David António Freire Moura

Bachelor in Computer Science

Exploratory Analysis of Individual Metrics in Team Sports Using *IoT* Sensors

Dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Computer Science and Engineering

Adviser: Carmen Pires Morgado, Assistant Professor,
Faculdade de Ciências e Tecnologia da
Universidade Nova de Lisboa

Co-adviser: Ruben Duarte Dias da Costa, Director,
Knowledge Biz Consulting

Examination Committee

Chair:	Name of the committee chairperson
Rapporteurs:	Name of a raporteur
	Name of another raporteur
Members:	Another member of the committee
	Yet another member of the committee

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

DRAFT: 2020-02-07

Exploratory Analysis of Individual Metrics in Team Sports Using Inertial Sensors

This document was created using the (pdf)LaTeX processor, based on the "novathesis" template[1], developed at the Dep. Informática of FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Lorem ipsum.

Acknowledgements

The acknowledgements. You are free to write this section at your own will. However, usually it starts with the institutional acknowledgements (adviser, institution, grants, workmates, ...) and then comes the personal acknowledgements (friends, family, ...).

Abstract

Team Sports like basketball, football and baseball rely on data insights to improve player and team performance, practice scheduling and injury recovery. Recent improvements in data mining and machine learning techniques have encouraged the research of better ways to collect player data.

There are different approaches currently available, using video tracking systems or wearable sensors. The latter systems can be very precise in tracking position outdoors using GPS or indoors using Ultra-Wideband or RFID positioning, they fall short in the analysis of game related metrics, like accelerations and jumps or passes and shoots in the case of Basketball.

This work proposes a solution that can track both position and metrics of players and teams in real-time using inertial sensors, providing the users (e.g. players, coaches) an application with insightful information in order to improve the performance of the individual player and the whole team.

Resumo

Desportos colectivos tais como o basquetebol, futebol ou basebol usam dados para melhorar o desempenho em jogo, o planeamento de treinos e a recuperação de lesões de jogadores e da equipa. O aperfeiçoamento de técnicas como *data mining* e *machine learning* têm levado à pesquisa de formas de recolher mais e melhores dados dos jogadores.

Hoje em dia há várias abordagens diferentes: usando sistemas de captura de video ou utilizando sensores. Esta última abordagem pode ser bastante precisa no posicionamento em exteriores, utilizando sistemas como o GPS, ou em interiores, utilizando sistemas de posicionamento como Ultra-Wideband ou RFID. No entanto, estes sistemas não efetuam o reconhecimento de métricas de jogo como acelerações ou saltos, ou passes e lançamentos no basquetebol

Este trabalho propõe uma solução que visa medir em tempo real o posicionamento de jogadores e da equipa no campo, assim como recolher métricas de jogo, utilizando sensores de inércia. Deste modo, pretende-se fornecer aos utilizadores (e.g. jogadores, treinadores) uma aplicação com informações úteis e detalhadas sobre o desempenho dos jogadores individualmente, e da sua equipa como um todo.

Contents

List of Figures

List of Tables

Listings

Acronyms

BLE	BLuetooth Low Energy
BR/EDR	Basic Rate/Enhanced Data Rate
ECG	Electrocardiography
EPU	Edge Processing Unit
FIFA	Fédération Internationale de Football Association
GLONASS	GLObal NAvigation Satellite System
GNSS	Global NAvigation Satellite system
GPS	Global Positioning system
IMU	Inertial Measurement Unit
IoT	Internet of Things
KBZ	Knowledgebiz Consulting
MEMS	Micro-electromechanical systems
NBA	National Basketball Association
NFC	Near Field Communication
NFL	National Footbal League
PPG	Photoplethysmogram
RFID	Radio-frequency identification
UWB	Ultra-wideband

Symbols

Chapter 1

Introduction

1. Big picture dos desportos e tracking em desportos
2. Falar da monitorização de jogadores com câmaras e sensores
3. Problemas dessas abordagens.
4. Motivação e desafios
5. Objetivos:
    - Maior precisão em métricas individuais usando wearables
    - Abordagem low-cost com IMUs
    - Solução wearable com processamento local e central, com vários sensores por jogador
    - Métricas a medir

Chapter 2

Related Work

Porque é que vou falar destes tópicos? Porque não vou falar do GPS? "Como falado nos objetivos....."

