IEEE Access
The journal for rapid open access publishing

Received July 24, 2015, accepted August 16, 2015, date of publication September 9, 2015, date of current version September 23, 2015.

Digital Object Identifier 10.1109/ACCESS.2015.2477299

A Cloud-Based Smart-Parking System Based on Internet-of-Things Technologies

THANH NAM PHAM[1], MING-FONG TSAI[1], DUC BINH NGUYEN[1], CHYI-REN DOW[1], AND DER-JIUNN DENG[2]

[1]Department of Information Engineering and Computer Science, Feng Chia University, Taichung 407, Taiwan
[2]Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua 500, Taiwan

Corresponding author: D.-J. Deng (djdeng@cc.ncue.edu.tw)

ABSTRACT This paper introduces a novel algorithm that increases the efficiency of the current cloud-based smart-parking system and develops a network architecture based on the Internet-of-Things technology. This paper proposed a system that helps users automatically find a free parking space at the least cost based on new performance metrics to calculate the user parking cost by considering the distance and the total number of free places in each car park. This cost will be used to offer a solution of finding an available parking space upon a request by the user and a solution of suggesting a new car park if the current car park is full. The simulation results show that the algorithm helps improve the probability of successful parking and minimizes the user waiting time. We also successfully implemented the proposed system in the real world.

INDEX TERMS Smart-parking system, performance metrics.

I. INTRODUCTION

In the development of traffic management systems, an intelligent parking system was created to reduce the cost of hiring people and for optimal use of resources for car-park owners. Currently, the common method of finding a parking space is manual where the driver usually finds a space in the street through luck and experience. This process takes time and effort and may lead to the worst case of failing to find any parking space if the driver is driving in a city with high vehicle density. The alternative is to find a predefined car park with high capacity. However, this is not an optimal solution because the car park could usually be far away from the user destination. In recent years, research has used vehicle-to-vehicle [21] and vehicle-to-infrastructure [23] interaction with the support of various wireless network technologies such as radio frequency identification (RFID), Zigbee, wireless mess network [22], and the Internet. This study aimed to provide information about nearby parking spaces for the driver and to make a reservation minutes earlier using supported devices such as smartphones or tablet PCs. Furthermore, the services use the ID of each vehicle in booking a parking space. However, the current intelligent parking system does not provide an overall optimal solution in finding an available parking space, does not solve the problem of load balancing, does not provide economic benefit, and does not plan for vehicle-refusal service.

To resolve the aforementioned problems and take advantage of the significant development in technology, the Internet-of-Things technology (IoT) has created a revolution in many fields in life as well as in smart-parking system (SPS) technology [20]. The present study proposes and develops an effective cloud-based SPS solution based on the Internet of Things. Our system constructs each car park as an IoT network, and the data that include the vehicle GPS location, distance between car parking areas and number of free slots in car park areas will be transferred to the data center. The data center serves as a cloud server to calculate the costs of a parking request, and these costs are frequently updated and are accessible any time by the vehicles in the network. The SPS is based on several innovative technologies and can automatically monitor and manage car parks. Furthermore, in the proposed system, each car park can function independently as a traditional car park. This research also implements a system prototype with wireless access in an open-source physical computing platform based on Arduino with RFID technology using a smartphone that provides the communication and user interface for both the control system and the vehicles to verify the feasibility of the proposed system.

A. RELATED WORKS

In some studies [1]–[3], the authors proposed a new algorithm for treatment planning in real-time parking. First, they used an algorithm to schedule the online problem of a parking system into an offline problem. Second, they set up a mathematical model describing the offline problem as a linear problem. Third, they designed an algorithm to solve this linear problem. Finally, they evaluated the proposed algorithm using experimental simulations of the system. The experimental results indicated timely and efficient performance. However, these papers do not mention the resource reservation mechanism (all parking requirements are derived immediately and are placed in the queue), the mechanism for assessing the resources system, the mechanism to guide vehicles to the parking space, the mechanism for handling situations when the request for service is denied and do not calculate the average waiting time and average total time that each vehicle spends on the system.

In another study [4], the authors propose an SPS based on the integration of UHF frequency, RFID and IEEE 802.15.4 Wireless Sensor Network technologies. This system can collect information about the state of occupancy of the car parks, and can direct drivers to the nearest vacant parking spot by using a software application. However, in this work, the authors have no mathematical equations for the system architecture and do not create a large-scale parking system. The results of this paper only implement the proposed architecture; they do not mention the performance of the parking system. Hsu *et al.* [5] proposed an innovative system including the parking guidance service. A parking space can be reserved by a smartphone via Internet access. Upon entering the car park, the reserved parking space will be displayed on a small map using wireless transmission for vehicles under the dedicated short-range communication protocol DSRC. An inertial navigation system (INS) is implemented to guide the vehicle to the reserved space. The system will periodically update the status of the parking space in real time to help ensure system accuracy. System performance is measured through the accuracy of the inertial navigation systems run in an indoor environment, and the system implementation is evaluated by considering the accuracy of the GPS. In this paper, the authors have not evaluated the performance of the parking services, they do not provide any mathematical model of the system, and do not consider the waiting time of each vehicle for service.

Other researchers have designed architecture for parking management in smart cities [6]. They proposed intelligent parking assistant (IPA) architecture aimed at overcoming current public parking management solutions. This architecture provides drivers with information about on-street parking stall availability and allow drivers to reserve the most convenient parking stall at their destination before their departure. They use RFID technology in this system. When a car parks or leaves the IPA parking spot, the RFID reader and the magnetic loop detect the action and send this information to the unit controller to update the information on the car park status. This study uses only some simple mathematical equations for the system architecture and does not create a large-scale parking system. In other works, authors have designed and implemented an SPS [7] to solve the parking problem. A part of this system is implemented in the Zigbee network which sends urgent information to a PC through a coordinator and then updates the database. The application layer can quickly pass the parking information over the Internet, and use the advantages of a web service to gather all the scattered parking information for the convenience of those who want to find a parking space. This paper simply reports the design and implementation of an SPS and does not evaluate the system performance.

Bonde *et al.* [24] aimed to automate the car and the car parking. The paper discusses a project which presents a miniature model of an automated car parking system that can regulate and manage the number of cars that can be parked in a given area at any given time based on the availability of parking spaces. The automated parking method allows the parking and exiting of cars using sensing devices. Entry to or exit from the car park is commanded by an Android based application. The difference between the Bonde system and the other existing systems is that the authors were aiming to make the system as little human dependent as possible by automating the cars as well as the entire car park; on the other hand, most existing systems require human intervention (the car owner or other) to park the car. Lambrinos and Dosis [19] described a new SPS architecture based on the Internet of Things technology. The architecture of this system consists of a Zigbee Wireless Sensor Network (WSN), an IoT middleware layer and a front-end layer as the final user interface that provides data reporting to the user. However, there are disadvantage as it does not use a suitable application protocol for the transfer of data from the WSN to the server, such as the constrained application protocol (CoAP), there is no mathematical model for the system operations, and there is no system performance evaluation.

B. CONTRIBUTIONS

With the aim of overcoming the disadvantages of the systems mentioned above and inspired by [1]–[4] and some relevant works [17], [18], we introduce new SPS architecture based on IoT and build a mathematical model of the system operation. First, our algorithm adopts a mechanism to search car parks at the least cost. Second, we adopt a mechanism for forwarding the vehicles to another car park if the current car park is full. We propose a network of car parks such that each park is a node in a network. Each node obtains the information from the neighboring node, thus ensuring smooth movement of vehicles at low cost and increasing the probability of finding a free parking space. Our system achieves better performance compared with other parking systems. We evaluated the performance of our system through simulation and implementation. The results of the simulation are close to our mathematical models and achieve better performance than the other systems. The proposed system reduces the number of vehicles failing to find a parking space and minimizes the costs of moving to the car park. The cost defined here is the time that the user must wait for the service, thus helping users save time and money and reducing environmental pollution. We have also successfully implemented our system in a university parking system.

