Tennis Gets Smart:

Machine Learning Based Tennis Personal Coach

SS 2017

Wearables Computing Laboratory

Embedded Systems chair, University of Freiburg

Amir Kallel | Alexander Kozhinov | Suresh Dharani Parasuraman

09.08.2017

Table of Contents

09.08.2017

1. Introduction

Tennis is a fun game to play, however quite difficult to get better at: a Tennis club membership is usually the only option for ambitious players who want to level up their game. This comes with a big cost and time commitment.

Fortunately, a smart racket may replace the effort of a personal trainer at the club as well as the pricey bill that follows. This project serves as proof of concept as an alternative to a personal coach while training, where it guides the player via live feedback of his tennis moves in order to develop a proper technique.

Figure 1 : Professional tennis player swing motion illustrated [1]

2. Theory and Methods

2.1. Tennis swing modeling

A parametric model has to be defined to model the correct swing motion. 93 data set was recorded while performing correct swing motions. This data is then plotted and analyzed using Matlab and modeled using the curve fitting toolbox plugin with different fit types: Polynomial, gaussian etc. The most appropriate fit type determined is the gaussian fit (see equation 2.1) due to the close resemblance to the data recorded as seen in figure 5.

$$y = Ae^{-\frac{(x-\mu)^2}{2\sigma^2}} + b \text{ [2]} \tag{2.1}$$

Where:

$A = \dfrac{1}{\sigma\sqrt{2\pi}}$ **- no it is not. Just leave it be A - the amplitude can be any value between -20 m/s^2 and 20 m/s^2 for acceleration for example. So A- is the parameter to be estimated and not depends on sigma in our case!!!!**

μ : Mean representing the amplitude of the data recorded.

σ : Standard deviation representing the duration of the swing recorded.

b : Bias

5 out of 9 sensor data channels were modeled using 1st and 2nd gaussian distributions as shown in the following table:

Table 1: Fit types and features of sensor channels models

Sensor channel	**Fit type**	**Features**	**Number of features**
X Acceleration axis	2nd Gaussian	$A_1, \mu_1, \sigma_1, A_2, \mu_2, \sigma_2, b$	7
Y Acceleration axis	1st Gaussian	A_1, μ_1, σ_1, b	4
Y Rotation speed axis	1st Gaussian	A_1, μ_1, σ_1, b	4
Z Rotation speed axis	2nd Gaussian	$A_1, \mu_1, \sigma_1, A_2, \mu_2, \sigma_2, b$	7
β Euler angle	1st Gaussian	A_1, μ_1, σ_1, b	4
Total features			26

In total, 26 parameters define the model obtained. For dimensionality reduction purposes, Linear Discriminant Analysis (LDA) algorithm is used on the 93 recorded data set to obtain a linear classifier that is later used to predict goodness of swing motion..

Here you need to say about window approach and fitting more:

- How the data were synchronized?

- How window length were defined and why it is so and what the window length is?

- What is the domain knowledge and assumptions on the data (the noise of fitted parameters (this all A, mu, sigma, b) is gaussian distributed and has zero mean) - it is must have for LDA!

- And add as last point: We assume for sake of simplicity that the good and bad movements are linearly separable. In spite of this we use LDA.

2.2 Training and Simulationg Prediction

Here you need to explain how training and testing works. How works the window approach for simulation of real prediction conditions. Why and how we gave the labels to all 93 data sets.

2.2.1 Creating Prediction Framework

Please write the one i-th row of the LDA training matrix X:

x_i = [accX-Feautures, ….] - you know how to put it together

Then write the X matrix as the number of x's:

X = [x_0, …. x_n]^T, where n = 93

All n=93 data sets get their labels y = [y_0, …. y_1] where y_i = +/-1 for good and bad move accordingly. y is has dimension [1 x n]

Then you need to say about training and testing procedure:

1) X matrix with 93-fitted parameters (n=93) were splitter to 75% of data (#train = 0.75 * n) for training set X_train and 25% of data (#test = 0.25 * n) for test set X_test randomly.

2) Data were trained on X_train and tested on X_test with LDA.

Add information about precision of testing prediction. Saying that it newer falls under 80% is enough.

2.2.1 Simulating LDA

Here you need to add information about prediction and how it works.

For LDA it is just the dot product of wighting vector w with size [1 x n] and adding the bias b_0. With w and b_0 were trained in training part.

The simulation is based on window approach - how it works. (See simulatingAsInReal() - or something like this in python script). Just run it and you'll see. Main idea:

a) Taking one of 93 data sets without cutting it - let's call it D.

b) Moving window of length L=141 sampling point without overlap over the D and fitting (according to section 2.1).

c) Introducing threshold on a_x: a_x_threshold - for detection robustness.

d) Building the vector x (you should explain it in 2.2.1)

e) Providing the dot product of (y = w * x + b_0). The sign of y will give the prediction score +/-1. Looking at threshold: If a_x_threshold <= min(a_x) - the predicted score is correct, else predicted score is -1

Now the live swing detection can be explained from here.

