Monitoring Soccer Players Using a Smartphone

Andrea Tubak, Francesco Ronchieri, Carmelo Aparo, Mattia Daole
a.tubak@studenti.unipi.it, f.ronchieri@studenti.unipi.it, c.aparo@studenti.unipi.it, m.daole@studenti.unipi.it

ABSTRACT

In our work we identified the importance of performance monitoring in modern soccer. Commercial solutions are very expensive, and they cut out many possible customers. We decided to develop a cheaper monitoring system based on an android app and with the use of a set of BLE beacons from kontakt.io. The application allows to track both individual and collective training sessions, and it saves all the data on a Firebase Realtime Database. We also developed an algorithm to try and estimate the position of the user in the pitch depending on the RSSI received on the device. This algorithm is based on the analysis of the RSSI collected in experimental tests and performs with decent accuracy in a discrete cell positioning system. If more precision is required, we should consider switching to a different signaling technology other than BLE.

1 Introduction

In recent years almost all professional soccer teams have adopted the use of specialized technology to track the performances of their players. This additional data can be useful in many applications; from letting the coach understand better the status of a player, to instructing players on how to refine their position on the pitch, to the creation of beautiful infographics to be shown on live events.
The technologies used can be divided in two groups: wearable technologies and in-field technologies. Wearable technologies include all sensors that could be worn by the athletes, for example, GPS, accelerometer, gyroscope and so on. In-field technologies include the devices that are installed in the soccer arena to monitor the players, for example, multiple HD cameras mounted all around the field.
The procedures and technologies adopted to gather this data varies depending on the type of activity being monitored: e.g., official matches, unofficial matches, training sessions etc. Generally, the best results can be achieved by combining data from multiple sources, this approach, however, has some impracticalities to be addressed. An in-field system needs to be carefully mounted and calibrated, once it is in place that will be the only place in which the service is going to be provided. It also needs the assistance of a specialized crow to check in real-time if the system is working correctly.[1][2] These problems do not cause major concerns for official events as they always occur in the same stadiums and provide the economic incentive to install such a system. For training sessions, however, in-field systems would not be ideal. Wearable sensors, on the other hand, are not adopted in official events primarily because soccer players have resisted adding tracking technology into their equipment.[1] Mainly, wearable technologies are used to track the players' physical performance during unofficial matches, whereas in-field technologies are used to track the physical performance during official matches.[3]
There is, however, a remarkable scarcity of technical information around this topic, mainly because each team or organization relies on a different specialized company to install the equipment and to analyze the data produced (with proprietary software, of course). One thing is clear though, these systems are not cheap, and this is the main reason why we only see such systems in the highest levels of the sport.
This is where our application comes into play. We reckon there are many soccer-centered realities that would like to have access to performance tracking services but at a much lower price; to name a few we can think about five-a-side soccer, beach soccer, junior soccer academies etc.
In order to create a cheap alternative to professional solutions we needed to use equipment that is widely available, cheap to buy and maintain, easy to use and easy to install. Given the fact that android is the most popular OS and that android devices are already packed with many useful sensors, we decided to develop an android application to track the activity of the users and to estimate their position in the field with the use of carefully placed BLE (Bluetooth Low Energy) beacons from kontakt.io.
Our application offers the possibility to track both individual and collective training sessions and it also allows for fast and easy relocation of the whole positioning system.
The use-case is pretty straight forward: download the app, login with your username, buy and place the beacons (at least 4) in the configuration explained in 3.1 Experimental Testbed, and then you are ready to track your activity or the activities of your team.
Finally, our design has also the possibility to be extended by placing more beacons to cover a wider area, although we should point out that our focus was on soccer activities which are carried out on fields with a limited footprint.

2 Architecture

The core of the architecture is the android app named Training Stat which interacts with the smartphone sensors to collect user activities data and store that data into a NoSQL database (Firebase Realtime Database) to retrieve them later.