C. ORGANIZATION

This paper is organized as follows: Section II describes the proposed architecture including the system and network architectures. Section III discusses the algorithms and the mathematical models of the system. Section IV presents the simulation. Section V is the implementation of the system. Section VI is our conclusion and suggestions for future work.

II. PROPOSED ARCHITECTURE

A. SYSTEM OVERVIEW

The system is derived from the idea of IoT [13], [14]. The system uses the WSN [15] consisting of RFID technology to monitor car parks. An RFID reader counts the percentage of free parking spaces in each car park. The use of RFID facilitates implementation of a large-scale system at low cost. The system provides a mechanism to prevent disputes in the car park and helps minimize wasted time in looking for a parking space. After logging into the system, the user can choose a suitable parking space. Information on the selected parking location will be confirmed to the user via notification. Then, the system updates the status of the parking space to "pending" during which time the system will not allow other users to reserve it. If after a certain period of pending time the system determines that no car is parked in that space, then it changes the status to "available." The system will update the status from the WSN node (the status of car park spaces) when a new car joins in the system. Therefore, the status of the overall parking system is always updated in real time. The system will help plot the parking time for each parking space in real time and can support the business with hourly parking charges.

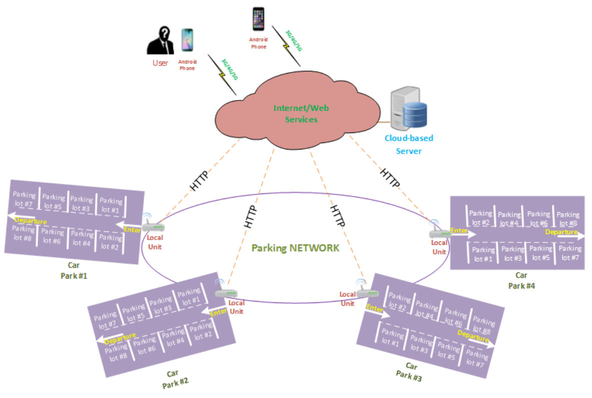


FIGURE 1. Architecture of the proposed system.

B. SYSTEM ARCHITECTURE

Fig. 1 shows our smart IoT parking system.

Elements in the system:

- *Cloud-Based Server:* This is a Web entity that stores the resource information provided by local units located at each car park. The system allows a driver to search and find information on parking spaces from each car park without the need to directly access the local server node by directly accessing the cloud-based server.

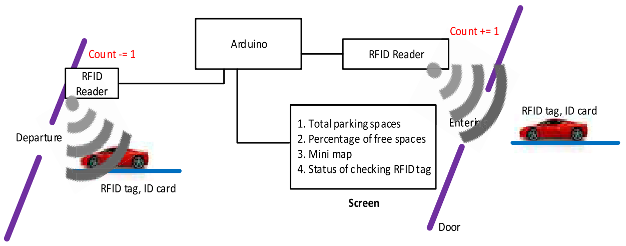


FIGURE 2. Local unit.

- *Local Unit:* This unit is located in each car park and stores the information of each parking space, as shown in Fig.2. The local unit includes the following:
  - *Control Unit:* This is an Arduino module, which is connected using an RFID reader. The card reader authenticates the user information and then displays this information on the screen. If the information of the RFID tag or card is correct, the Arduino module will control the opening of the door for the vehicle to enter. The Arduino module connects with the cloud server through an Internet connection to transfer data from the local car park to the cloud server database.
  - *Screen:* This displays information on the capacity of the local car park, the total current percentage of free spaces, the status of the RFID tag check, the user card when entering, and a mini map of the local car park.
  - *RFID Tag or ID Card:* This is used to check and authenticate user information and calculate the percentage of total free spaces in each car park.
- *Software Client:* This is an application software system. Running on Android operating system, the users will install it on their smartphones and use it to reserve parking spaces. The users access the system via 3G/4G mobile connections.

C. NETWORK ARCHITECTURE

In general, we will use the term "user" when referring to the driver or vehicle and the term "resources" when referring to the parking spaces.

1) PARKING NETWORK

We use the car park network (CPN) architecture infrastructure/backbone. The architecture is shown in Fig. 3(a), where the dashed lines indicate wireless link and the solid lines indicates wired link. This type of parking network includes routers that form as the infrastructure for connected clients. The CPN infrastructure/backbone can be built to allow sensor networks to connect using wireless radio technologies. The routers form a self-configuring and self-healing link network.

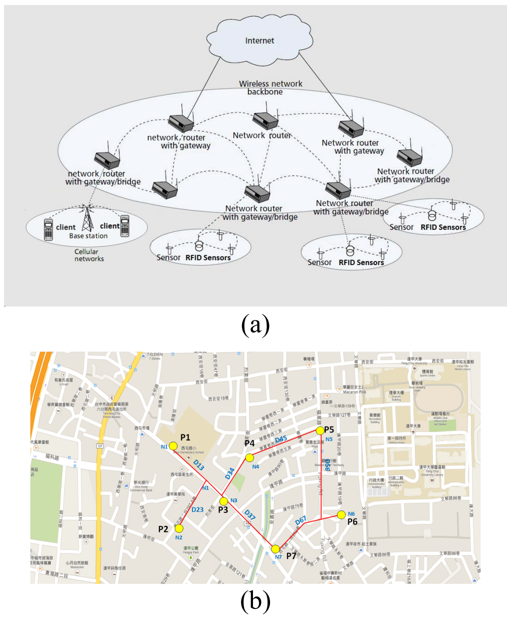

Internet
Wireless network backbone
network router with gateway
Network router
Network router with gateway
network router with gateway/bridge
Network router with gateway/bridge
Network router with gateway/bridge
Network router with gateway/bridge
client
client
Base station
Cellular networks
Sensor
RFID Sensors
(a)
P1
P2
P3
P4
P5
P6
P7
N1
N2
N3
N4
N5
N6
N7
D12
D13
D23
D34
D37
D45
D56
D67
(b)

FIGURE 3. **(a) Infrastructure/backbone of the CPN architecture. (b) CPN deployment for car parking system.**

Routers can be connected to the Internet by gateway functionality. This approach, also referred to as infrastructure meshing, provides the backbone for conventional clients and enables integration of CPNs with existing WSNs through gateway/bridge functionalities in the routers. Conventional clients with the same radio technologies as the routers can directly communicate with the routers.

We have assumed that each car park is a node in a CPN. The deployment network in a real environment is shown in Fig. 3(b) where each car park is labeled.

- P_1 is car park number 1; N_1 is the total parking spaces in P_1.
- P_2 is car park number 2, N_2 is the total parking spaces in P_2.
- P_n is car park number n, N_n is the total parking spaces in P_n.

The total capacity of the system is $N = N_1 + N_2 + N_3 + \ldots + N_n$ (spaces). D is the real distance between two nodes in the network. D_{ij} is the distance between nodes P_i and P_j. Fig. 4 shows our network.