Trabalho relacionado sobre:

1. Algoritmos de tracking de posição usando IMUs
    - Analysis of NBN23 system:
    - Madgwick
    - Carl Fischer
    - https://iopscience.iop.org/article/10.1088/1757-899X/138/1/012005/pdf
2. Algoritmos de métricas
    - 9 movement recognition with neural networks:
3. Outros trabalhos de métricas de performance (em equipa?)
4. ???Rede de Sensores???

Chapter 3

System Proposal

To achieve the goals set in Chapter 1, a system that collects and processes the raw data from the IMU sensors and transforms it into meaningful metrics, storing it, and then presents it in a clear way to the user has to be built. This chapter proposes a system architecture

3.1 Requirements

1. escalável
2. indoors
3. n de jogadores basket
4. real-time

3.2 Architecture

The proposed architecture is shown in Figure 3.1, and it is comprised of three segments: Edge - Data Collection and Processing; Server - Data Storing and Application Server; Client - Frontend Application

—Justificar escolhas da arquitetura—

3.2.1 Edge

The Edge segment is composed of at least one Inertial Measurement Unit (IMU) Sensor and one Edge Processing Unit (EPU), that controls and receives information directly

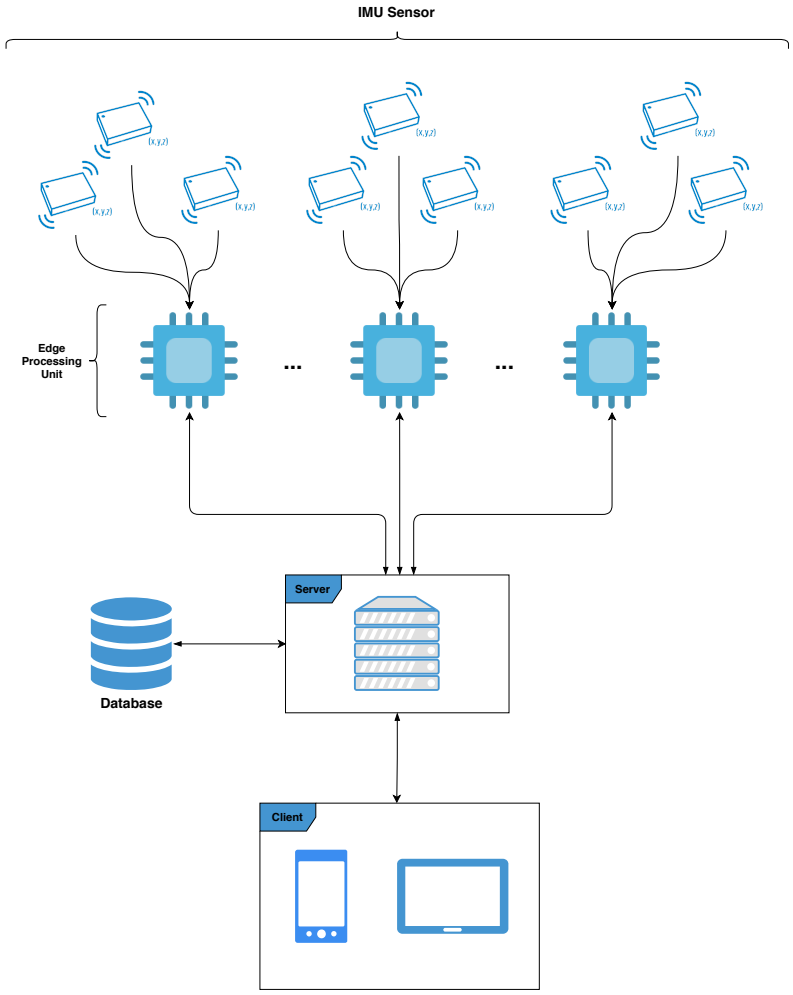

Figure 3.1: Proposed Architecture

from the IMU Sensor. The EPU should also be capable of calculating insightful metrics, calculated with the raw data acquired from the IMU Sensors.

