Determining the position of the device in the room based on Wi-Fi

Dominik Koszkul, Michał Oleszczyk

25 January 2016

Class Intermediate Project
Instructor's name Dr inż. Witold Paluszyński
Department W-4 (Electronics)
University University of Technology in Wrocław

Abstract

The goal of the project was to create a system, which could detect smartphones or devices, which have Wi-Fi module on board and are able to create a hotspot and localize them in fixed area, ex. in a room. The project was splitted to two main parts – first was to create and program a hardware with Wi-Fi and Bluetooth modules and Arduino board. Second part was to create an desktop application which is able to communicate with PCB board. In assumptions the application is a master system, which decided when devices should start/stop sending messages, handles all computations and shows the results in special designed GUI. The system was created from two identical PCB boards. It has a few requirements, which have to be fulfilled before starting. The boards should be turned on before firing main application. To enable the algorithm, which calculates device position we have to provide a configuration, which contains size of the room and points where the boards are located.

In this project we successfully detected the device. After providing the configuration the could see the position of device in the application. The results of this project open a wide field for expansion this project. Nowadays almost everyone has a smartphone in the pocket. Base on this project we can in quite easy way create a intelligent home, which can react when we are near or in the house.

Contents

1 Introduction

1.1 Background

The main idea in this project was to create the system, which could be installed in every house and, what is important, which could be also useful. Most of modern ,,smart" systems installed in houses are based on some external applications, which have to be operated by a human, ex. Android apps connected to the server through the Internet. We wanted to create the system, which should be configured only once and then it will simply work and help us in everyday life. We noticed that Wi-Fi network is used to communicate between particular system parts but not to detect the devices. Strength of Wi-Fi signal RSSI[1] can be measured in quite simply way. So the Wi-Fi module can be used as a sensor. To detect where some device is actually located we have to use minimum two sensors (more sensors — better quality and accuracy of localization). We did not find any solution that would provide this kind of sensors to the intelligent houses so we decided to try to build the system, which will be based on ,,Wi-Fi sensors".

1.2 Assumptions and constraints

Due to limited cash and time we had to also limit work on the project. Before starting work we wrote a few assumptions and constraints:

a. the system will be prepare to work based on **two** sensors,

b. device, which should be detected has a Wi-Fi module on board and is switched to hotspot mode,

c. PCB board should communicate with a desktop application wirelessly using Bluetooth module,

d. the desktop application will be the master — decides, which PCB boards are active and handles whole computation part,

e. the desktop application has to have provided a configuration to proper working,

f. the configuration should contains size of the room and points where the boards are located

2 Software and hardware

2.1 Hardware

To design the hardware we had to decide, which parts we should use. The PCB board should not be large and has to contain three main modules:

- MICROCONTROLLER
  It should handle SPI communication, serial communication and be fast enough to get data from Wi-Fi module, parse and send them to the master application. It also be easy to program. We chose **Arduino nano** with ATmega328p on board with frequency $16MHz$

[1]Received Signal Strength Indication is a measurement of the power present in a received radio signal

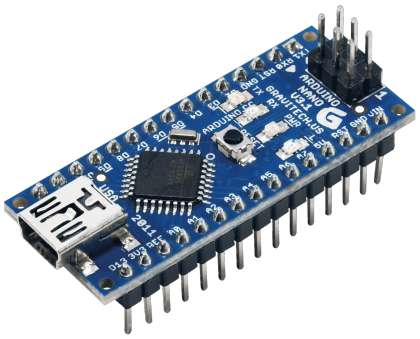

Figure 1: Arduino Nano v3.0

- Wi-Fi module
  It should communicate with μC through SPI and should have some libraries which provide basic functions allow to communicate with it. We chose **CC3000** by SparkFun. This module has on-board antenna so we do not have to buy extra one to extend scanning area. Provided antenna is good enough for project purposes.

