Proc. IEEE Sensors (2015) 1–4. https://doi.org/10.1109/ICSENS.2015.7370451
8 Z. Guo, I. G. Harris, and L.-F. Tsaur: Proc. IEEE Wireless Communications and Networking Conf. 2 (IEEE, New Orleans, LA, USA, 2015) 363–366. https://doi.org/10.1109/WCNC.2015.7127705
9 A. Gogic, A. Mujcic, S. Ibric, and N. Suljanovic: Int. J. Comput. Electr. Autom. Control Inf. Eng. **10** (2016) 1033. https://doi.org/10.1999/1307-6892/10004623
10 Mongo DB: https://mongobooster.com (accessed April 2018).
11 Redis: https://zh.wikipedia.org/wiki/Redis (accessed April 2018).
12 JSON: https://zh.ikipedia.org/wiki/JSON (accessed April 2018).
13 Xcode: https://developer.apple.com/xcode/ (accessed April 2018).
14 Sublime: https://www.sublimetext.com/ (accessed April 2018).
15 Docker: https://www.docker.com/ (accessed April 2018).
16 Swift: https://developer.apple.com/swift/ (accessed April 2018).
17 Infrared distance sensor: https://home.roboticlab.eu/en/examples/sensor/ir_distance (accessed April 2018).

About the Authors

Ing-Chau Chang received his B.S. degree in computer and information science from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1990, and M.S. and Ph.D. degrees in computer science and information engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1992 and 1997, respectively. He is currently a professor in the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, Republic of China. His current research topics include wireless networks, ad-hoc network routing protocols, and Internet of Things (IoT) systems. He is a member of the Institute of Electrical and Electronics Engineers (IEEE). (icchang@cc.ncue.edu.tw)

Peng-Jun Hsu received his B.S. degree from the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C., in 2018. His research interests include wireless networks and BLE routing protocols. (hsu.pengjun@icloud.com)

Yuan-Sheng Lin received his B.S. degree from the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C., in 2018. His research interests include wireless networks and BLE routing protocols. (sam19950810@gmail.com)

Yu-Sheng Wu received his B.S. degree from the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C., in 2018. His research interests include wireless networks and BLE routing protocols. (mouoreo@gmail.com)

Pin-Lin Chen received his B.S. degree from the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C., in 2018. His research interests include wireless networks and sensors. (bulss8516@gmail.com)

Xian-Feng Lin received his B.S. degree from the Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C., in 2018. His research interests include wireless networks and sensors. (kevenlove1029@gmail.com)

Chin-En Yen received her B.S. degree from the Department of Nutrition from Chung Shan Medical University, Taichung, Taiwan, Republic of China, in 1995 and M.S. and Ph.D. degrees from the Institute of Nutrition, Chung Shan Medical University, Taichung, Taiwan, Republic of China, in 1999 and 2009, respectively. She is currently an assistant professor in the Department of Early Childhood Development and Education, Chaoyang University of Technology, Taichung, Taiwan, Republic of China. Her current research topics include nutrition, computer-aided education, and nursing. (ceyen@cyut.edu.tw)