3.2.1.1 IMU Sensors

A IMU Sensor is an integrated sensor package that measures a body's specific force, angular rate and orientation.

Specific force is a measurement of coordinate acceleration, which can be obtained by removing the gravitational acceleration. It is measured with accelerometers. Angular rate is the rate at which a body rotates, measured by gyroscopes. Orientation is a description of how a body is placed in the place it occupies. Using magnetometers, a body can be oriented relatively to the Earth's magnetic field.

For this application, IMU Sensors armed with 3-axis accelerometer and gyroscope should be used.

The IMU sensors should attached to the player's body, in different locations (like the

back, the foot or the hand), to track different metrics.

3.2.1.2 Edge Processing Unit

The EPU does the "hard work". Besides handling the connection and controlling the IMU Sensors operations, like starting and stopping the raw data collection, this unit has to collect the raw data sent by the IMU Sensors several times per second, and perform calculations with the gathered raw data to measure in-game metrics, which are then sent to the server, to be stored and shown to the User.

3.2.2 Server

The Server is the main piece of the architecture, allowing the interaction between the User and the IMU Sensors.

The Server can send instructions to the EPU, controlling the IMU Sensors operations. When data is received from the EPU, the Server stores it, so that it can be shown to the User, through the Client application.

The Server also hosts the Client Application, which communicates tightly with the Server.

3.2.3 Client

The Client Application provides the User an interface in which he can control and consult information about players, teams and games.

Indirectly he should trigger the start and stop of the data collection in the IMU Sensors, which will then send the raw data to the EPU. After the game metrics are calculated in the Edge, they are shown to the user, in the Client Interface.

Chapter 4

System Implementation

—//—

1. Raspberry Pi
    a) quais os algoritmos
        i. Dribbles
        ii. Saltos
        iii. Passos/Distância/Posição
        iv. Tempo parado/andar/correr
    b) operações sensores
    c) como se controlam os sensores
    d) como são enviados os dados (mqtt)
    e) feito em node
2. Server
    a) REST API feita em Node
    b) DB em mySQL, explicar modelo
    c) como troca mensagens com o rPi
3. Cliente
    a) Feito em React
    b) Permite gerir equipas, jogadores e jogos
    c) métricas por jogo, vários sensores por jogador