Each node has a neighbor table to maintain information on the current status of the network and a queue with predefined length. The neighbor table for each node contains information on the neighboring nodes directly linked to it. On the other hand, the queue is used to control the number of vehicles forwarded to the node, which aims to prevent overloading in the number of vehicles beyond the capacity of the node. In our proposed system, each node will broadcast a message to its neighboring nodes after a new node joins or leaves it. This message includes information on its total free resources. The neighboring node that receives this message will update its neighbor tables. We have assumed that, in our network,

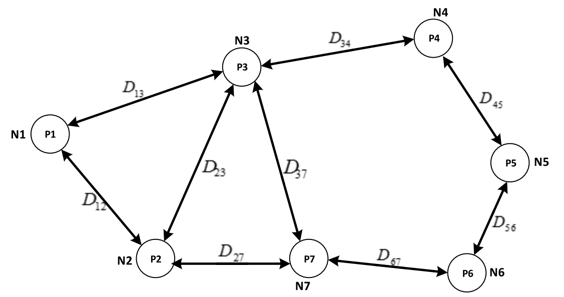
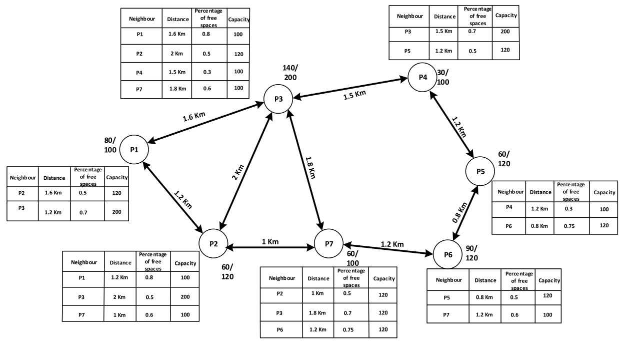


FIGURE 4. **Parking network.**

$N_1 = 100$ spaces, $N_2 = 120$ spaces, $N_3 = 200$ spaces, $N_4 = 100$ spaces, $N_5 = 120$ spaces, $N_6 = 120$ spaces, $N_7 = 100$ spaces; $D_{12} = 1.2$ km, $D_{13} = 1.6$ km, $D_{23} = 2.0$ km, $D_{27} = 1$ km, $D_{34} = 1.5$ km, $D_{37} = 1.8$ km, $D_{45} = 1.2$ km, $D_{56} = 0.8$ km and $D_{67} = 1.2$ km. These parameters are shown in Fig. 5 using simple neighbor tables. In Fig. 5, we assume that the total free spaces in $N_1 = 20$, in $N_2 = 60$, in $N_3 = 60$, in $N_4 = 70$, in $N_5 = 60$, $N_6 = 30$ and in $N_7 = 60$. To increase the performance of finding a free parking resource, the neighbor table in each node contains information on the current number of free parking resources in the neighboring nodes. Our idea is to use the number of total free parking resources in each node to calculate the cost for choosing a car park.



Neighbour	Distance	Percentage of free spaces	Capacity
P1	1.6 Km	0.8	100
P2	2 Km	0.5	120
P4	1.5 Km	0.3	100
P7	1.8 Km	0.6	100

Neighbour	Distance	Percentage of free spaces	Capacity
P3	1.5 Km	0.7	200
P5	1.2 Km	0.5	120

Neighbour	Distance	Percentage of free spaces	Capacity
P2	1.6 Km	0.5	120
P3	1.2 Km	0.7	200

Neighbour	Distance	Percentage of free spaces	Capacity
P4	1.2 Km	0.3	100
P6	0.8 Km	0.75	120

Neighbour	Distance	Percentage of free spaces	Capacity
P1	1.2 Km	0.8	100
P3	2 Km	0.5	200
P7	1 Km	0.6	100

Neighbour	Distance	Percentage of free spaces	Capacity
P2	1 Km	0.5	120
P3	1.8 Km	0.7	120
P6	1.2 Km	0.75	120

Neighbour	Distance	Percentage of free spaces	Capacity
P5	0.8 Km	0.5	120
P7	1.2 Km	0.6	100

FIGURE 5. **Simple neighbor tables.**

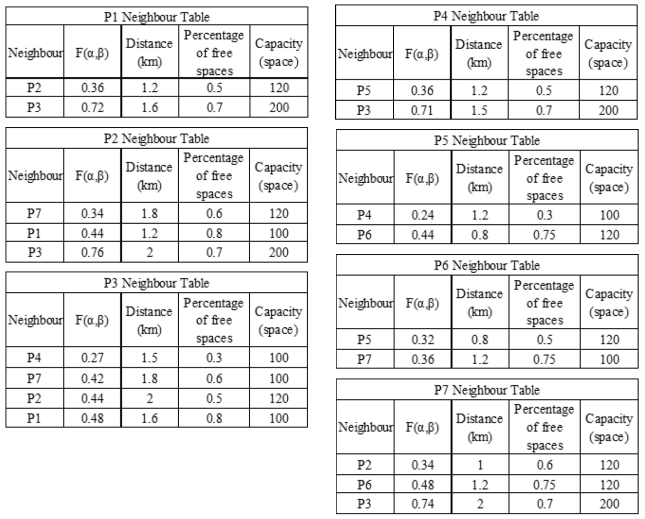
2) CONSTRUCTING THE NEIGHBOR TABLE OF NODES

We use a function named $F(\alpha, \beta)$ to calculate the cost between the nodes in the network. $F(\alpha, \beta)$ is a function that depends on the distance between two nodes and the number of free parking spaces in the destination node. $F(\alpha, \beta)$ is considered to be a weighted link between two nodes in the parking network. If two nodes are not directly linked, then $F(\alpha, \beta) = \infty$. If the vehicle comes into a node and that node is full, the vehicle will be forwarded to the next node, which is a neighbor of this node with the smallest value of $F(\alpha, \beta)$ in the neighbor table. We calculate the cost function $F(\alpha, \beta)$ from node P_i to node P_j, i.e.,

$$F_{ij} = F_{ij}(\alpha, \beta) = \alpha \times \frac{d_{ij}}{D_{up}} + \beta \times \frac{t_j}{T_{up}} \quad (1)$$

where α is a coefficient that depends on the length of the path between two nodes and β is a coefficient that depends on the number of free slots in the destination node. F(α, β) is inversely proportional to the distance between two nodes and directly proportional to the total free slots in the destination node. Depending on which parameter we consider to be the more important of the two parameters, i.e., the distance or the free slots, we can adjust α and β to achieve better network performance. α and β are parameters derived from the experiment, and their value is [0, 1]. If $\alpha = 0$, we only consider the number of free spaces to calculate the cost to the user. If $\beta = 0$, we only consider the distance between two nodes to calculate the cost to the user.

In Eq. (1), we calculate the cost function based on the distance between two nodes and the percentage of free parking spaces at each node. We use the upper bound of the distance between two nodes and the upper bound of the capacity for parking in each car park. In Eq. (1), d_{ij} is the distance between nodes P_i and P_j, D_{up} is the upper bound of the distance and is a global parameter, t_j is the number of spaces that are occupied at node P_j, and T_{up} is the upper bound of the capacity of the overall parking network and is a global parameter. We assume a network with seven nodes as in Fig. 5 and calculate the value of function F with $\alpha = 0.2$, $\beta = 0.8$, D = 2 km, T = 200 spaces: $F_{12} = 0.36$; $F_{13} = 0.72$; $F_{21} = 0.44$; $F_{23} = 0.76$; $F_{27} = 0.34$; $F_{31} = 0.48$; $F_{32} = 0.44$; $F_{34} = 0.27$; $F_{37} = 0.42$; $F_{43} = 0.71$; $F_{45} = 0.36$; $F_{54} = 0.24$; $F_{56} = 0.44$; $F_{65} = 0.32$; $F_{67} = 0.36$; $F_{72} = 0.34$; $F_{73} = 0.74$; $F_{76} = 0.48$. The neighbor table of each node with the F(α, β) function is shown in Fig. 6. Fig. 6 shows that the new neighbor table for each node follows Eq. (1). We will use this routing table in choosing the next node where to forward the user when a car park is full.



