Activity Recognition – Natural Running Motion

|  |  |
| --- | --- |
| Helder Pestana  1191595 | Nuno David Santos  2009408 |

# ABSTRACT

In this paper we describe the idea and respective implementation of assignment 4 for Software Architecture for Users Interfaces class. This assignment is based in human activity recognition, the processing of information in order to make interferences about tasks or activities that are taking place.

# INTRODUCTION

After some lost ideas and a lot of discussion we found that running has a potential for activity recognition, but still was vague. There were a lot of potential ideas that can be chosen for developing an activity recognition application, but also almost everything is implemented for runners, mainly indoors equipment.

At first we thought about switching the physical sensors that are used in the indoor equipment for activity recognition sensors, but there was nothing innovative here. This led us to think in outside environment.

When jogging or running many people find it difficult to maintain a controlled rhythmic pace over time, majority tends to start too fast and never get to complete the full period they planned. So, this is a real issue and presents some challenges and innovations of the kind we are trying to achieve.

It went immediately through our mind that music could be the most interesting way to take control of the user’s pace and our first thought was to control by motivation letting the user choosing playlists for worst music and best music of his preference. Although users may be able to achieve the distance and time planned, for sure that doesn’t control our user’s pace.

After some research we found that music beat and rhythm takes effect on user pace, and this is it. This report will explain measures, studies, implementation and heuristic evaluation of using songs playlist to achieve user goals when running.

# Rhythm and beat

Beat is the basic unit of music and is what determines the speed of music which reflects how we keep in time when we dance or even when we tap the foot or fingers. The speed of music is indicated by a tempo mark above the first measure of the musical score, is measured in Beats per Minute and rarely changes during a song. It can be identified in many different ways from easy identifiable drums to the most subtle of pulses emanating throughout the music.

Rhythm is the variation of length and accentuation of a serious of sounds or others events and can be described as the flow of the music over the underlying beat. Is the part of the music that is emotive and most influences our response to the music.

Ideal running music should have many musical elements and should not be just high energy repetitive sounds because can become very boring after a short while. It should contain all the elements of well composed and well constructed music with a great variety enabling the runner to run as they want.

Many people are very disappointed when they take the favorite music out on a run. They are usually filled with excitement that their music will drive them better than ever before, but they find that is not seem to inspire them or help them as much as they expected. This is due to the fact that the beat and the rhythms are all wrong to the exercise.

# Linking BPM to running speed

We found a research already done by run2rhythm[1], so here follows the table with the results:

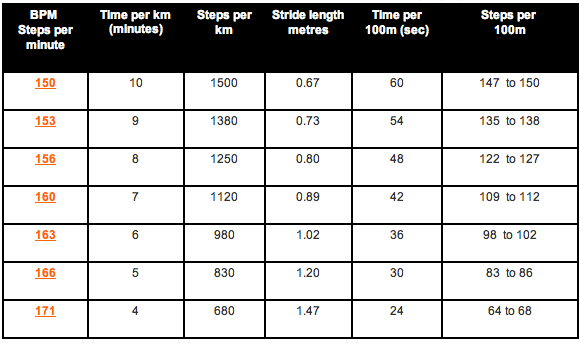


Table – Relationship between BPM and times per km

We can see that BPM are related with the steps per minute so we can easily relate BPM with distance. With this we can easily build a playlist when user inserts a route plan.

# Concept

Our concept is to connect your portable media center width some sensor in your body or accessories as Nike and iPod did[2].



Fig. – iPod and Nike personal trainner

This product provides time and distance control, voice feedback and all your favorite music.

Our idea is to use runner favorite music, analyzing the BPM of each one. So when runner inserts a plan (kilometers and time) the music will be chosen by the BPMs in order to achieve the runner goal.

The sensor will be monitoring the runner activity, so if the runner gets slow or faster the playlist will change in order to compensate the runner goal, i.e. If the runner slows down too much, the playlist will pick music with higher BPM and vice versa.

# Main Goals

We didn’t have the opportunity or time to build all the desired concept so our main goals were be to received data from an accelerometer of the wii devices (wiimote and nunchuk), analyze the received data to compare with step time and distance reference we got from the run2rhythm table, in order to achieve consistency.

Another goal we had was to try and get the BPM from the songs in order to be able to build a playlist that respects the runner’s plan. After the playlist is done we had to be able to change it in accordance with the data received from the wiimote accelerometer.

We had to make a full heuristhic evaluation of our concept, validating the main idea. Are the BPM really related with the steps per minute?

# Implementation

The first challenge we faced was to use the wiimote to measure the steps per minute and the relationship between them and the beats per minute of the music they were listening to while running.

In order to do that, we had to connect to the wiimote and acquire the data from the accelerometer in order to measure the movements of the runner.

This quest lead us into a series of interaction library’s in Java which we ended up not using because of a few problems encountered while trying to connect to it to our systems (Mac’s running windows).

After these small problems, we proceeded the study of the different libraries available and ended up choosing a library (WiimoteLib) developed in the Microsoft .NET framework, in c# language, which we chose as our main language to implement various phases of the project.