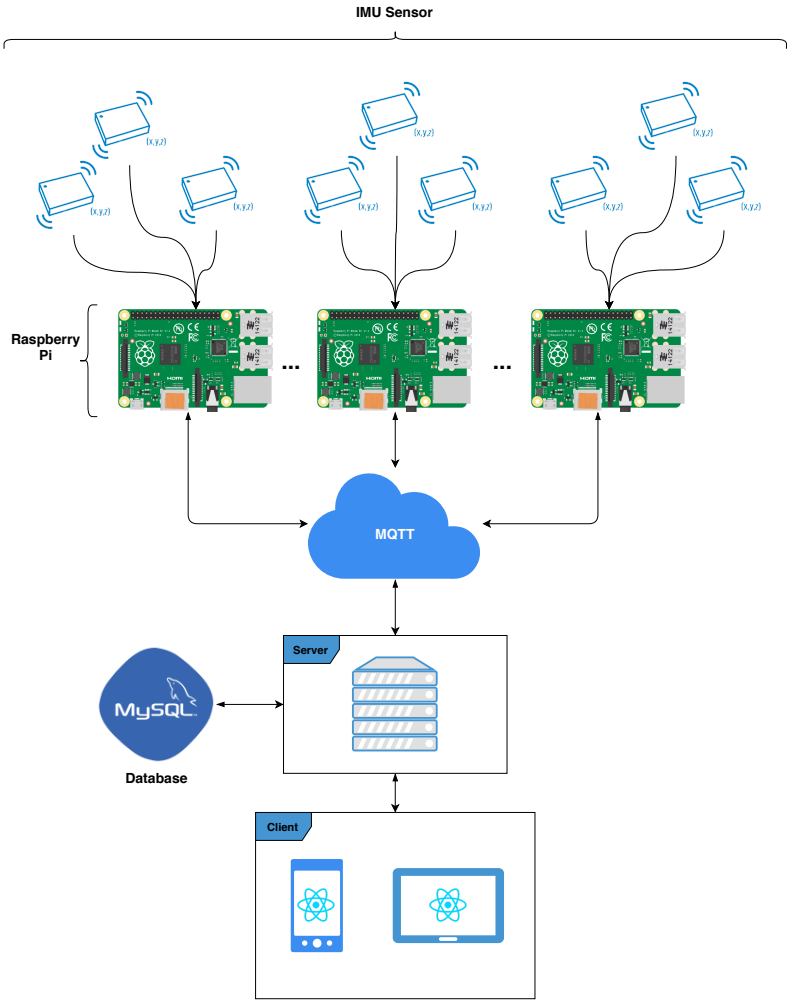

Figure 4.1: Implemented Architecture

aaa ——//——

The description of the System Implementation according to the proposed architecture will be separated in the three segments: Edge, Server and Client. Figure 4.1 represents the implemented architecture.

4.1 Edge

As described in section 3.2.1, the edge segment is composed by one or more IMU Sensors and one or more EPU.

The IMU Sensors used in the implementation phase are armed with 3-axis accelerometer, which measures the specific force in m/s^2, 3-axis gyroscope, which measures the angular rate in $°/s$, and 3-axis magnetometer, which measures the strength of magnetic fields, in μT. The IMU sensors used also have a battery, internal storage, and communicate through Bluetooth Low Energy.

In this phase, a Raspberry Pi was chosen to have the role of EPU. A Raspberry Pi is a single-boarded, small and low-cost computer developed by the Raspberry Pi Foundation, with the goal of enabling people of all ages to explore computing. It was chosen for this system for its easy accessability in the market, reduced size, high portability and low price, its wide range of features, like Bluetooth and WiFi communications, and being a good platform for developing an application prototype.

The number of IMU sensors used can vary in each player by the number of metrics being measured (different metrics may require the use of more sensors) and the number of players being tracked.

As the number of sensors can grow, and due to the limitation of the number of connections to a Bluetooth receiver, the number of Raspberry Pi's can also grow, in order to establish connection with all the IMU Sensors. This also helps to share the computation of the game metrics between the Raspberry Pi's.

4.1.1 IMU Sensor Control and Communication

The role of the IMU Sensors is to receive instructions issued by the user, and send accelerometer and gyroscope raw data to the Raspberry Pi they are connected, over Bluetooth.

The following operations are supported by IMU Sensors:

- State control
  - Start Raw Data Collection
  - Stop Raw Data Collection
  - Shutdown
- Sample Rate
- MPU Configuration

The IMU Sensors must pair with a Raspberry Pi, which is searching for the known sensors. After being paired, the operations can be communicated to the IMU Sensors through Bluetooth Notifications.

Before starting the data collection, it is necessary to set the sample rate to 50 Hz (necessary for the metrics algorithms), and to activate the gyroscope and the magnetometer sensor by changing the MPU Configuration. By default only the accelerometer sensor is active.

To start collecting data from an IMU Sensor, it is needed to provide information like the MAC Address of the sensor, an identification of the game being played and the player wearing the sensor (because the sensors can be changed from game to game between the players). It is also needed to send the position of the sensor in the player's body, to

calculate the metrics, as each algorithm uses data collected from a different part of the body.

When the metrics are calculated, they are sent to the server via MQTT, to be stored there. The Raspberry Pi's don't store data, they only server as a vehicle of instructions and data between the server and the sensors.

A single Raspberry Pi is then responsible by maintaining the connection with several IMU Sensors; communicate instructions; receive accelerometer, gyroscope and magnetometer data, and stores it temporarily; depending on the position of the sensor, calculate the according metrics algorithms with the received data; send the calculated metrics to the Server.