P1 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P2	0.36	1.2	0.5	120
P3	0.72	1.6	0.7	200

P2 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P7	0.34	1.8	0.6	120
P1	0.44	1.2	0.8	100
P3	0.76	2	0.7	200

P3 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P4	0.27	1.5	0.3	100
P7	0.42	1.8	0.6	100
P2	0.44	2	0.5	120
P1	0.48	1.6	0.8	100

P4 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P5	0.36	1.2	0.5	120
P3	0.71	1.5	0.7	200

P5 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P4	0.24	1.2	0.3	100
P6	0.44	0.8	0.75	120

P6 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P5	0.32	0.8	0.5	120
P7	0.36	1.2	0.75	100

P7 Neighbour Table				
Neighbour	F(α,β)	Distance (km)	Percentage of free spaces	Capacity (space)
P2	0.34	1	0.6	120
P6	0.48	1.2	0.75	120
P3	0.74	2	0.7	200

FIGURE 6. **Neighbor tables sorted by descending values of F(α, β).**

III. ALGORITHM AND MATHEMATICAL MODEL

A. ALGORITHM

We propose an algorithm to describe the operation of the system.

1) SYSTEM OPERATIONS

When a user wants to find a parking slot, he must login to our system. After successful login, a request message is sent to search for a free parking slot. Then, the system will send back a response message containing the information, including the car park address and the directions to reach it. The choice of the car park is based on the function F(α, β), which is calculated based on the current location of the vehicle and the location of the car park. The system will forward the vehicle to a car park with a minimum F(α, β) value if the current car park is full. When the user arrives at the car park, he must be authorized to enter. This authorization is achieved via the RFID technology or by scanning the user card. This mechanism is simple but economical. If the information is correct, the user is allowed to park. If the current car park is full, the system will send a suggestion message that includes information on a new car park, including the address and new directions, with a minimum cost. The new car park will be selected based on the neighbor table of the current car park (the first node in the neighbor table), as shown in Fig. 7.

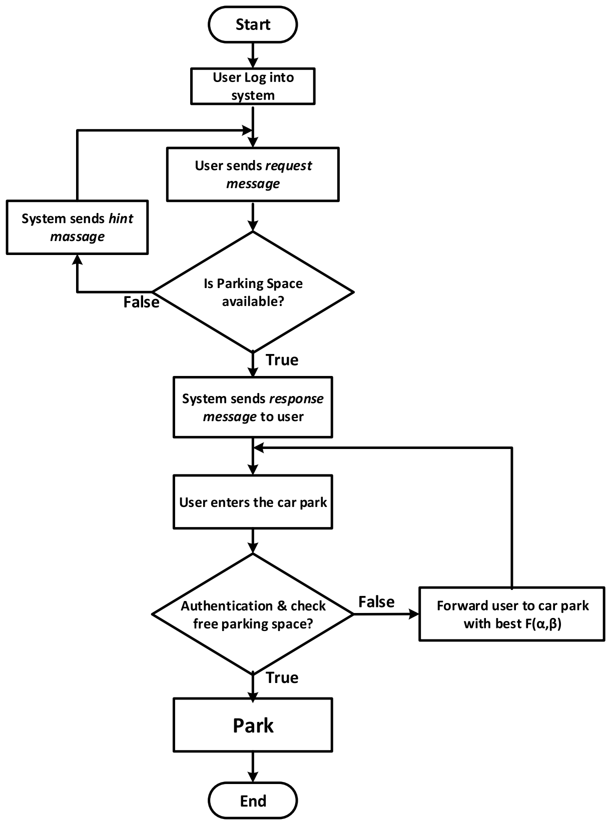

Start
User Log into system
User sends *request message*
System sends *hint massage*
Is Parking Space available?
False
True
System sends *response message* to user
User enters the car park
Authentication & check free parking space?
False
Forward user to car park with best F(α,β)
True
Park
End

FIGURE 7. **Algorithm of the system operations.**

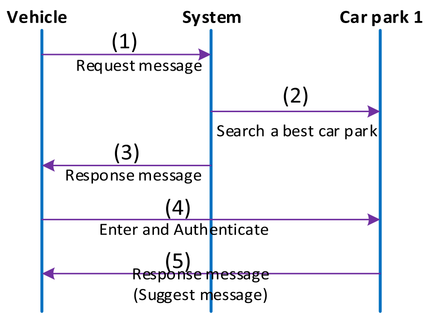


FIGURE 8. **Vehicle processes.**

Our proposed system involves two processes: reservation and entering.

- *Reservation Process:* Starting from (1) to (3) shown in Fig. 8, if the user is looking for a free parking space, he will send a request message to the system (1), which is done using a smartphone. When the system receives this request, it will find car park P_1 with the least cost [minimum value of $F(\alpha, \beta)$] and forward this message to the user. In this case, the least cost is the minimum value of function $F(\alpha, \beta)$. The value of $F(\alpha, \beta)$ is calculated as the distance (between the vehicle and car parks) and the number of free spaces in each car park. If this car park has free parking slots, it will send a response message to the user (3). The response message includes the address of car park P_1 and its directions. Because we use the percentage of total free spaces in suggesting a new car park, a high probability of success exists in finding a free parking space.
- *Entering Process:* Starting from (4) going to (5), if a user enters car park $P_{1,}$ he must be authorized using an ID or an RFID card (4). If authorized, the door is opened, and the count will increase by one. The system will send a response message to the user to notify successful parking (5). If the car park is currently full, it will send a response message suggesting an alternative car park, including relevant information on new car park P_2, with the least cost.

2) CALCULATING THE TOTAL FREE PARKING SPACES AND UPDATING THE NEIGHBOR TABLE

In our proposed system, we use RFID technology to calculate the percentage of total free parking spaces in each car park. In each car park, an RFID reader is installed at the entrance. We use a variable named "Count" to calculate the total number of vehicles in the car park. Count = Count + 1 when a vehicle enters, and Count = Count −1 when a vehicle leaves. When Count = N_i, car park *i* is full. The process of updating the neighbor table is described as follows: when a change in the value in the *Counter* occurs, which changes the percentage of the total free parking spaces at this node, this node will send a message containing updated information to the cloud-based server. The cloud-based server will update the neighboring tables of its neighboring nodes, as shown in Fig. 9.