2.2 Live Swing detection

Once the linear classifier is determined, it can be integrated in the program that collects, plots and fits live data coming from the racket as illustrated in figures 4 and 5.

A minimum Threshold value for the X acceleration axis (see figure 4) can be configured on the program's interface to launch the fitting algorithm once a motion has exceeded that threshold. The goal is to enhance performance via fitting only when a suspected swing motion is detected.

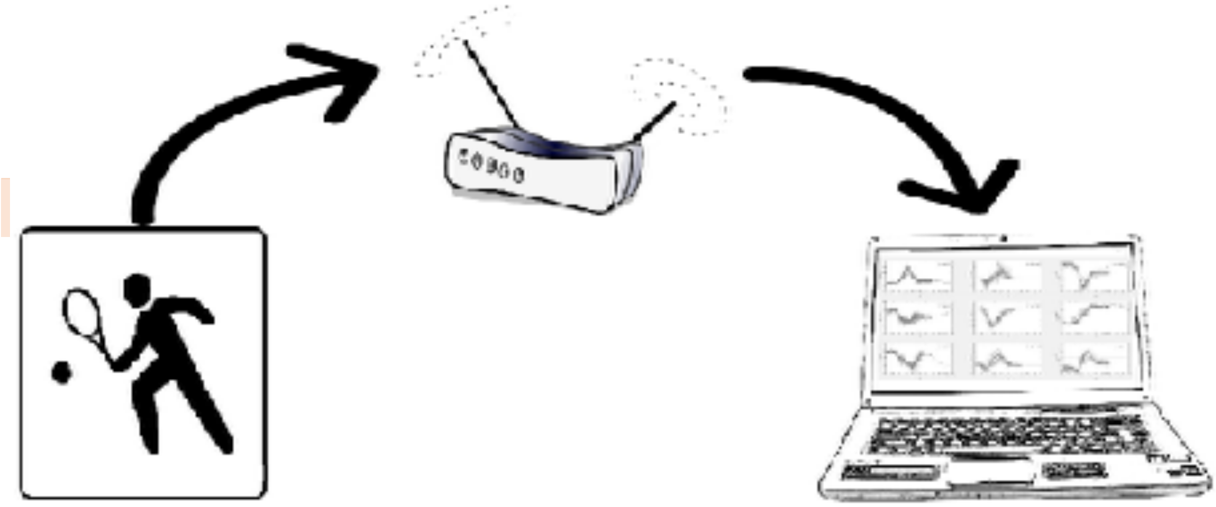
This part can be skipped if you explain 2.2.1 correctly. Just write her only one sentence: The live detection of a swing works like it is described in 2.2.1. The only one thing were added, were the adapting of fitting features live so that every window of length L=141 samples can be processed until new window arrives.

Once the fitting algorithm is launched, it buils a window where the minimum value lies in the middle with half of the fit window length (141 datapoints as seen in figure X) on each side: 70 points on each side of the minimum value. This approach ensures correct modelling of the motion recorded. If the fit modeled is multiplied by a cross product with the linier classifier already created, if it is classified as a swing motion, the fit is plotted in green, otherwise the fit is plotted in red.

3. Project Setup

The architecture model of the project is client-server as illustrated in the following figure:

Figure 2: Client-server architecture model approach used in the project



3.1. Hardware

An MPU-9150™ 9-axis motion tracking device from invensense® [Gyroscope / Compass / Accelerometer] is embedded in the racket on a 3D printed platform along with an Intel Edison serving as a sensor data broadcaster through a client application written in C++ as shown in the following figure:

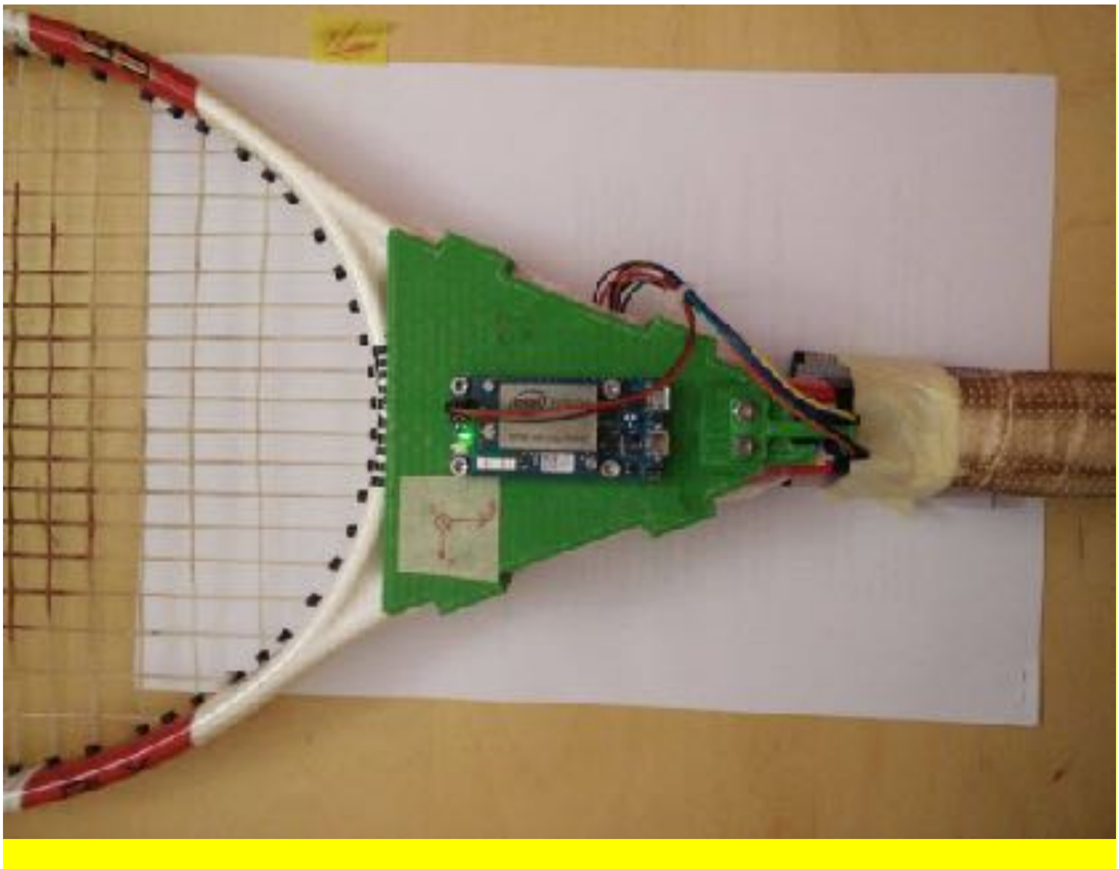


Figure 3: Racket hardware setup

The server side application is written in C++ running on QT. It may run on a Linux or Mac OS running machine with the following recommended requirements:

The script used to determine the swing model is written in C++. It may run on a Linux or Mac OS running machine

3.2. Software

Flowchart to be inserted:Suresh input.

4. Results

The following are screenshots of the server side application while running:

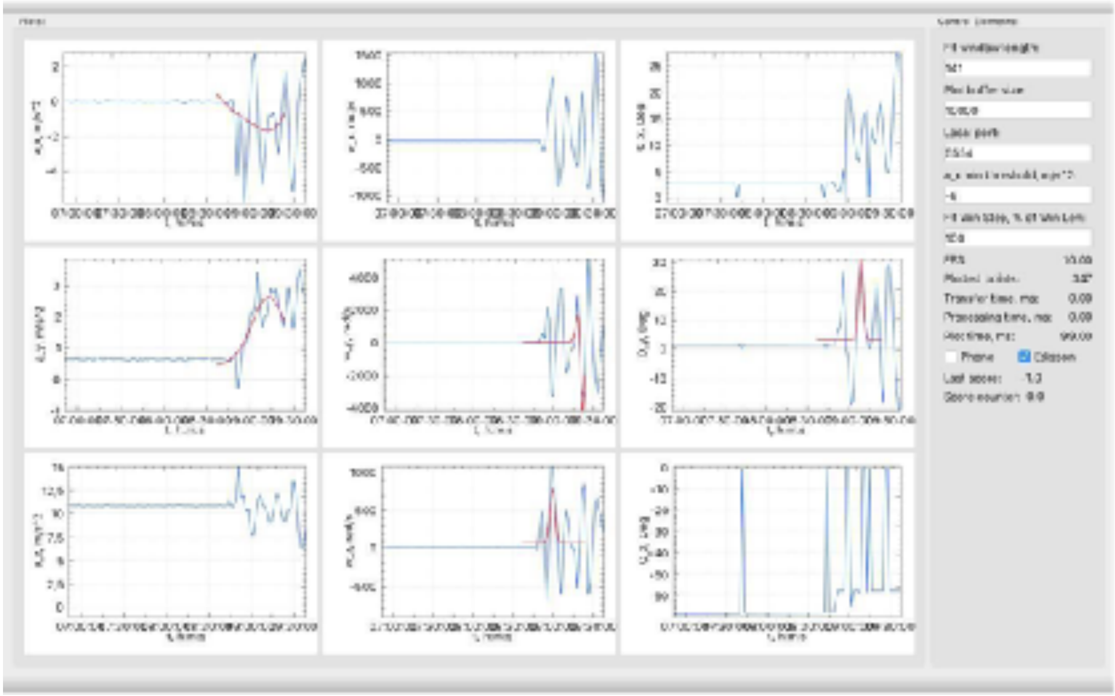


Figure 4: Data plotted and predicted correctly as a wrong swing move

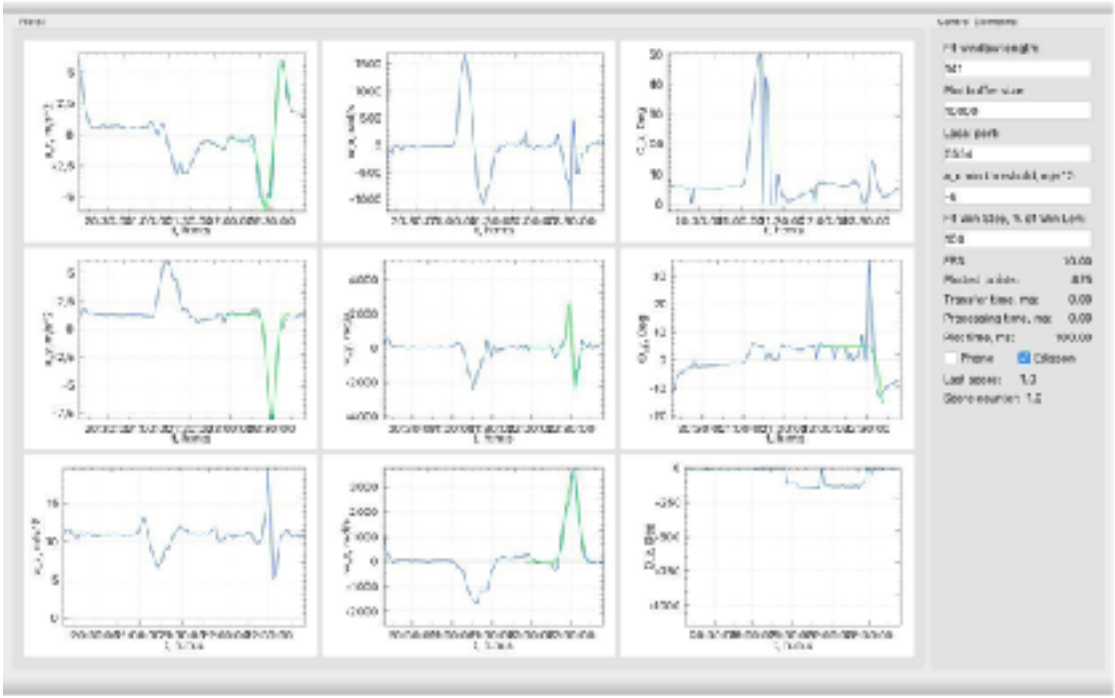


Figure 5: Data plotted and predicted correctly as a wrong swing move

Control elements are explained in the following table:

Table 2 Control elements of the server user interface

Control Element	Description
Fit window length	The number of data points of the desired window to be fitted. The value is defined manually from correct swing motion data
Plot buffer size	Determines how many points are plotted on the x axis of the plot.
Local port	The port number to listen to
a_x min threshold	The minimum Acceleration X axis threshold that once exceeded, the fitting algorithm is launched. (Explained in section 2.2)
Fit window step % of window length	….
Phone/Edissoon checkbutton	Determines whether the server is listening from a smartphone sensor broadcaster app or from the Intel Edisson

5. Conclusions

The challenge addressed in this project revolves around reducing the dimensionality of swing model defined: LDA algorithm is used to easily classify 26 features of gaussian distribution models representing 5 sensor data channels by creating a linear classifier used to seperate good swing motions from other motions.

The system can be improved through the following:

- Record more data for the training set for better swing prediction accuracy.
- Train a classifier with nonlinear characteristics.
- Better signal filtering.
- Train the system automatically while playing.
- Qualitify swing motions by comparing it to the training set.
- Provide live feedback along with game statistics and estimated calories burned through a smartwatch

6. References:

[1] Navsharan Singh. (2014, Juin 14). How do you put spin on a tennis ball? [Blog post]. Retrieved from http://www.quora.com/How-do-you-put-spin-on-a-tennis-ball

[2] Courtney Taylor. (2016, August 30). Formula for the Normal Distribution or Bell Curve. Retrieved from http://www.thoughtco.com/normal-distribution-bell-curve-formula-3126278