2.1 Android Application

The Application is written in Java using Android Studio and it is composed by five main android Activities:

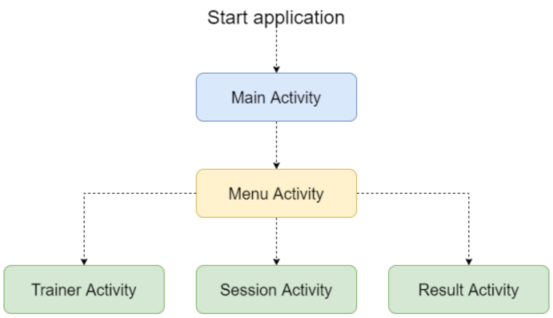
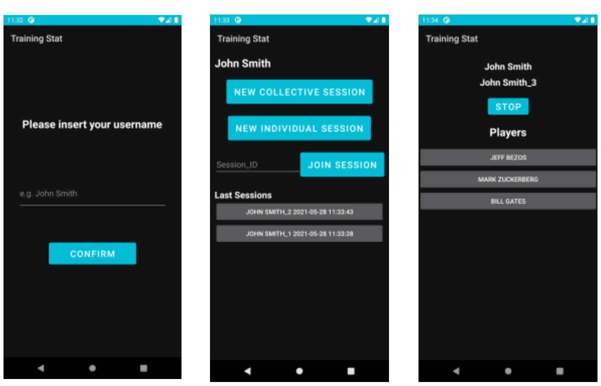
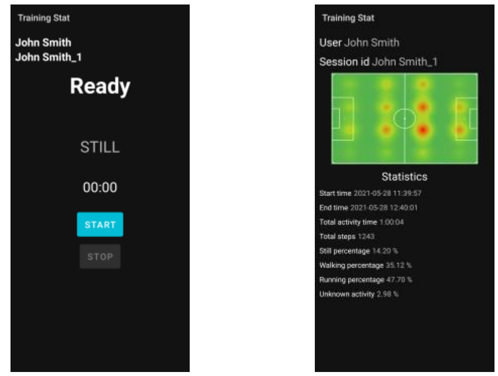


Figure 2.1: navigation hierarchy

The Main Activity (Fig.2.2.a) is responsible for the authentication of the user (to focus our resources on more relevant parts of the application, the authentication phase is just a text box where the user can insert a username and continue as that user). In the Menu Activity (Fig.2.2.b) the user can start a training session, join a training session, or start an individual session, he/she can also see a list of past sessions in order to join them again, if they are still running, or watch the results, if they are terminated.
Trainer Activity (Fig.2.2.c) is triggered when a user wants to start a new training session from the menu, the user becomes the trainer of the session and from there he/she can see players joining in his/her session and when they terminate their session, he/she can consult their results by clicking on their names. When all players terminate their session, a new entry is available with the username of the trainer and from that entry the trainer can see the aggregated data of all players.
Session Activity (Fig.2.2.d) is opened when a user starts an individual session or joins a training session, in this activity the user can start/pause or stop the session and during monitoring the application collects data from smartphone sensors and periodically stores it in the database.
Finally, the Result Activity (Fig.2.2.e) is accessible when a user stops his/her individual session, by the trainer in the players list of the Training Activity and by the menu in the Past session section. In this activity we can see a set of statistics: total activity time, the percentages of type of activity, total steps and a heatmap of the areas of the field most traveled by the user.



(a) (b) (c)



(d) (e)

Figure 2.2: Screenshots of the activities

2.2 Collecting data

As already said the Session Activity is responsible to collecting user activity data, in order to do that it makes use of events and intents. Inside the Session Activity there are 3 main components that are in charge to interface with 3 different service

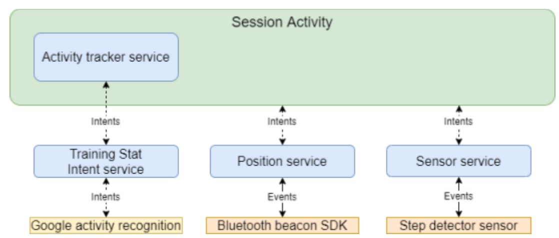


Figure 2.3: Session Activity structure

Sensor service is a foreground service which setup a listener and notifies the Session Activity with a broadcast Intent every time a step is recognized by the step detector sensor of the device.