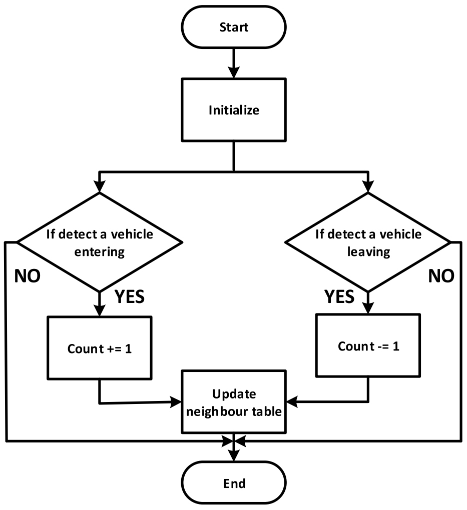


FIGURE 9. **Algorithm for updating the status of the car park.**

B. MATHEMATICAL MODELS

We build the mathematical models of our proposed system based on the results in [1] and [2]. We create a parking planning strategy. We let P denote the set of all vehicles with parking queries in the queue. We let S denote the total of all available car parks. We let W denote the set of w_{ij}, where w_{ij} is the cost between vehicle p_i ($p_i \in P$) and car park S_j ($S_j \in S$). We can achieve W by calculating the distance from the vehicle to the car park (GPS address) and the number of free spaces in car park S_j. We let M and N be the size of P and S, respectively. Therefore, the size of W is M × N. By assuming that vehicles are jobs and parking places are servers, W_{ij} is the cost for server S_j to do job P_i. We save the solution in X, where $x_{ij} \in X$, i.e.,

$$x_{ij} = \begin{cases} 1, & \text{if } P_i \text{ will park at } S_j \\ 0, & \text{if } P_i \text{ will not park at } S_j. \end{cases} \tag{2}$$

We let *C* be the total cost for all vehicles in P to go to the parking places assigned to them by the SPS, i.e.,

$$C = \sum_{i=1}^{M} \sum_{j=1}^{N} w_{ij} \times x_{ij}. \tag{3}$$

In our study, we use $F(\alpha, \beta)$ as the cost; thus, we have a new total cost.

$$C = \sum_{i=1}^{M} \sum_{j=1}^{N} F_{ij}(\alpha, \beta) \times x_{ij} \tag{3'}$$

To decrease the cost to the user, we will choose the minimum value of $F(\alpha, \beta)$ in (3′). We aim to make *C* minimum on the condition that each vehicle obtains exactly one parking resource and each car park space can be assigned to only one vehicle, i.e.,

$$\begin{cases} \sum_{j=1}^{N} x_{ij} \leq 1 \\ \sum_{i=1}^{M} x_{ij} = 1. \end{cases} \tag{4}$$

$\sum_{j=1}^{N} x_{ij} \leq 1$ indicates that any user in the queue may be assigned at most one car park but may also fail to get an assignment. On the other hand, $\sum_{i=1}^{M} x_{ij} = 1$ still guarantees that each user in the queue maintains a car park assignment. In our proposed system, if a vehicle does not find a free parking space on arrival at a full car park, forwarding it to a different car park will be suggested. We let h denote the number of forwarded vehicles. Each car park can be assigned to k_j vehicles (k_j is the total free slot in S_j; $\sum_{j=1}^{n} k_j = N$), i.e.,

$$\begin{cases} \sum_{j=1}^{N} x_{ij} \leq h \\ \sum_{i=1}^{M} x_{ij} = k_j. \end{cases} \tag{5}$$

The time complexity of our algorithm is O(n*k). We will try to reduce the time cost for each vehicle in finding a free parking resource. Our mathematical model has reduced the total cost and reaches a better solution in distributing the users to the overall network resources.

C. QUEUE MODELS

We modeled the system into a service queue. It includes all users entering each car park. The entering process at each node is considered to be a first-in first out (FIFO) queue and a Markov process, as shown in Fig. 10.

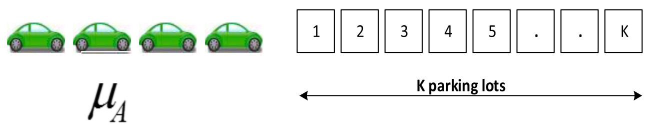


FIGURE 10. Service queue.

The mathematical model for the entering process can be described as follows: M/M/1/K/FIFO. The first "M" denotes that the distribution of the arrival process is Markovian (Poisson distribution), the second "M" stands for the service-time distribution, which is also Markovian (exponential distribution), "1" denotes the server, and K is the number of lots. μ_A is the inter-arrival time between two users, and μ_S is the service time (parking time). By assuming that $\mu_S > \mu_A$, the queue does not explode. The average waiting time in the queue (expected from a long time) in the M/M/1 queue is expressed as

$$T_a = \frac{\mu_s^2}{\mu_s - \mu_A}. \tag{6}$$

The average waiting time in the queue (expected from a long run) in the M/M/k queue is expressed as

$$T_a = \frac{\mu_s^2}{k(\mu_s - \mu_A)}. \tag{7}$$

The average waiting time of the system is

$$\overline{T_a} = \frac{\sum_{i=1}^{N} T_{a_i}}{N}, \tag{8}$$

where N is the total number of parking spaces.

IV. SIMULATION

A. SIMULATIONS

1) SETUP

To evaluate the performance of the processes, we simulated a network deployment, including the car park architecture mentioned above. We used the network simulation tool Arena to simulate this network. To simulate the mathematical and queuing models, we randomly created vehicles to join the network. The arrival process of the vehicles followed the Poisson distribution in our simulation, which was denoted as POIS(X), where X is the inter-arrival time between successive vehicles arriving at the car park. In this simulation X = 15 and 20 min. We considered the vehicle as the job and the parking space as the entity doing the job. The time for doing the job followed an exponential distribution, which was denoted as EXPO(Y) in this simulation, where Y is the average service time that a vehicle stays in the parking space. We chose Y = 60 min in this case. We simulated a parking network with five car parks as five nodes. We assumed that the network nodes are interconnected, as shown in Fig. 11. We set up each car park with a capacity for four parking spaces as the resources. We also created the same number of random vehicles to arrive at each car park. In this simulation, the number of vehicles arriving at each car park is 60, 70, 80, 90, and 100. We ran the simulations until all vehicles were serviced.

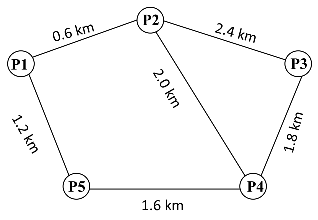


FIGURE 11. A five-node network.

To compare the network performance and provide an optimal solution for our proposed network, we set up a simulation based on various values of α and β. We simulated all cases of alpha and beta in the range from zero to one. The following are some outstanding values of α and β in the range from zero to one. The value of alpha changed from $\alpha = \{0, 0.2, 0.5, 0.8, 1\}$. The value of beta changed from $\beta = \{0, 0.2, 0.5, 0.8, 1\}$. We set up the distance between the network nodes as follows: $D_{12} = 0.6$ km, $D_{15} = 1.2$ km, $D_{23} = 2.4$ km, $D_{24} = 2.0$ km, $D_{34} = 1.8$ km, and $D_{45} = 1.6$ km. We chose the value of the upper bound of the distance as $D_{up} = 2.4$ km and that of the upper bound of the capacity as T_{up} = four spaces. All set-up parameters of the simulation are summarized in Table 1.

TABLE 1. Simulation parameters.

Parameter	Value	Unit
Number of vehicles arriving at each car park	{50, 60, 70, 80, 90, 100}	Vehicles
Inter-arrival rate	POIS(15), POIS(20)	min
Service rate	Expo(60)	Min
Coefficient of distance α	{0, 0.2, 0.5, 0.8, 1}	
Coefficient of distance β	{0, 0.2, 0.5, 0.8, 1}	

2) ARENA SIMULATION TOOL

Arena is discrete event simulation and automation software developed by Systems Modeling and acquired by Rockwell Automation in 2000 [9]–[12]. It uses the SIMAN processor and simulation language. Arena is good software that simulates many types of real-time systems such as parking systems. The basic building blocks of Arena models are modules. Flowcharts and data objects define the process to be simulated and are chosen from panels in the project bar. Flowchart modules describe the dynamic processes in the model. The types of flowchart modules available are Create, Dispose, Process, Decide, Batch, Separate, Assign, and Record. Other panels may contain many additional types of flowchart module. Data modules define the characteristics of various process elements such as entities, resources, and queues. They can also set up variables and other types of numerical values and expressions that pertain to the whole model.

