SYSTEM FOR MONITORING DRONE POSITION INSIDE BUILDINGS

Valeriu Manuel IONESCU
University of Pitesti, Romania
valeriu.ionescu@upit.ro

Keywords: interior positioning system, Raspberry Pi, drone, client-server application, IPS

Abstract: *The constant need to monitor devices has increased in recent years to the point that devices are not only tracked outside, via GPS or similar systems, but inside too, using Internal Positioning Systems. These systems are usually using Wi-Fi signal and other sensor information to determine de the smart device location. This article presents the implementation of a simple position monitor for smart devices inside buildings using Raspberry Pi as the smart device and the Wi-Fi signal strength for distance indication. Wi-Fi AP location is considered fixed for this implementation.*

1. INTRODUCTION

In the last ten years, the interest in locating the users inside buildings has increased. One of the targets of this system is to offer new commercial opportunities for users and businesses.

One possible application envisioned is to offer to the user advertisement and information based on the shops that the user is standing by for example in a commercial mall. Another is to guide the user inside a museum and offer information about different exposition halls.

Different applications can be developed for museums, train stations, airports, hospitals, office buildings and industry. There are even Software Development Kits that have been create for indoor navigation that are offered freely of commercially [1].

In Location Alliance (ILA) [2] was created by the mobile phone industry to accelerate the adoption of interior location systems and to improve this way the user experience.

Google has developed Indoor maps that extend the Google Maps application with navigation inside large buildings.

Finally, Apple has introduced starting with IOS 11 the Maps Indoor application for the same purpose as seen in Fig. 1.

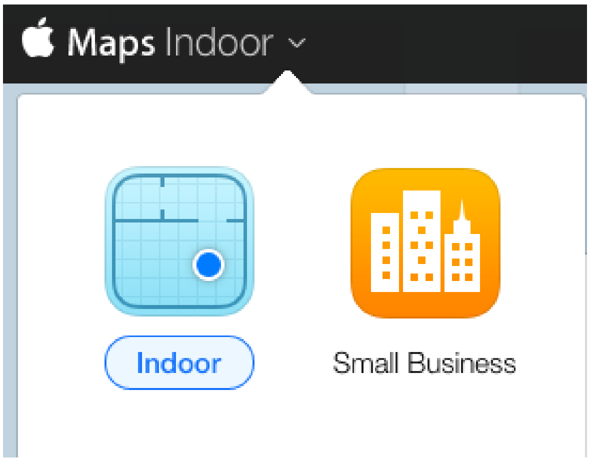

Fig. 1. Apple's Indoor maps application

Designing a positioning system based on Wi-Fi signal strength has multiple challenges compared to a global location system based on satellite position [3-5]. The first is to measure correctly the Wi-Fi strength. Some of the problems are:

- the way the signal for these devices is received (with reflections being one of the problems);
- the fact that different AP's have various signal strengths and these can be adjusted by the user;
- some areas may not have enough Wi-Fi sources to allow proper device location.

There are also some characteristics of smart devices that help in this process.

One of them is that this information can be combined with a multitude of sensors that give data on the user's location (such as accelerometer, compass or gyroscope) to improve the localization system.

Another is that for a specific public location, the number of users that move through an area is large and estimates can be made using statistics of the data from different users.

This paper focuses on recording the user Wi-Fi data coming from a Raspberry Pi 3 device. The data is used to approximate the user location based on the data coming from at least 3 Access Points. The Raspberry Pi system is mounted on a drone that communicates with a stationary server as seen in Fig. 2.

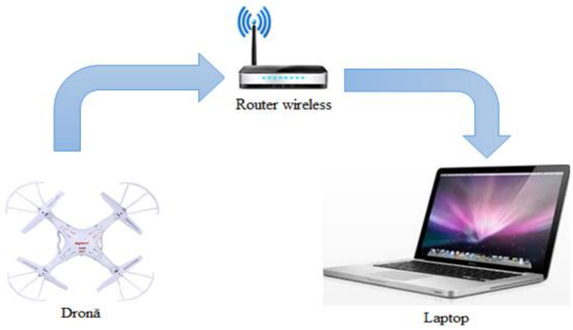

Fig. 2 Data flow between the Raspberry Pi W system and the stationary server

2. SYSTEM DESIGN

The smart system was Raspberry Pi W, which has 512MB of RAM and 1GHz frequency. A 3,7V 800mAh battery was used; therefore an adapter was required to power the 5V Raspberry Pi as seen in Fig. 2.

