Real-time fitness monitor.

Mohannad Banayosi

May 15, 2013

Abstract

IMPACT is a multidisciplinary project in which students of computer science (more precisely: software engineering), material sciences, mechatronics and embedded system design come together to create Thai-boxing pads with builtin impact sensors and wireless connection to a base station. In this project, we will implement a multi-sensor tracking and analysis tool for a physical workout routine specific to Thai/kick-boxing or similar contact sports to track and display performance and fitness level of a practitioner over a single and over multiple sessions. It will be used to record and correlate the inputs from impact sensors and technique recognition and track improvemets of practitioners over time.

1 Background

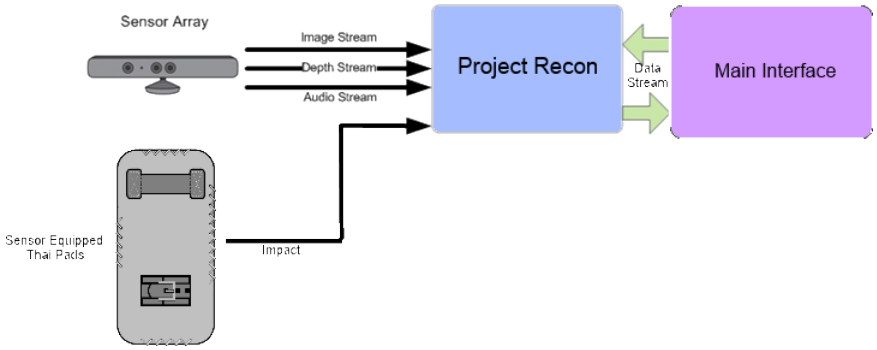

Figure 1: General Overview

1.1 Problem

In the world of martial arts, there is no accurate way to study and analyze the performance of the trainee. This absence of analysis is due to the lack of tools presented to the trainers.

1.2 Aim of this project

The aim of this project is to deliver an easy-to-use interface that displays training sessions' info and statistics. This in turn will help the trainers to get an accurate analysis of the trainee's performance.

1.3 Technologies and frameworks

A lot of technologies and frameworks have been researched, some of them were chosen to be used and others not. Following is a list of them. Most of the technologies researched where not used because of their violation of the core feature of the interface being generic.

1.3.1 Depthjs

Depthjs was developed by four MIT students, Aaron Zinman, Doug Fritz, Greg Elliott and Roy Shilkrot in 2010. Depthjs allows any web page to interact with the Microsoft Kinect using Javascript. It provides the low-level raw access to the Kinect as well as high-level hand gesture events to simplify development.
Depthjs is an open source project, but with no enough documentation, compatible with Google chrome only, and is more helpful in web navigation.

1.3.2 Zigfu

Zigfu was developed by Ted Blackman, Shlomo Zippel, Roee Shenberg, Amir Hirsch, Bryce Tucker. They developed the ZDK (Zigfu Development Kit) to make cross-platform, motion-controlled apps with Kinect in HTML5/Javascript, Unity3D and

Flash. Applications made with this development kit are portable across all operating systems, web browsers, computer vision middlewares, and 3D sensors.
Zigfu is not an open source project, still have problems in performance and accuracy in gesture recognition.

1.3.3 OpenDepth

OpenDepth - Extending the Web - is a client server application developed by Ecaterina Paun, Narcis Paun and Mircea Piturca. OpenDepth brings Kinect SDK methods and data to the web with the server side written on C# and built on top of Kinect SDK 1.5, while the client side is a written in JavaScript and uses the Three.js webGL rendering engine.
It is an open source project, with enough documentation, and is more helpful in web navigation.

1.3.4 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Developed by a fast-moving online-news operation, Django was designed to handle two challenges: the intensive deadlines of a newsroom and the stringent requirements of the experienced Web developers who wrote it. It lets you build high-performing, elegant Web applications quickly.

1.3.5 Web Sockets

WebSockets technology provides a new World Wide Web Consortium (W3C) JavaScript API and protocol for two-way communication over the Internet. This new protocol makes it easier to work directly with fixed data formats, and it bypasses the slower document-based HTTP protocol.

2 My approach

I will implement a multi-sensor tracking and analysis tool for a specific physical workout routine to track performance and fitness level of a practitioner over a single and over multiple sessions.

This interface should have the capability to get several plugins connected to it.

2.1 Architecture

The following architecture's design represents the different components included in the interface.