4.1.2 Metrics Algorithms

To achieve the goal to collect insightful information of a basketball game, a set of metrics were chosen, focusing on three body locations: foot, lower back and hand. Using the sensor in the foot we can infer the number of steps, traveled distance and player trajectory.

In the hand, it is possible to detect the number of dribbles, and time dribbling. With this, it is also possible to calculate the time of ball possession of the whole team. In the lower back, the sensor was used to collect data and analyze it to detect the number of jumps and time in air.

The following sections will explain in detail how the algorithms work.

4.1.2.1 Trajectories

To track the player's trajectories, the selected approach was Implementing a Pedestrian Tracker Using Inertial Sensors [1],

4.1.2.2 Steps

The detection of steps takes advantage of the algorithm explained in 4.1.2.1, especially the Zero-Velocity Detection. Whenever a Zero-Velocity is detected, it is recorded in a binary array (zeros meaning no movement, ones meaning movement). Afterwards, when the Trajectory detection algorithm ends, the array is analysed.

The analysis is made by counting the meaningful "sections" of ones. Listing 4.1 shows an example of a step detection array, where we can clearly see there are two main kernels of ones, meaning that using this example the algorithm should count two steps.

Listing 4.1: Steps Array Example

```
1 [0000000000100001111111111111111000100000000000000010111111111111111111100110000000000]
```

However the algorithm must take into account the outliers, and shouldn't identify it as steps. This can be ensured by only counting after a number of consecutive ones is observed. If a zero appears in the middle, the possible step is discarded. The behaviour of the algorithm is described in Listing 4.2

Listing 4.2: Steps Counting Algorithm

```
steps = 0;
no_ones = 0;
no_zeros = 0;
last = 0;
out = [pos_r(1,:); pos_r(2,:); walking(:)'];
printf('{"data":␣[');
%% Algorithm to identify steps in ones %%
for idx = 1:data_size
    pos_r(:,idx) = rotation_matrix*[pos_n(1,idx) pos_n(2,idx)]';
    current = walking(idx);
    if (last == 0 && current == 1)
        no_ones++;
        no_zeros = 0;
    elseif (last == 1 && current == 1)
        no_ones++;
        if (no_ones == 12)
            no_zeros = 0;
            steps++;
            %%%STEP%%%
        endif
    elseif (last == 1 && current == 0)
        no_zeros++;
    elseif (last == 0 && current == 0)
        no_zeros++;
        no_ones = 0;
    endif
    last = current;
endfor
```

4.1.2.3 Dribbles

The analysis of dribbling is made using an IMU Sensor placed in the wrist of the dominant hand, as shown in figure , and uses the data from the Y-axis accelerometer and X-axis gyroscope.

4.1.2.4 Jumps

4.2 Server

The server is built as a Rest API, developed with Node.js, and taking advantage of the Express framework that enables a fast development of Web Applications and API's.

To manage peripherals (IMU Sensors), Players, Teams, and Games, there are API endpoints for each one of this resources, and all the data is saved in a MySQL Database.

4.2.1 API Description

The Developed API exposes endpoints to control the different resources available in the system. Those resources are:

- Peripherals (IMU Sensors)
- Players
- Teams
- Games

4.2.1.1 Peripherals

4.2.1.2 Players

4.2.1.3 Teams

4.2.1.4 Games

4.2.2 Database Model

The database, developed in MySQL, was designed in order to store all the data from the main resources, its relations and meaningful data in those relations.

The main resources are represented in the following tables: Peripherals; Players; Teams; Games; Metrics.

Each entry in the Peripheral table represents an IMU Sensor, and it's stored the peripheral MAC Address, and its attributed number for easier identification.

In the Players table, each player has stored its name, and an identifier of the team they belong to.

The Team table only stores the name of the team.

To store information about a game, the Game table has the date of the game, and a reference for both teams attending the game.

For the metrics, each player has a set of metrics per game, with the data gathered from all the IMU Sensors he's using.