The PCB design was made in Proteus application.

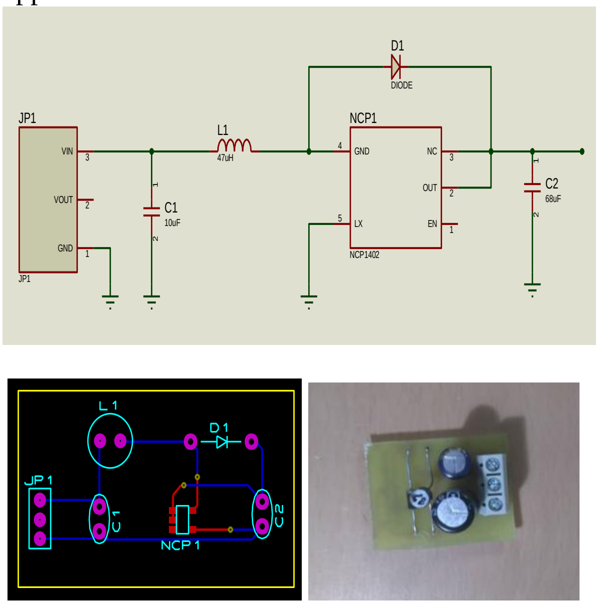

Fig. 3 Raspberry Pi W system power adapter using NCP 1402: schematic, PCB and result.

The system position was read from the WiFi adapter running Raspbian OS using the command [6]:

```
iwlist wlan0 scan
```

The result (signal level component as link quality varies too little to be used in close quarters) was parsed and the signal level was retrieved.

Three APs were used in order to estimate the drone's position. The three APs used were placed in the form of a triangle, to allow the approximation of the system/drone's position very easily. The system communication and the AP placement are seen in Fig. 4.

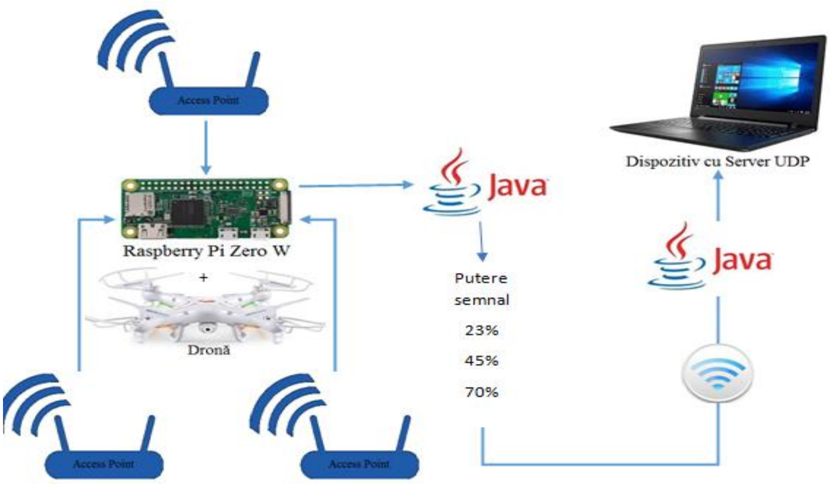

Fig. 4 System components and the communicating server and client applications

The information was then sent using a UDP client written in Java to a server. This was chosen instead of TCP in order to handle frequent disconnects.

The server application reads the UDP data and displays the result.

For developing the client and server Java applications the NetBeans environment was used.

The initial drone used was Parrot Airborne Cargo Travis (seen in Fig. 3) is able to lift approximately 30 grams. This drone has a dedicated library that can be used for communication, allowing even the control of the drone; however the system weight proved to be too much. While being able to lift off, the drone autonomy was of only a few seconds making it unusable for the testing.

Fig. 5 Parrot Airborne Cargo Travis drone with Raspberry Pi W and battery mounted.

Because of this, another small drone was chosen as the support for the Raspberry system: Syma X5C-1. This drone can lift up to 200 grams making it ideal candidate for the tests.