Position service uses the kontakt.io SDK to collect data from the device's Bluetooth and understand where the user is positioned based on beacon information received, then notifies the position to the activity with a broadcast intent, this service is also a foreground service.

The Training Stat Intent service is an Intent service, and it is necessary to receive information from the Google activity recognition API, then it applies some logic to received intents (such as filtering unnecessary information) and forward them as a broadcast intent to our application.

Finally, the Activity Tracker service is a java class inside the Session Activity that listens for broadcast intents from the Training stat Intent service and stores information about the received status making them available to the Session Activity by object calls.

2.3 Database

As mentioned before, we make use of the Firebase Realtime Database to store our data which is a NoSQL Database based on JSON objects.

Under this assumption for our application it has been identified this JSON data organization

```
{
  "beaconPositions" : {
    "...": "..."
  },
  "trainingSessions" : {
    "John Smith_3" : {
      "...": "...",
      "userSessions" : {
        "Bill Gates" : {
          "...": "..."
        },
        "Jeff Bezos" : {
          "...": "..."
        },
        "Mark Zuckerberg" : {
          "...": "..."
        }
      }
    }
  },
  ...
```

```
...
"users" : {
    "Bill Gates" : {
      "...": "..."
    },
    "Jeff Bezos" : {
      "...": "..."
    },
    "John Smith" : {
      "...": "..."
    },
    "Mark Zuckerberg" : {
      "...": "..."
    }
  }
}
```

Note: Even the individual sessions are considered as training sessions with only one user whose username is the same of the trainer.

3 Experiments

In the next section we describe the experiments done in order to understand the BLE beacons behavior in order to define a position algorithm.

So, we start describing the experimental testbed and the patterns found during the data analysis and then, we explain the position algorithm testing also its accuracy and errors.

3.1 Experimental Testbed

Our mobile application wants to provide at the end of the training session an heatmap for each player.

The positioning system we developed during our work, which make use of BLE beacons, is based on a logical grid system where we divide the squared region with four beacons as the vertexes, in cells of the same size. This choice is tied to the fact that in an heatmap we are not interested in the position of the player with a very high precision (like centimeters), we focus more on the area where the player is active.

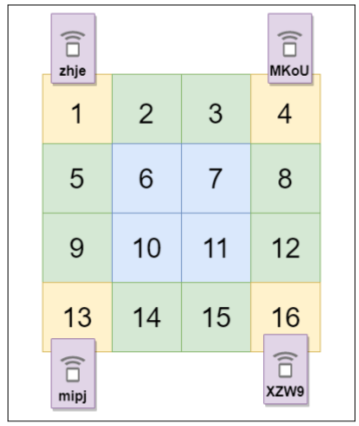


Figure 3.1: Logical grid system of the experimental testbed used

In figure are highlighted three different types of cells:
in yellow we represent the “angular” cells, where the beacons are positioned, in green the “perimetral” cells and lastly in blue the “central” cells.
The reason behind this distinction will be further explained in the positioning algorithm.

These squared regions should be replicated to cover the whole soccer field to obtain a complete coverage over the soccer field for the heatmap, for practical reasons we experimented using only one squared region.

The beacons were placed at 8 meters of distance between each other and multiple measurements have been taken standing still for a prolonged period of time on the center of each type of cell.

Dealing with a scenario in which walking and running are involved, and having the beacons transmit their advertisement messages roughly every 0.3 s, we adopted a scanning period of 0.5s in order to receive at least one advertisement from each beacon around the user, still maintaining a short period in order to avoid spatial smearing problems [4].

3.2 Data Analysis

In the dataset created during the experimental phase we tried to identify specific RSSI pattern depending on the type of cell we were considering, with beacons placed at 8 meters between each other.