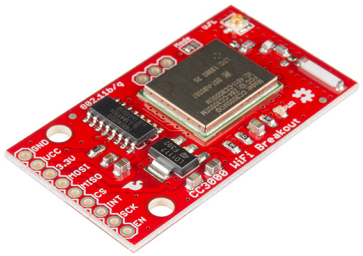

Figure 2: CC3000 by SparkFun

- bluetooth module
  We chose well known by students modules **HC-05** and **HC-06** which are easy to use and can be connected to the serial port in the μC

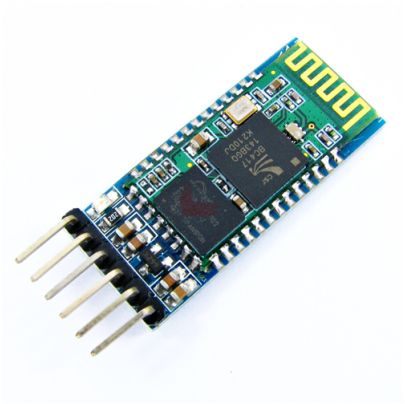

Figure 3: HC-05

In the computer is single Bluetooth module so we can connect only to one device at the time. To handle two devices we use additional USB-Serial converter to which we connect HC-05 module

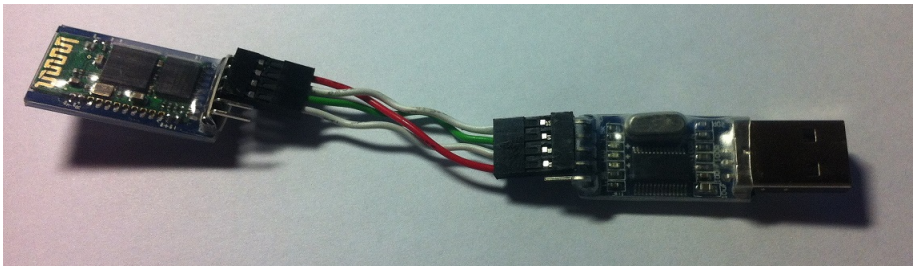

Figure 4: HC-05 - Serial - USB converter

2.2 Software

2.2.1 PCB board

To create a program to the μC we used Arduino IDE. This environment has ready to use libraries for Arduino boards, ex. to handle serial communication. The program was written in C++ language.
SparkFun provides with theirs CC3000 module the library, which we used in our project only partially. The module is able to provide MAC addresses of devices, which are in scanning area and we had to extend the library in this way that a function could also provide required by our system MAC address of the device.
The program checks before starting main loop if it is connected to CC3000 - if yes then program executes next lines of code, otherwise it is send an error though serial port and program stops working. After confirming connection with Wi-Fi module program waits for the orders from master. There are two simple orders which turn on and off scanning of area:

- *START_SCAN* — starts scanning the Wi-Fi area in the loop
- *STOP_SCAN* — stops scanning the Wi-Fi area and switch to ,,waiting for orders" mode

Minimum refresh time, which means scanning the area, parse data to JSON and send it to the master, equals 6 seconds.

2.2.2 Desktop application

Desktop application is written in Java 8. It is splitted into a few parts:

- SERIAL PORT READER
  Provides methods to connect and handle communication with the hardware.

- FRONT CONTROLLER
  GUI is written in JavaFX, which allows to create a professional looking windows and handle actions in very simple way. Program allows the user to see what data are received from each device (left part of the window), to choose MAC address of device, which will be tracked and to select the configuration containing size of the room and points where the ,,Wi-Fi sensors" are located.