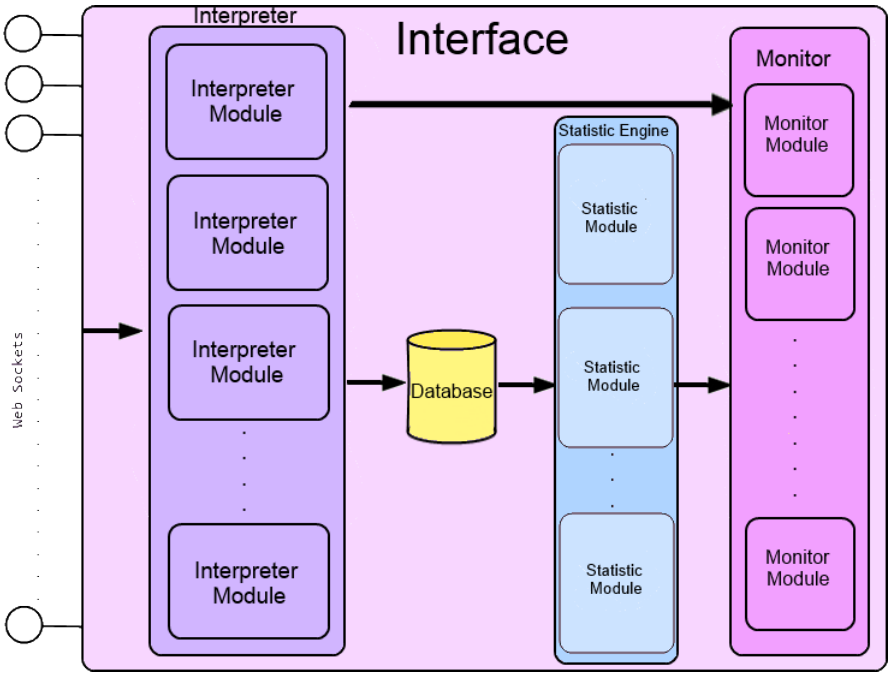

Figure 2: The Architecture

2.1.1 Interpreter

Containing several interpreter modules, the interpreter serves as the brain of the interface receiving the data (initially received through the sockets) and then begins digesting and translating it into actual info. It then sends it to the monitor (to display the data received) and the database manager (to save the data).

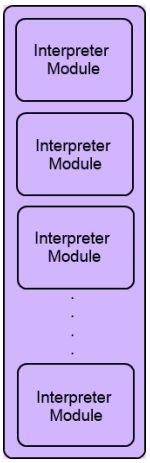

Figure 3: The Interpreter

2.1.2 Database engine

The database engine saves the users and their data; this data, or history, is collected from previous sessions the trainee took.

2.1.3 Statistics engine

Containing several statistics modules, this engine generates the analysis (this analysis depends on the current data received + the history of the trainee) and sends it to the monitor.

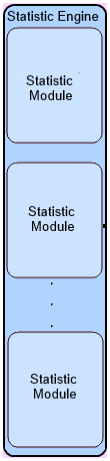

Figure 4: The Statistics Engine

2.1.4 Monitor

The monitor is the front end component of the interface that displays all the info (including live data and statistics-related data). It contains several monitoring modules together make up the monitor.

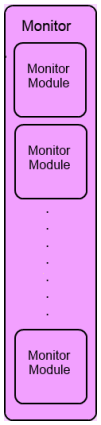

Figure 5: The Monitoring System

2.2 Hand shaking

This is the process where each plugin reserves a web socket to start the data stream with the interface. This begins when the plugin sends a request to the interface via a socket, then the interface reserves the socket for the plugin and sends an OK signal. Once the OK signal is received, the plugin can start the data stream at any time.

2.3 Data stream

The data stream contains the type of the data sent (on-session or off-session), a timestamp and the actual data.

2.4 The Interface

The application consists of three main features, live session monitoring, statistics and a small social part.

2.4.1 Live session monitoring

In this feature, the user creates a session (with a number of rounds, round duration and break duration) and then connects the sensor(s) needed and can do off-session measurements. Then the user starts the session and records the actions done.

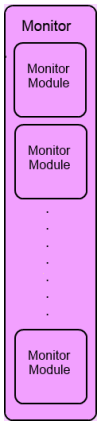

Figure 6: Live session monitoring

One of the biggest challenges faced during the implementation of this feature was that the rounds should be saved to the database very quickly that the user won't notice that their is something happening in the background of the application. A proposed solution was that the round is saved at the end of it, this caused a small lag in the timer (this was very noticeable in sessions that had a short break between the rounds). The solution implemented was creating and saving the rounds objects to the database while creating the session and then when something is needed to be updated in the round's data, it is saved during the round, therefore minimizing the number of queries executed during the session.

2.4.2 Statistics

2.4.3 Social