iParking: an Intelligent Parking System for Large Parking Lots

Dongxu Zheng, Xi Zhang, Yuanchao Shu, Chongrong Fang, Peng Cheng and Jiming Chen
State Key Laboratory of Industrial Control Technology, Zhejiang University, China
{dxzheng, xz19900623, ycshu, crfang}@zju.edu.cn, {pcheng, jmchen}@iipc.zju.edu.cn

***Abstract*—In large parking lots, it is generally difficult for drivers to know ahead of time whether there will be available parking spots and where they are. Besides, it is logistically cumbersome to wait in line for paying parking fees when leaving. In this demo, we develop an intelligent parking system called iParking. iParking could help users monitor parking states, search for a vacant parking spot, reserve it and make a self-service payment. It is low-cost providing much convenience for both drivers and parking lot management.**

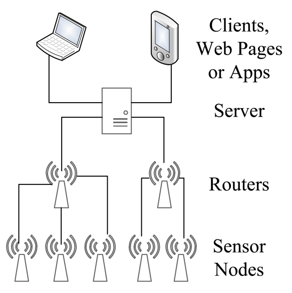
I. Introduction

Parking management has become a crucial task due to the ever-increasing use of vehicles and limited space of parking. In large parking lots, drivers find it hard to know ahead of time whether there will be available parking spots and where they are. So they often spend much time in finding an available parking space. Besides, drivers also have to spend time in waiting in line for paying parking fees.

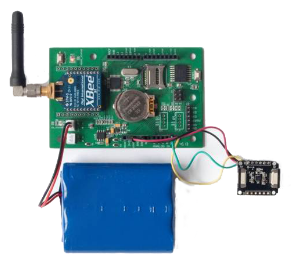
We developed a parking system called iParking, which is low-cost, and could detect the entering and leaving of vehicles automatically. The states of the parking spots could be monitored remotely via the Internet. iParking also provides other interactive features when users install the iParking Apps on their smart phones, such as searching for and reserving a vacant parking spot before parking, and make a self-service payment when leaving. The implementation shows that iParking is of high efficiency for managing large parking lots.

II. System Architecture

The system have four components: sensor nodes, routers, a server, and user interfaces (clients, web pages or Apps).



(a) Architecture Diagram



(b) Sensor Node

Fig. 1. System Architecture

Sensor nodes are deployed on each parking spot to monitor if the spot is vacant or occupied. We choose the magnetic field sensor due to its following advantages: low cost, easy deployment and little affection by environmental factors. As a vehicle consists of significant amounts of ferrous metals, it can affect the magnetic field nearby. The change of the magnetic field can be measured to estimate the state of a parking spot. Then the node sends the state to the server directly or through routers. As sensor nodes may be deployed far away from the power supply, lithium batteries and corresponding power management strategies are necessary for prolonging the node's lifetime.

Routers are deployed to forward the data from the sensor nodes or other routers. A router is designed in a similar way as a sensor node, only with the removed magnetic sensor module. Routers can be powered by lithium batteries as well as solar panels to guarantee the long lifetime of the whole system.

The server is designed to collect all the data sent by the sensor nodes and forwarded by the router nodes and to store the data in the database.

Users are allowed to read the states of the whole parking lot through clients, web pages or Apps. In addition, when one installs the iParking App on his smart phone, he can also search for the vacant parking spots nearby, reserve one of them before parking, and make a self-service payment when leaving.

III. Detection algorithm

To obtain the state of a parking spot, we develop an algorithm based on the Adaptive Threshold Detection Algorithm (ATDA) [1] with several steps shown as follows.

First, data are collected from the magnetic sensors and in order to reduce noises, a mean filter is adapted. As the magnetic field intensity can be characterized by a vector, we get three values from the three axes: $M(X)$, $M(Y)$, $M(Z)$. Then, as the magnetic sensor is easily affected by the temperature, calibration is necessary. We use the quadratic fit to calibrate readings affected by temperature.

We choose the values of three axes initially as the baseline $B(X)$, $B(Y)$, $B(Z)$, when the parking spot is vacant. Then we measure the variation ΔX, ΔY, ΔZ of each axis between the current value and the baseline. Next, if the variation of the magnetic field intensity is larger than the given threshold, we set the state $S(i) = 1$, otherwise $S(i) = 0$. As the exact position of the parking car is uncertain, both the magnitude and the direction of the magnetic field may be changed. So it is necessary to take the values of all the three axes into consideration. To this end, we define $S(i) = a \times \Delta X + b \times \Delta Y + c \times \Delta Z$, where a, b, c are constants. In our implementation we choose $a = b = 0.2$ and $c = 0.6$, for the Z axis is most sensitive to the change of the entering of a vehicle.

In order to filter out the disturbance from the environmental factors such as temperature, or the effects from the parking spots nearby, we establish a buffer before the state changes from vacant to occupied or the other way around. There are six states shown in Fig. 2. When 30 minutes have passed, or the state changes from the occupied high buffer state to the occupied state, or from the vacant buffer state to the vacant state, the sensor node sends its own state to the server.