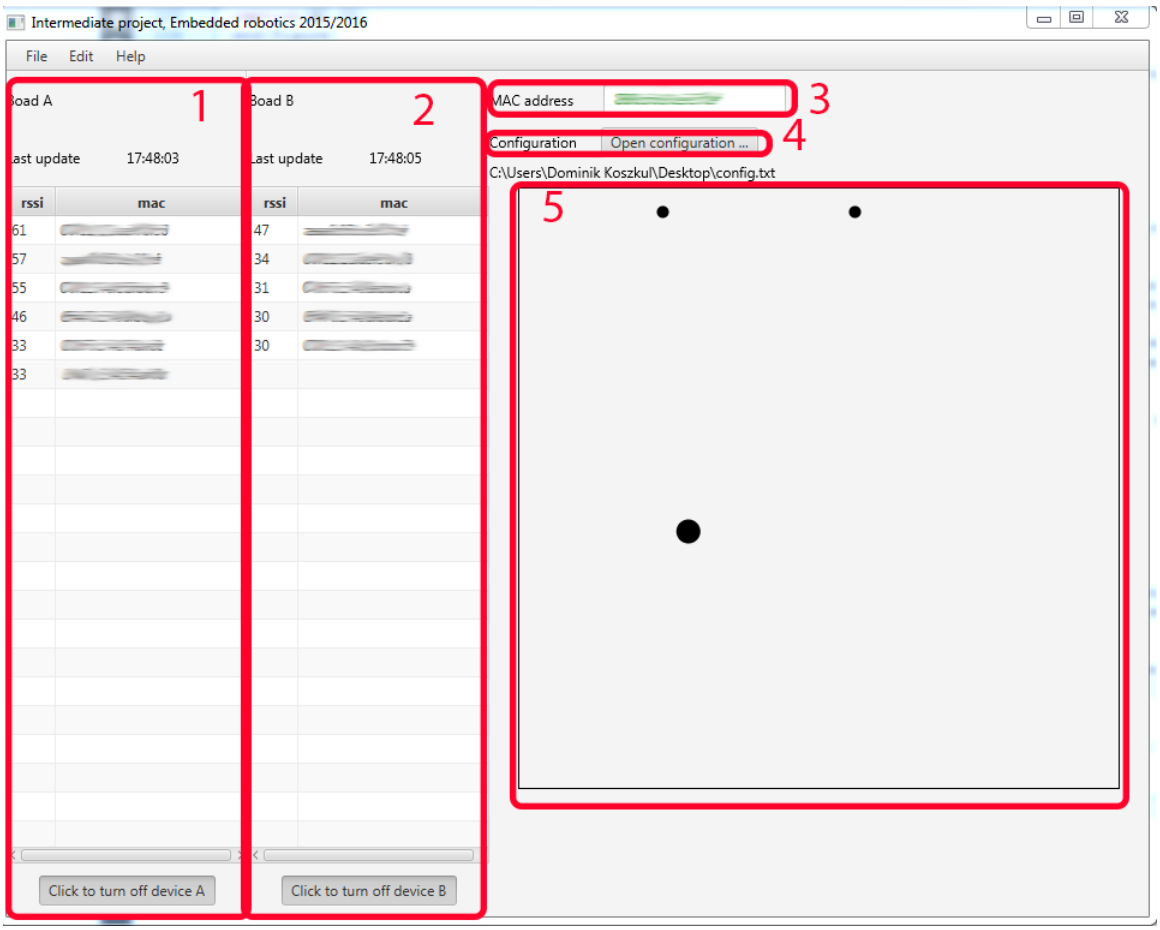

Figure 5: GUI of the desktop application

1 — raw data from device A,
2 — raw data from device B,
3 — MAC address of the device, which will be tracked,
4 — configuration chooser,
5 — graphical interpretation of algorithm results.

- ALGORITHM

  The main part of the application. It computes position of the selected device based on data gathered from the hardware. Calculations are repeated every time when all data are refreshed. The RSSI signal is very non-linear. We can notice that when tracked device stay in one place then the signal also stay in constant. In first 3 – 4 meters from the sensor we can easily see changes of signal strength. In further distances the signal strength decreases much slower. For project purposes we assume that RSSI behaves sufficiently linear and we can recount it to the distance from sensor in meters:

```
private final static Integer MINIMAL_DISTANCE_IN_CM = 100;
private final static Integer MAXIMAL_RSSI_VALUE = 70;

private Integer rssiToCentimetres(Integer rssi) {
    int value = MINIMAL_DISTANCE_IN_CM;
    if (rssi > MAXIMAL_RSSI_VALUE) {
        return value;
    }
    int difference = MAXIMAL_RSSI_VALUE - rssi;
    value += ((difference / 5) + 1) * MINIMAL_DISTANCE_IN_CM;
    return value;
}
```

  This method was developed after series of experiments and gives results close to real. After recounting RSSI to centimetres to localize the device we have to do the following:

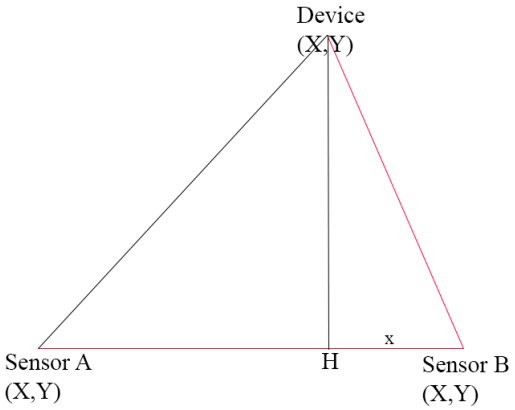

Figure 6: Explanatory figure for the algorithm purposes

1. **check dimensions**
   We have to ensure that we can build a triangle from distance from sensor A, distance from sensor B, distance between sensors. If no then we add minimal value to the shortest distance (Readings from first sensor and from second are slightly different and we can do nothing on hardware level).
2. **compute the area of the triangle**
   We use Heron's formula to compute it.
3. **compute high of the triangle which fall to the section which connects two sensors**

Having the area of triangle we can use another formula to compute the high which we are interested in. The formula in general is the following : $S = \frac{a \cdot h_a}{2}$

4. **using *Pythagorean theorem* computing x (look at figure 6)**
5. **choose one from two possible points**
   We have to choose two points — one under the section connecting sensors and one upper this line. We choose one, which simply has more sense. With only two sensors we cannot tell it explicitly.
6. **repeat the algorithm when new data arrive**

2.2.3 Desktop application configuration and pre-requirements

To fire the desktop application there have to be fulfil some pre-requirements:

- To the notebook or other used computer can be connected only USB-Serial converter (other USB ports must be free)
  Application is able to detect all devices, which are connected to the computer and then try to achieve the connection. App cannot distinguish whether connected device is Bluetooth or other unknown device like mobile phone.
- The Wi-Fi sensors have to be enabled and ready to connect.

To start the algorithm we need to load a configuration and type MAC address of the device that will be tracked. The configuration is written in JSON format. This format was chosen just to simplify way of parsing it in the application.

```
{
        "X" : 500, /* width  of the room       */
        "Y" : 500, /* height of the room       */
        "AX": 120, /* X coordinate of device A */
        "AY": 20,  /* Y coordinate of device A */
        "BX": 280, /* X coordinate of device B */
        "BY": 20   /* Y coordinate of device B */
}
```

3 Summary

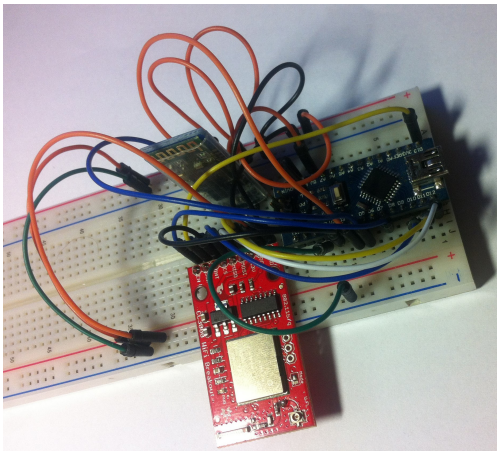

Figure 7: The Wi-Fi sensor in the prototyped PCB

3.1 Hardware part

We created a hardware board based on three main parts: Arduino Nano module, Wi-Fi CC3000 module, HC-05 Bluetooth module. Library delivered with Wi-Fi module was slightly changed by us that we could achieve MAC address in a well known form. Scanning the area by our sensor is controlled by a master application by sending simple orders using Bluetooth. Results of a scanning are parsed into JSON format and send to the master. Whole hardware part was made on the prototype PCB. We were planning to design and create dedicated PCB but due to lack of the time we were unable to do this before project deadline. Despite this our device works perfectly.

3.2 Desktop application part

To control the hardware and make a computations on raw readings from it we developed a desktop application. We handled serial connection, computations, which result is tracked device position and user interface in which we can control the devices and see results of the algorithm. The application is able to detect the devices by self. All we need to do is to turn on our devices, load the configuration, which is required by algorithm and type MAC address of a device, which we want to track.

3.3 Final results

We finish the project with positive results. Despite of limitations of measuring a strength of Wi-Fi signal we are able to tell whether the tracked device is inside or outside the room and if it is in then more or less where the device is located. Having only two devices we achieve very good accuracy of the computations. There can be situations when computed position looks wrong but it depend on readings and non-linear behaviour of Wi-Fi signal.