Taking for example the 1° cell, which belong to the “angular” type, we observed that the RSSI measured for the beacon deployed in that cell was much stronger than those of the other beacons, in figure is shown the average over all measurements obtained:

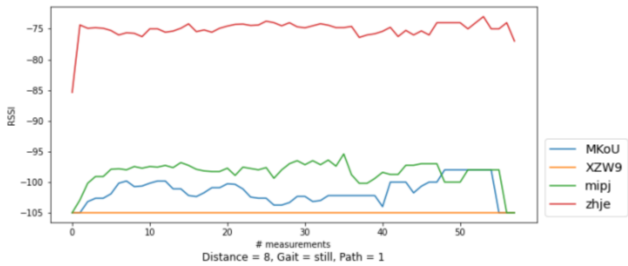


Figure 3.2: Average Rssi Measurement Graph for cell 1

This is an expected result, because in this test the beacon with id “zhje” was the nearest to us, while the others were 7-10 meters away.

For the “perimetral” case instead, for example in cell 2, we observed a different behavior, the two nearest beacons are the only ones that are detected in most of the measurements:

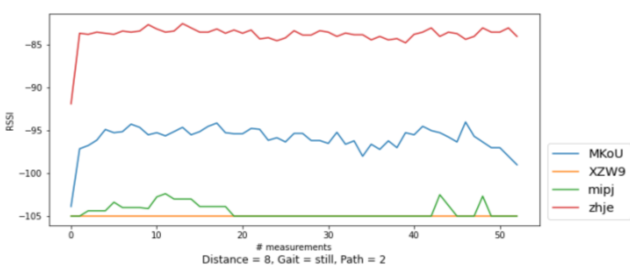


Figure 3.3: Average Rssi Measurement Graph for cell 2

In the case of “central” cells instead the signals are received by 3 or all 4 beacons in each scan:

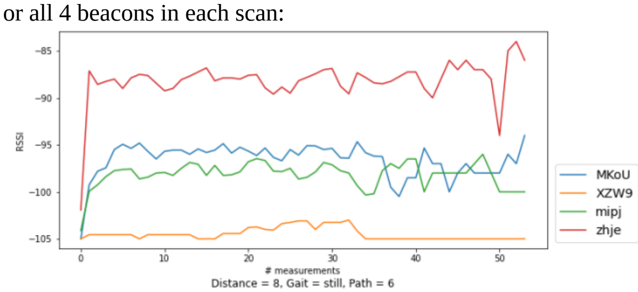


Figure 3.4: Average Rssi Measurement Graph for cell 6

3.3 Position algorithm

The position algorithm is based on a grid similar to the one used in the experimental testbed and on the observations done in the previous paragraph.
The soccer field is covered by a set of grids 4x4 and one beacon is put on every angle of each grid.
The coordinates of the cell nearest to a beacon are stored in a table, so that the application can find in this table the association between the ID of the beacons received during the BLE scanning and their coordinates on the grid that covers the soccer field.
Due to the distance among the beacons (at least 8 meters), on each 4x4 grid the signals received by the user are only the ones of the beacons that are on the angles of that 4x4 grid. For this reason, the 4x4 grid on which the user is can be identified by looking to the beacons received and then the application can identify the position of the user inside that 4x4 grid.
The position algorithm is divided in two parts:

1. the aim of the first one is to identify the type of cell on which the user is. The possible types are: angular, perimetral and central;
2. the aim of the second one is to identify the coordinates of the cell on which the user is.

During the first part the algorithm checks the number and the Received Signal Strength Indicator (RSSI) of beacons received during the scanning using the following criteria:

- if the number of beacons received is 1 or the strongest RSSI is much stronger than the other ones, the cell is estimated as angular, since it means that the user is very close to the beacon with highest RSSI and very far from the other ones;
- if the number of beacons received is 2 or the two strongest RSSI are much stronger than the other ones, the cell is estimated as perimetral, since it means that the user is between the two beacons with highest RSSI and far from the other ones;
- if the number of beacons is more than 2 and the three strongest RSSI are similar, the cell is estimated as central, since it means that the user is at the center of the grid.