In particular, in the parking system simulation, Arena supports many random distributions of arrival and service processes, such as Poisson, Normal, Exponential, Triangular, Uniform, Beta, Gamma, Logarithmic, and Weibull distributions. It allows statistical calculations and exports the average values of the parameters used to evaluate the performance, such as the average vehicle waiting time for parking requests and average time the vehicle stays in the parking system. Many previous researchers have shown that the simulation results in Arena are close to those in actual practice. Owing to the abovementioned advantages, we chose Arena as the simulation tool in this study.

3) SCENARIOS

As mentioned in Section IV-(1), we created a simulation network consisting of five nodes. To compare the performance of the algorithm that we proposed with an existing parking system, we used two different models for evaluation. In the first network model, as shown in Fig. 12(a), we implemented the greedy method. In this method, when vehicles arrive at a full car park, they will be placed in a queue and wait for the service until this car park has a free parking space. This queue is the FIFO queue. The greedy method is a common method that represents the traditional parking system with no planning to solve this problem. A loop is used to loop vehicles until the node has an available parking space.

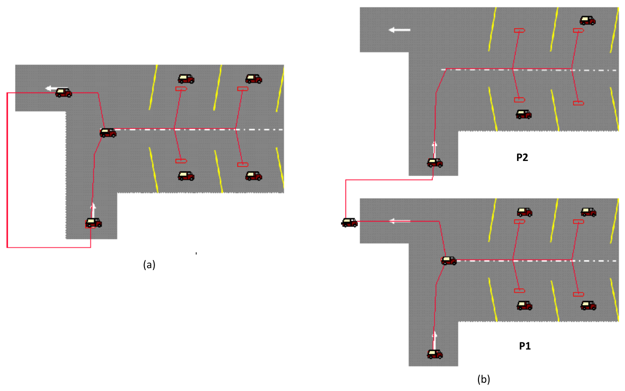


FIGURE 12. (a) Network model with the greedy method. (b) Network model with forwarding.

To reduce the waiting time of vehicles in the system, we use the second network model in planning to solve this problem. This network model is shown in Fig. 12(b). In this network model, when a vehicle arrives at a car park that is currently full, it will be forwarded to a different car park that has free parking spaces. The forwarding is based on the algorithms that we have proposed. We simulated two network models in the Arena simulator, and we compared the average waiting time and average total time that a vehicle resides in each node.

B. RESULTS AND EVALUATION

To evaluate the performance of the proposed system, we determined the parameter for system performance as the *cost* in terms of user time in the system. The *cost* to the user is the time that the user spends in the parking system for service. If this cost can be minimized, we can reduce the other costs such as monetary, fuel, and environmental pollution costs. The time in this study is the average waiting time for the service to the user and the average total time of the user in the system, including the waiting, travel, and service times. A smaller cost value leads to better system performance. Given the parameters we simulated, the parameter with the smallest time cost value will be considered as the optimal solution and is used as a proposal to deploy a similar model in practice. Fig. 13 shows our comparison of the average waiting time in a normal network with a loop and our proposed network. In the simulation, we used 50, 60, 70, 80, 90, and 100 vehicles that arrived at each node. The distribution of the inter-arrival time = POIS (15 min), which means that

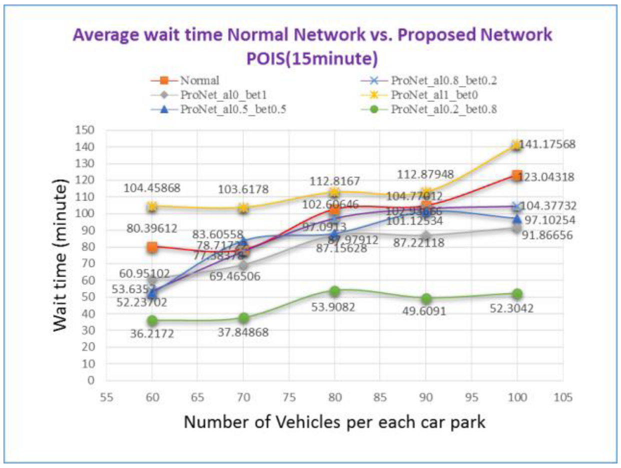


FIGURE 13. Average waiting time (15 min).

four vehicles arrive at each car park per hour. The results show that our algorithm achieves better performance than the network model without planning. We can see that if the value of α is 0.8 and the value of β is 0.2, our proposed network achieves the best performance with minimum waiting time. If the value of α is 1 and β is 0, our proposed network has the longest average waiting time, which is the worst case because we only use the distance parameter to calculate F(α, β). If the user is only forwarded to the car park with the shortest distance, a high probability exists that at the next car park, the user will still not find a free parking space because the percentage of available parking spaces is not taken into account. The network performance in this case is not equivalent to a normal network. We realize that if we use the percentage of free spaces in each car park as a parameter for planning with regard to forwarding the users, the waiting time of the user for the service will be greatly reduced compared with that in an ordinary network.

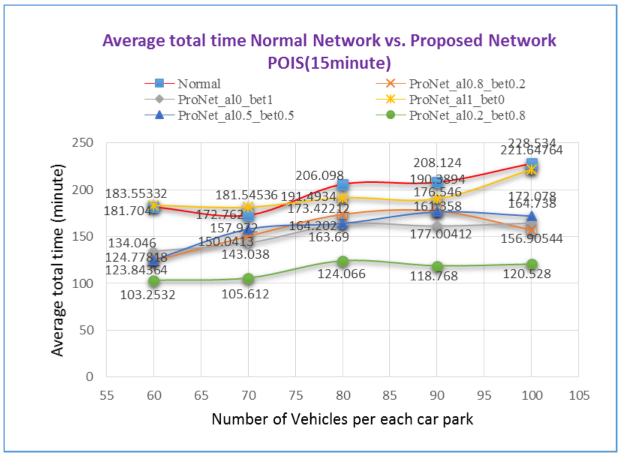


FIGURE 14. Average total time (15 min).

Fig. 14 shows our comparison of the average total time of each vehicle in a normal network model and our proposed network model. We can see that if the value of α is 0.8 and that of β is 0.2, our proposed network achieves the best performance compared with the other (α, β) pairs. If the value of α is 1 and β is 0, the average total time is approximately equal to the average total time in a normal network, which is the worst case. The best solution in this network with ($\alpha = 0.2$ and $\beta = 0.8$) reduces the average total time the user stays in the system by approximately 50%. Fig. 15 shows our simulation of 50, 60, 70, 80, 90, and 100 vehicles arriving at each node. The distribution of the inter-arrival time = POIS(20 min). The results show that our algorithm achieves better performance than the system with no parking planning. If the value of α is 0.8 and that of β is 0.2, our proposed network realizes the best performance in the range from 60 to 70 vehicles arriving at each car park. In the range from 70 to 90 vehicles arriving at each node, the pair ($\alpha = 0$ and $\beta = 1$) realizes the best performance. We can explain this result by the fact that the number of parking spaces at each network node is only four, and the arrival rate of the users decreases (20 versus 15 min); thus, the percentage of free parking-space factor is more meaningful than the distance. If the value of ($\alpha = 1$ and $\beta = 0$) is still the worst case, the system experiences the longest average waiting time.