At first we encountered a few problems to connect the wii to the system because of a sync problem that we were not aware of and that caused us to lose a lot of time trying to figure out why couldn’t we connect the wiimote to the working software even after it was added correctly to the operating system.

After some search we found out that the operation to add the wiimote to the system had one more step in order for it to be able to connect to our library.

The first application we built had the solo goal of acquiring presenting and registering the data sent via Bluetooth from the wii devices to the computer.

The project is divided in a set of projects where NRM.OO represent the libraries of domain objects, NRM is the Windows forms application that handles the different interactions with the user: acquiring Data from the wii; handling the data from the user playlist distance of the next run and time to run that distance, and NRM.Analytics represent the library responsible for determining for handling the data handed by the user and building the playlist.

In the NTM.OO library we built a set of classes to represent the data we wanted to handle and that is represented in the diagram bellow.



Fig. – Class diagram

The class WiiData is the class built to represent a single record of data from the accelerometer, where we record acceleration in three axys (x, y, z).

WiiDataColl represents a collection of WiiData objects.

BPMInterval is a class meant to represent an interval of BPM’s (beats per minute).

SongData is a class that represents a song in a playlist characterized by the full name, length and BPM.

SonDataColl represents a collection of SongData objects (Playlist).

BPM’s

The second part of the project was to implement the construction of a playlist based on the course distance the user wanted to do and the time to accomplish that run.

This task revealed itself more challenging than we expected once the availability of libraries to make this analysis is scarce and have a few details that handicapped our initial expectations. Nonetheless we strived to accomplish this objective and build a playlist based on the users playlist instead of the currently marketed options where u have to buy a predefined set of sounds, this way you could still make use of the music to “keep your pace” but use your own set of songs to be chosen from.

In order to achieve this task we studied a library “Adion’s BPM Detection DLL” that allowed us to calculate the BPM’s of a music.

This library even though it’s shareware allowed us to open the mp3 files and determine their BPMs building this way the users songs playlist we aimed for.

This library allows you to open a file and determine the BPM’s while reading it and inferring it with an accuracy property. This allowed us to determine the precision we needed in order to build the global playlists BPMs wh ich would later be the critical data used to build the course motivating playlist☺.

# Data Analysis

To retrieve data for further analysis we did two different researches. The first one was running indoors in a treadmill, with the goal to obtain data from the Wii devices. The second research was running outdoors in order to link music beats with running steps.

## Indoor Research

For this research we went to a gymnasium and attached the wiimote to our foot using a roller bandage. The next pictures illustrate how we implemented our indoor research:



Fig. – Installing wiimote for data retrieve



Fig. – Retrieving data with wiimote

With all devices set we started acquiring data. We picked data running at different speeds with data being acquired during one minute. The captured data from the Wii devices is related with 3 axis (x, y, z) and with this information we made a new line showing the steps as seen in the charts.

The graph with the acquired data follows:

* 6 km/h

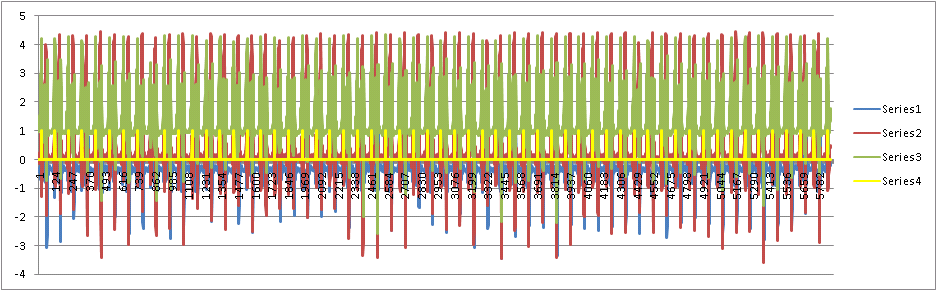


Chart – Data at 6 km/h

At this speed we’ve counted 118 SPM (steps per minute) and the Wiimote has counted 116 SPM.

* 7,5 km/h

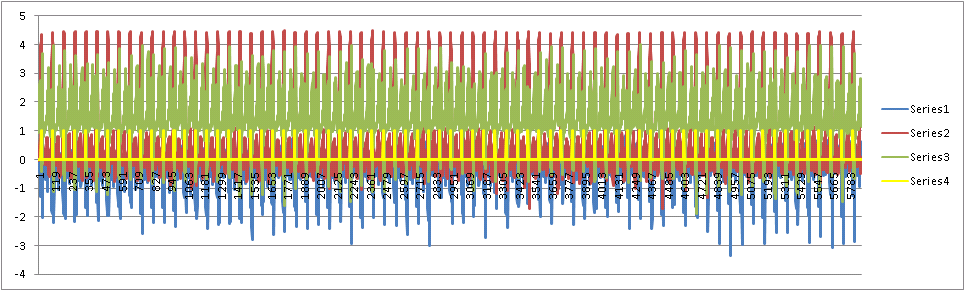


Chart – Data at 7.5 km/h

The number of steps counted raised to 135 but wiimote gives us a real different number: 150