During the second part, starting from the type of cell and the coordinates of the beacons with strongest RSSI, the algorithm finds the coordinates of the cell in which the user is according to the following criteria:

- if the cell is angular, the user is in the same cell of the beacon with strongest RSSI, so the estimated coordinates are the ones of the cell nearest to that beacon;
- if the cell is perimetral, the user is between the two cells with strongest RSSI, so the estimated coordinates are the ones of the perimetral cell in the middle of the angular cells where there are the two beacons and nearest to the cell of the beacon with strongest RSSI;
- if the cell is central, the user is at the center of the 4x4 grid, so the estimated coordinates are the ones of the central cell nearest to the cell of the beacon with strongest RSSI.

3.4 Results

In order to test the accuracy of the algorithm described in the previous paragraph, a set of 2280 measures on the different types of cells were collected by putting 4 beacons at the angles of a 8m x 8m square.

Cell type	Accuracy	Error distance	Error
Angular	78,87624615	1,277979458	2,555958917
Perimetral	64,05286231	1,036176179	2,072352358
Central	66,15277062	1,063404895	2,126809789

Figure 3.5: Table of accuracy and errors for the different types of cells

The figure 3.5 shows the results obtained from the test. The best accuracy is obtained on the angular cells where the cell is correctly identified almost 79% of the time, instead the worst one is on the perimetral cells where the cell is identified correctly only 64% of time.
The column "Error distance" represents the mean distance between the correct cell and the estimated cell when the cell is wrongly estimated. As the table shows, the mean error distance is about 1 on all cell types, that means that on wrong estimations the estimated cell is adjacent to the correct one.
More in detail, the mean error in meters goes from 2 meters to 2,55 meters depending on the type of the cell.
The accuracy obtained from this test is not really high, so it can be supposed that this system is not very good to estimate the position of the user with high accuracy. However, in the case of Training Stat application the positioning system is used only to build a heatmap, so the application wants to identify the area on which there is the user and not the exact position in the field. For this reason, it is useful to do a comparison between the real heatmap and the heatmap estimated by the positioning system.

(a) **(b)**

Figure 3.6: (a) Real heatmap. (b) Estimated heatmap.

Figure 3.6 shows the real (a) and the estimated (b) heatmaps. In this case the heatmap is built from a single 4x4 grid where beacons were 8 meters far from each other. In the image the grid covers the entire soccer field, however, on a real deployment, the soccer field is covered by many 4x4 grids, so each one covers a much smaller area of the field with respect to the one seen in these figures.
It can be noticed that the shape of the heatmaps is very similar, they highlight the same areas of the field with some small difference on the intensity of the colors.

So, even if the accuracy of the system is not very high, it can still be used to identify the area on which the user is by building a heatmap.

4 Conclusion

After analyzing the experimental results, we observed that the accuracy of our approach stands between 79% and 64% depending on the type of cell in which the user is placed. The granularity of the position estimation algorithm is not great compared to professional solutions but considering the overall price range we believe it to be a valuable low-cost tool in many circumstances in which strict precision is not required.

The main issue limiting the effectiveness of our solution is the unreliability of BLE (Bluetooth Low Energy) signal with respect to obstacles: e.g., other players between the line of sight, the orientation of the player with respect to each beacon. The resulting RSSI signal is too noisy, therefore it is much harder to differentiate between different cells, let alone estimating the position of the user with greater precision.

A possible extension on this work may be looking into different signaling technologies in order to guarantee a clearer distinction between different cells, and possibly obtain a greater precision. A more extensive set of statistics should also be considered.

REFERENCES

[1] https://www.vice.com/en/article/gvyy4q/this-system-turns-the-beautiful-game-into-big-data

[2] https://www.youtube.com/watch?v=ujakhyFWQ8E

[3] https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-020-01156-4

[4] Location Fingerprinting With Bluetooth Low Energy Beacons