The software application had two components:

- The UDP client running on Raspberry Pi, executing the *iwlist* command in a loop (using the Process *Java* class), reading the output and searching for the chosen AP MACs, then – when one MAC was found – reading the signal strength. Information about signal strength for other MACs is ignored.
- The UDP server running on a desktop computer connected to an AP where the Raspberry Pi W is connected. The server runs also in a loop waiting for messages from the UDP clients. The information is then displayed on the screen in the command line.

The text output of the *iwlist* command is seen in Fig. 5. It is easy to notice that when parsing the lines, the MAC address is first, and then the line with Signal level and Noise level is present.

```
Cell 01 - Address: 00:1F:F3:02:04:81
                ESSID:"Apple Network"
                Mode:Master
                Channel:1
                Frequency:2.412 GHz (Channel 1)
                Quality=42/100  Signal level:-84 dBm  Noise level=-127 dBm
                Encryption key:on
                IE: WPA Version 1
                    Group Cipher : TKIP
                    Pairwise Ciphers (1) : TKIP
                    Authentication Suites (1) : PSK
                IE: IEEE 802.11i/WPA2 Version 1
                    Group Cipher : TKIP
                    Pairwise Ciphers (2) : CCMP TKIP
                    Authentication Suites (1) : PSK
                IE: Unknown: 2D1A2C0217FFFF000000000000000000000000000000000000000000
```

Fig. 6 The output of the iwlist command

3. TESTS AND RESULTS

A problem in the system development was that, due to the low system speed of the Pi W system, the initial tests were made using a Raspberry Pi 3 connected to the power supply with long wires.

One of the major problems was the device's autonomy. Using an 800mAh battery powering only the Raspberry Pi 3, the system was able to stay online for 30 minutes, before the power ran out.

This was acceptable as the drone's autonomy was of less than 20 minutes.

The final system implementation is presented in Fig. 7.

Fig. 7. Syma X5C-1 drone with the Raspberry system connected

The system returns the Wi-Fi strength information, which can be interpreted as [7] presents in the range of: -30dBm (maximum signal strength) to -90dBm (the point where the signal can hardly be distinguished from other signals/noise).

Using this information -30dBm was considered as maximum (nearby AP) and -90dBm was considered the extreme point. Also the scaling for these values is logarithmic; therefore a 3dBm increase doubles the signal strength and a 10dBm increase relates to a tenfold signal increase. Triangulation is made using these references and classical mathematical formulas.

4. CONCLUSIONS

The system was tested and returns the Wi-Fi strength values to the server.

Because the drone can lift even more weight, it could support a larger battery for longer flights and more components added to the Raspberry Pi (such as a camera).

As the drone and the Raspberry Pi use two different batteries, their autonomies are not linked. A future improvement of this paper would be to link the two systems so they behave like one and implement a battery monitor that shuts down the system when the battery gets too low (and even one that lands the drone).

Finally the system should use in the future the Wi-Fi strength information in order to compute the distance to the APs.

5. ACKNOWLEDGMENT

The author acknowledges the contribution of Amalia NICULAE in testing the ideas presented in this paper and making a working system prototype.

6. REFERENCES

[1] Accuware, Inc. Indoor Navigation, 2015, web, http://public.accuware.com/files/Accuware_Indoor_Navigation_Fact_Sheet.pdf

[2] In Location Alliance. “About InLocation Alliance”, 2017, web, http://inlocationalliance.org/about

[3]. S. Seidel and T. Rappaport, “914 mhz path loss prediction models for indoor wireless communications in multifloored buildings,” Antennas and Propagation, IEEE Transactions on, vol. 40, no. 2, pp. 207–217, 1992.

[4] C. Feng, W. Au, S. Valaee, and Z. Tan, “Compressive sensing based positioning using rss of wlan access points,” in INFOCOM, 2010 Proceedings IEEE, 2010, pp. 1–9.

[5] K. Kaemarungsi and P. Krishnamurthy, “Properties of indoor received signal strength for wlan location fingerprinting,” in Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on, 2004, pp. 14–23.

[6]. linux.die.net, iwconfig(8) - Linux man page, web, https://linux.die.net/man/8/iwconfig

[7] Joel Crane, Understanding WiFi Signal Strength , Web. https://support.metageek.com/hc/en-us/articles/201955754-Understanding-WiFi-Signal-Strength