One of the features of the system is to be able to handle multiple peripherals per player, in different body positions, that can be changed between games. To store this information, two tables are needed: one that stores the relation between the player and the game, and another that relates that with a peripheral, storing where the peripheral is placed in the player's body.

4.3 Client

The client application was developed in React, with the goal of controlling the IMU Sensors, and manage teams, players and games. It should also display Player and Team game-related metrics.

Chapter 5

Results

Table 5.1: Percentage of data loss (Raspberry Pi 3b+).

	0,6m	1,2m	3m	6m	9m	12m	15m	18m
1 Device	4,27	4,39	11,25	18,22	28,04	15,43	28,16	19,79
3 Devices	5,65	7,68	12,50	18,79	21,76	16,60	39,50	26,78
6 Devices	6,52	6,05	15,53	16,31	21,64	19,77	32,90	21,78
9 Devices	6,17	7,63	12,80	19,98	28,65	36,62	40,37	32,96

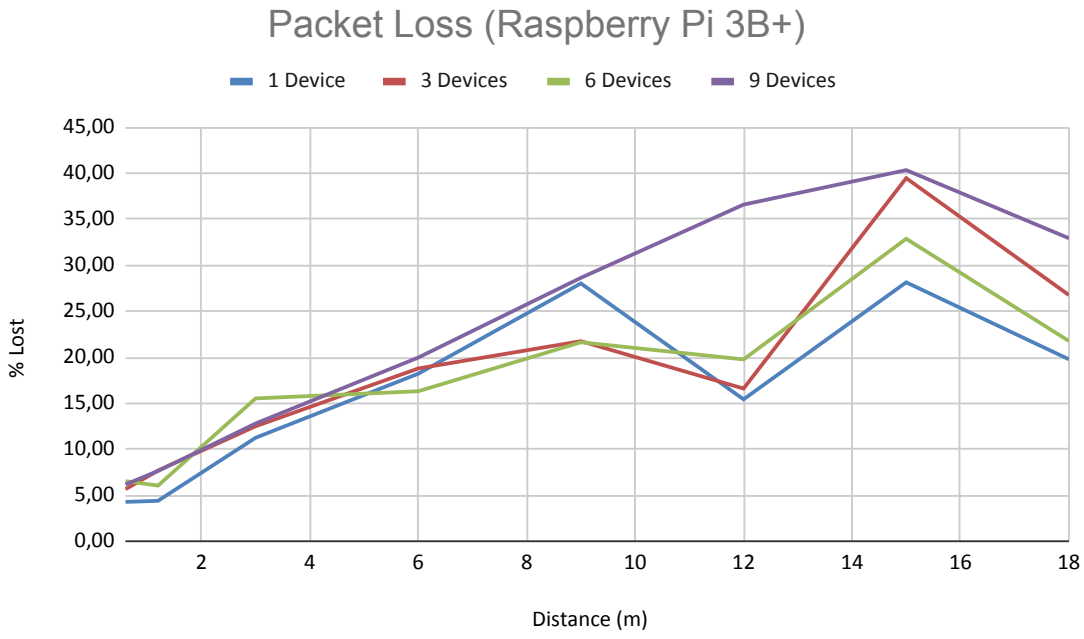

Figure 5.1: Percentage of data loss (Raspberry Pi 3b+)

1. Comunicação - Perda de pacotes IMU -> rPi
    a) raspberry Pi 3b+
    b) raspberry Pi 4
2. Fiabilidade dos algoritmos
    a) Dribbles
    b) Saltos
    c) Passos/Distância
3. Apresentação dos dados

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Revisitar os objetivos e comparar o trabalho feito com aquilo que foi definido. Avaliar o resultado final.

6.2 Future Work

ERROR: File 'elaboration7' does not exist!!!

Bibliography

[1] C. Fischer, P. T. Sukumar, and M. Hazas. "Tutorial: Implementing a Pedestrian Tracker Using Inertial Sensors." In: *IEEE Pervasive Computing* 12 (2012), pp. 17–27. DOI: 10.1109/MPRV.2012.16.