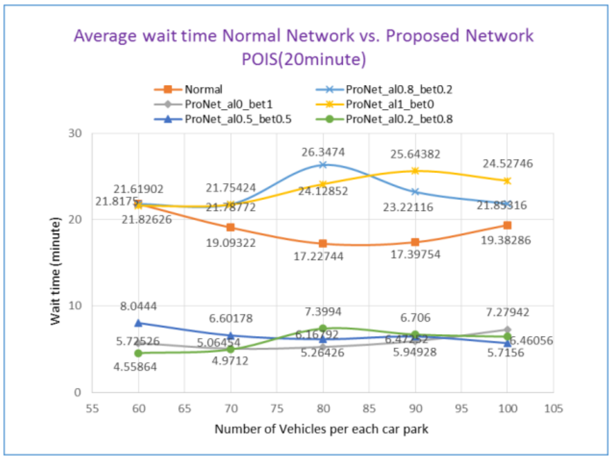


FIGURE 15. Average waiting time (20 min).

In this case, the expected inter-arrival time is 20 min, which is longer than 15 min. Thus, Fig. 15 shows that the average waiting time will be significantly reduced (approximately 10 times for the best case where $\alpha = 0.2$ and $\beta = 0.8$) compared with that shown in Fig. 13. The simple explanation is that if more vehicles come into the system per hour, the greater is the waiting time for service in terms of the total number of parking spaces in each node, which do not change. Fig. 16 shows the average total time of each vehicle in the normal network versus the proposed network in the case where the distribution of the arrival process is POIS(20 min). If the value of α is 0.8 and that of β is 0.2, our proposed network realizes the best performance (a minimum of the average total time). Based on the results of the simulation, we can conclude that if we only use the distance parameter in planning for parking, the network performance will be lower than that

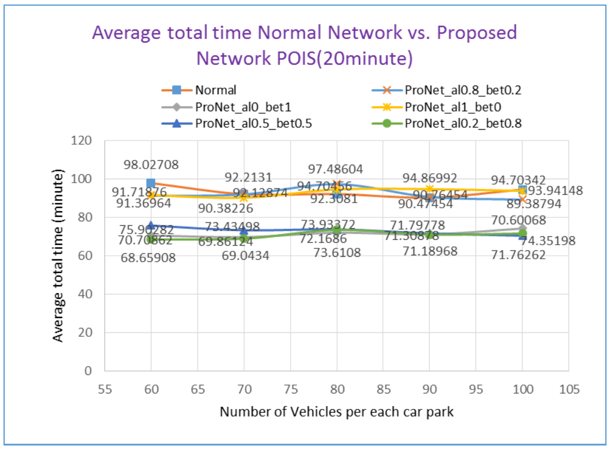


FIGURE 16. Average total time (20 min).

of the normal network. However, if we use the percentage of total free spaces in combination with the distance parameter in planning for parking, the network performance will significantly improve compared with that of the normal network. In all cases, the best network performance is achieved with the (α, β) pair of (0.2, 0.8).

V. IMPLEMENTATION

A. SOFTWARE SYSTEM

We designed a software client that runs on a smartphone based on the Android platform [8], which was built from the ground up to enable developers to create compelling mobile applications that take full advantage of all that a handset can offer. In this phase, we use the Android SDK Tools, Revision 24.3.4, which is a set of development tools used to develop applications for Android platform that can be used to write Android programs in the command prompt. The most common method is using an integrated development environment. In our ideal concept, users who want to use our system must be registered as a member of the system. Our cloud-based server is implemented on Apache Hadoop 2.7.1, and we use Apache HBase as our database. Apache HBase is a Hadoop database, a distributed, scalable, and large data store.

Fig. 17(a) shows the login interface of the system. Fig. 17(b) is a map describing the distribution of all car parks at a university. In this map, there are seven car parks with capacities of: $P_1 = 75$ spaces, $P_2 = 30$, $P_3 = 56$, $P_4 = 60$, $P_5 = 80$, $P_6 = 50$, $P_7 = 85$. Symbol S is the current location of the user. Fig. 17(c) indicates the result returned by the system when users search for a car park following the shortest distance. The result returned is P_5 and the distance from the user to the car park is 42 m. Fig. 17(d) indicates the result returned by the system when users search for a car park following our algorithms, here called the best case. The result returned is car park P_2, the distance from the user to the car park is 176 m.

B. ELEMENTS

Fig. 18 describes the implementation of the system elements, including RFID tags, the RFID reader, the

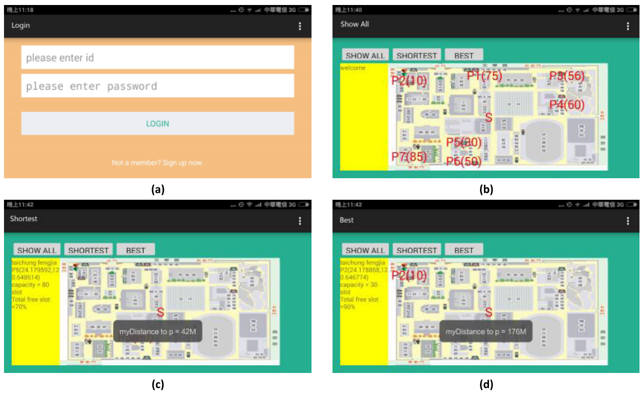


FIGURE 17. Implementation of the software system. (a) The login interface; (b) the map of all distributed car parks; (c) the result of the shortest case; (d) the result of our algorithm.

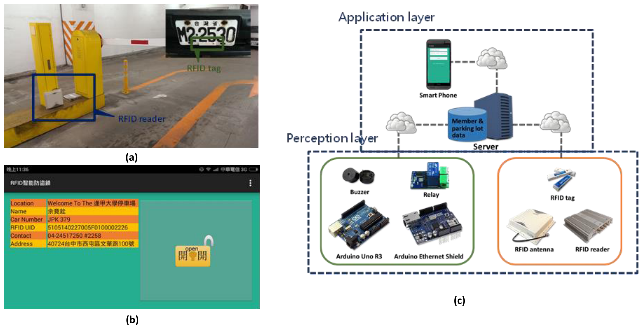


FIGURE 18. Implementation of elements in the system. (a) Implement of local unit; (b) the screen display information of RFID tags; (c) the implementation of the cloud-based server.

RFID antenna, Arduino Uno R3, Arduino Ethernet Shield, Screen and Cloud-based Server system. Fig. 18(a) shows the implementation of the local unit; Fig. 18(b) describes the screen display information of the RFID tags; Fig. 18(c) describes the implementation of the system cloud-based server.

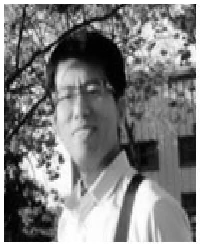
VI. CONCLUSION

This study has proposed a parking system that improves performance by reducing the number of users that fail to find a parking space and minimizes the costs of moving to the parking space. Our proposed architecture and system has been successfully simulated and implemented in a real situation. The results show that our algorithm significantly reduces the average waiting time of users for parking. Our results closely agree with those of our proposed mathematical models. The simulation of our system achieved the optimal solution when most of the vehicles successfully found a free parking space. The average waiting time of each car park for service becomes minimal, and the total time of each vehicle in each car park is reduced. In our future study, we will consider the security aspects of our system as well as implement our proposed system in large scales in the real world.