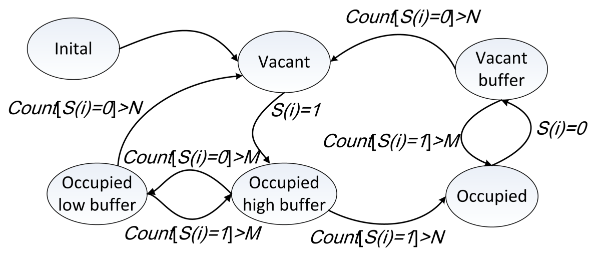


Fig. 2. ADTA Algorithmic Process

Initial state: a node goes to the initial state when it is powered on. When the node gets enough data for the mean filter, it goes to the vacant state.

Vacant state: this state indicates that the parking spot is vacant. When $S(i) = 1$, it goes to the occupied high buffer state.

Occupied high buffer state: this state indicates that the parking spot may be occupied when the variation is high enough. But we should wait for more data to ensure the occupation and to avoid the disturbance. If we get $S(i) = 1$ for N times, the node goes to the occupied state, otherwise if we get $S(i) = 0$ for M times, it goes to the occupied low buffer state. In our implementation we choose $N = 10$ and $M = 5$.

Occupied low buffer state: in this state, if we get $S(i) = 0$ for N times, the node goes to the vacant state, otherwise if we get $S(i) = 1$ for M times, it goes to the occupied high buffer state.

Occupied state: this state indicates that the parking spot is occupied. When $S(i) = 0$, it goes to the vacant buffer state.

Vacant buffer state: this state indicates that the parking spot may be vacant when the variation is low enough. To avoid the disturbance, if we get $S(i) = 0$ for N times, the node goes to the vacant state, otherwise if we get $S(i) = 1$ for M times, it goes to the occupied state.

Considering time-varying nature of the magnetic field intensity over time, an extra step is needed for calibration. Each node has to calibrate the baseline with the value of the measurement in the following way when it is in the vacant state: $B(i) = \alpha \times B(i-1) + (1-\alpha) \times M(i)$, where α is a constant. In our implementation we choose $\alpha = 0.05$.

Last, in order to prolong the node's lifetime, we choose low cost microprocessors and sensors for the implementation. And a dynamic sampling method is applied to save the energy further [2]. When a node is in the vacant or occupied state, a lower sampling rate of 0.1Hz is adopted. And the node goes to the sleep mode at other times to save energy. While if the node is in other states, the state may change at the next moment when a higher rate of 10Hz is adopted to ensure the accuracy.

IV. Implementation

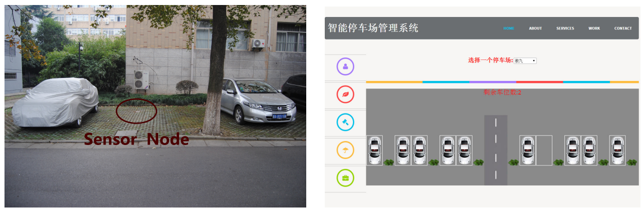
In order to implement our iParking system, we choose the Honeywell 3-axis digital compass IC HMC5883L, which is accurate and low energy cost, as the magnetic field sensor. Besides, we design our sensor node board based on the open source platform Arduino, which provides enough libraries and examples to improve the efficiency of our development. Moreover, we choose Atmega328P as the microprocessor to strike a balance between the performance and the power consumption.

We adapt the Zigbee protocol, based on the IEEE 802.15.4 protocol, for communication. All the nodes establish their routes by the AODV (Ad-hoc On-Demand Distance Vector Routing) protocol, which is low energy cost and robust enough for the application. Moreover, Zigbee protocol allows all sensor nodes to use the cycle sleep mode to save energy.

We also choose Meshlium designed by the company of Libelium as the server, since it embed a Linux system to assist developing, and offers a firm design for the outdoor application.

Last, we develop a client on Windows platform, a website, and an app on Android platform, to implement abundant interactive features of the iParking.

We have implemented iParking in Yuquan Campus at Zhejiang University, where sensor nodes are deployed in different areas. The automatic system brings a lot of convenience to the teachers, students, and other users.



(a) Node Deployment (b) Interface on the Web Page

Fig. 3. System Implementation

V. Conclusion

In this demo, we develop iParking, an intelligent parking system for large parking lots. iParking provides abundant functions which are helpful for large parking lots. iParking is low-cost, easy to deploy and to use.

References

[1] S.-Y. Cheung and P. P. Varaiya. *Traffic surveillance by wireless sensor networks: Final report*. California PATH Program, Institute of Transportation Studies, University of California at Berkeley, 2007.

[2] Z. Zhang, X. Li, H. Yuan, and F. Yu. A street parking system using wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2013, 2013.