REFERENCES

[1] Y. Geng and C. G. Cassandras, "A new 'smart parking' system based on optimal resource allocation and reservations," in *Proc. 14th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2011, pp. 979–984.
[2] Y. Geng and C. G. Cassandras, "New 'smart parking' system based on resource allocation and reservations," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1129–1139, Sep. 2013.
[3] X. Zhao, K. Zhao, and F. Hai, "An algorithm of parking planning for smart parking system," in *Proc. 11th World Congr. Intell. Control Autom. (WCICA)*, 2014, pp. 4965–4969.
[4] L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and R. Vergallo, "Integration of RFID and WSN technologies in a smart parking system," in *Proc. 22nd Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, 2014, pp. 104–110.
[5] C. W. Hsu, M. H. Shih, H. Y. Huang, Y. C. Shiue, and S. C. Huang, "Verification of smart guiding system to search for parking space via DSRC communication," in *Proc. 12th Int. Conf. ITS Telecommun. (ITST)*, 2012, pp. 77–81.
[6] R. E. Barone, T. Giuffrè, S. M. Siniscalchi, M. A. Morgano, and G. Tesoriere, "Architecture for parking management in smart cities," *IET Intell. Transp. Syst.*, vol. 8, no. 5, pp. 445–452, 2014.
[7] C. Shiyao, W. Ming, L. Chen, and R. Na, "The research and implement of the intelligent parking reservation management system based on ZigBee technology," in *Proc. 6th Int. Conf. Meas. Technol. Mechatronics Autom. (ICMTMA)*, 2014, pp. 741–744.
[8] D. J. Bonde, R. S. Shende, K. S. Gaikwad, A. S. Kedari, and A. U. Bhokre, "Automated car parking system commanded by Android application," in *Proc. Int. Conf. Comput. Commun. Inform. (ICCCI)*, 2014, pp. 1–4.
[9] J. E. Hammann and N. A. Markovitch, "Introduction to Arena [simulation software]," in *Proc. Winter Simulation Conf.*, 1995, pp. 519–523.
[10] W. D. Kelton, R. Sadowski, and N. Zupick, *Simulation With Arena*, 6th ed. New York, NY, USA: McGraw-Hill, 2014.
[11] T. Altiok and B. Melamed, *Simulation Modeling and Analysis With ARENA*. Amsterdam, The Netherlands: Elsevier, 2007.
[12] M. D. Rossetti, *Simulation Modeling With Arena*. New York, NY, USA: Wiley, 2010.
[13] J. Höller, V. Tsiatsis, C. Mulligan, S. Karnouskos, S. Avesand, and D. Boyle, *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence*. Amsterdam, The Netherlands: Elsevier, 2014.
[14] I. Wigmore, *Internet of Things (IoT)*. Newton, MA, USA: TechTarget, Jun. 2014.
[15] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Netw.*, vol. 2, no. 4, pp. 351–367, Oct. 2004.
[16] K. Ashokkumar, R. B. Sam, and B. Arshadprabhu, "Cloud based intelligent transport system," in *Proc. 2nd Int. Symp. Big Data Cloud Comput. (ISBCC)*, vol. 50. 2015, pp. 58–63.
[17] Z. Suryady, G. R. Sinniah, S. Haseeb, M. T. Siddique, and M. F. M. Ezani, "Rapid development of smart parking system with cloud-based platforms," in *Proc. 5th Int. Conf. Inf. Commun. Technol. Muslim World (ICT4M)*, 2014, pp. 1–6.
[18] Z. Ji, I. Ganchev, M. O'Droma, and X. Zhang, "A cloud-based intelligent car parking services for smart cities," in *Proc. 31st URSI General Assembly Sci. Symp. (URSI GASS)*, Aug. 2014, pp. 1–4.
[19] L. Lambrinos and L. Dosis, "DisAssist: An Internet of Things and mobile communications platform for disabled parking space management," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 2810–2815.
[20] M. Du, J. Fang, and H. Cao, "A new solution for city parking guiding based on Internet of Things and multi-level multi-agent," in *Proc. Int. Conf. Electron., Commun. Control (ICECC)*, 2011, pp. 4093–4096.
[21] C. Rhodes, W. Blewitt, C. Sharp, G. Ushaw, and G. Morgan, "Smart routing: A novel application of collaborative path-finding to smart parking systems," in *Proc. IEEE 16th Conf. Bus. Infom.*, Jul. 2014, pp. 119–126.
[22] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Comput. Netw.*, vol. 47, no. 4, pp. 445–487, Mar. 2005.
[23] N. Mejri, M. Ayari, R. Langar, F. Kamoun, G. Pujolle, and L. Saidane, "Cooperation versus competition towards an efficient parking assignment solution," in *Proc. IEEE Int. Conf. Commun.*, Sydney, NSW, Australia, Jun. 2014, pp. 2915–2920.
[24] D. J. Bonde, R. S. Shende, K. S. Gaikwad, A. S. Kedari, and A. U. Bhokre, "Automated car parking system commanded by Android application," in *Proc. Int. Conf. Comput. Commun. Inform. (ICCCI)*, Coimbatore, India, Jan. 2014, pp. 1–4.

THANH-NAM PHAM received the B.S. and M.S. degrees in electrical engineering from the Hanoi University of Science and Technology, Vietnam, in 2010 and 2012, respectively. He is currently pursuing the Ph.D. degree in information engineering with Feng Chia University. His research interests include peer-to-peer networks, wireless sensor networks, and protocols for Internet of Things.



MING-FONG TSAI received the Ph.D. degree from the Department of Electrical Engineering, Institute of Computer and Communication Engineering, National Cheng Kung University, Taiwan, in 2011. He is currently an Assistant Professor with the Department of Information Engineering and Computer Science, Feng Chia University, Taiwan. His current research interests include error-control coding, multimedia communications, and vehicular communications.

DUC-BINH NGUYEN received the B.S. degree in information technology from the Thai Nguyen University of Information and Communication Technology, Vietnam, in 2008, and the master's degree in information technology from the Manuel S. Enverga University Foundation, Philippines, in 2010. He is currently pursuing the Ph.D. degree with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan. He was a Lecturer and the Manager of Network and Communication Department with the Thai Nguyen University of Information and Communication Technology, Thai Nguyen, Vietnam. His current research interests include mobile computing, vehicle ad-hoc network, wireless ad-hoc networks, and Internet of Things.

CHYI-REN DOW was born in 1962. He received the B.S. and M.S. degrees in information engineering from National Chiao Tung University, Taiwan, in 1984 and 1988, respectively, and the M.S. and Ph.D. degrees in computer science from the University of Pittsburgh, PA, in 1992 and 1994, respectively. He is currently a Professor with the Department of Information Engineering and Computer Science, Feng Chia University, Taiwan. His research interests include mobile computing, ad-hoc wireless networks, agent techniques, fault tolerance, and learning technology.

DER-JIUNN DENG received the Ph.D. degree in electrical engineering from National Taiwan University, in 2005. He joined the Department of Computer Science and Information Engineering, National Changhua University of Education, in 2005, as an Assistant Professor, and then became a Full Professor in 2012. His research interests include multimedia communication, quality-of-service, and wireless networks. In 2010, 2011, and 2012, he received the Research Excellency Award of National Changhua University of Education. In 2012, he also received the Outstanding Faculty Research Award of National Changhua University of Education. He served or serves as an Editor and a Guest Editor of several technical journals. He also served or serves on several symposium chairs and technical program committees for IEEE and other international conferences.