* 9 km/h

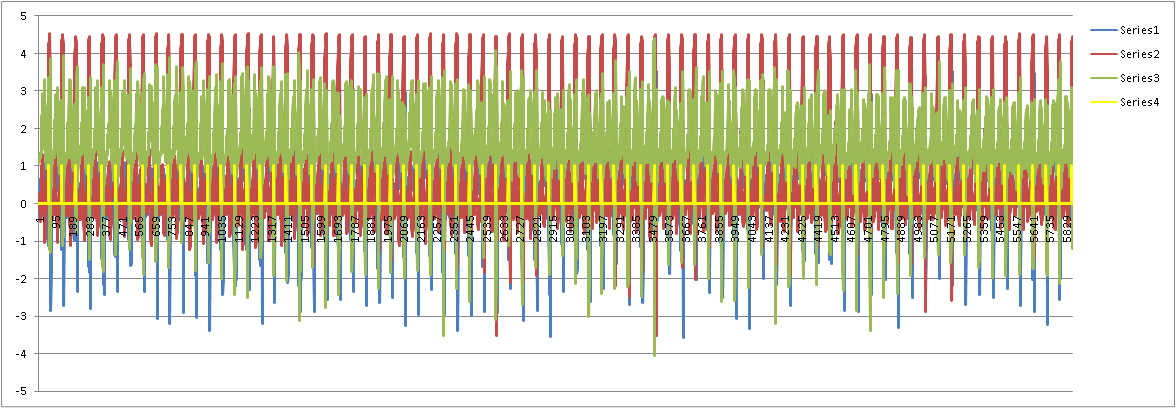


Chart – Data at 9 km/h

At this speed we also obtain a relative difference between the SPM counted by us 141 and the Wiimote SPM 156

* 10,5 km/h

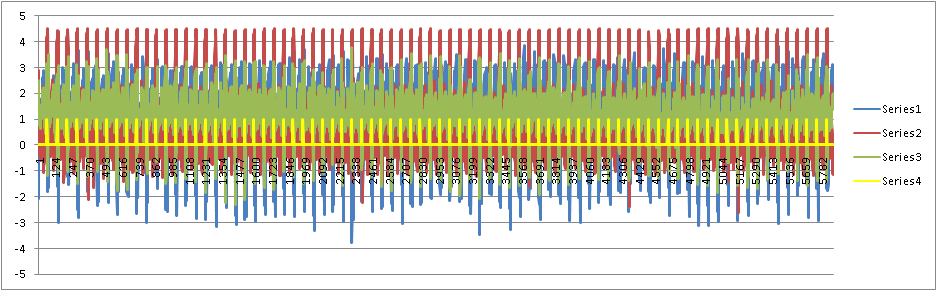


Chart – Data at 10,5 km/h

At this time our values became proximate 158 and 160, our count and Wiimote values respectively.

Making the calculus in order to compare with the reference table we have built a similar table with our data:

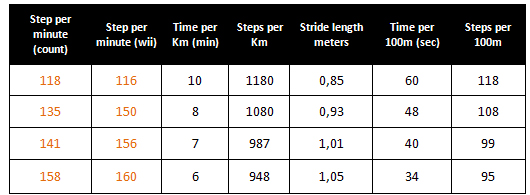


Table – Indoors research data

For the calculus we have used our steps count instead the wiimote. The reason for this is because the reference table was built also in human step count without any sensor step detection.

The values are similar to the reference table, so we can use this kind of table to associate the BPM with the SPT and with the right calculations we can build the playlist.

## Outdoor Research

The goal of the outdoor research was to find how the music that you hear can influence the number of steps that you take while running.

We’ve picked a random album and without measuring the BPMs we went out to run somewhere. The chosen spot wasn’t perfect once it has some climbs and downs and also many turns. In the future some extra measurements should be done in an athletics track.

The data was acquired from different songs and when the it was approximately at the middle. The next table shows the retrieved data:

|  |  |  |
| --- | --- | --- |
| **Music** | **Beats per Minute** | **Steps per Minute** |
| Sample1 | 140 | 134 |
| Sample2 | 117 | 124 |
| Sample3 | 160 | 136 |
| Sample4 | 100 | 122 |
| Sample5 | 140 | 134 |
| Sample6 | 149 | 152 |
| Sample7 | 140 | 136 |

Table – Outdoor research data

The samples are from the Sound in Light album from Blasted Mechanism and correspond to the 2nd till 8th music.

We notice that values don’t correspond exactly. This can be caused for the difficult to run and count at same time, also the spot chosen.

Looking at the data we can easily explain the big difference at sample 4, because if you have less than 122 SPM you’re not running.

In conclusion, the values from this research are not perfectly accurate as we wished but applying a correlation to the values we’ve got 0.77, which indicates that there is a strong relation between the values acquired.

We can say that BPMs are related with SPMs and if you can associate distance to the SPMs you can associate music beats to distance.

# Heuristic evaluation

We didn’t accomplish interview system due to the implementation constrains and the equipment available, but we have followed the heuristics as published in Nielsen’s book [3]:

* Visibility of the system status

This is a really small user interface application, user only needs to insert distance and time and almost immediately is created a playlist that will drive the user to accomplish his task. So, the visibility of the system status is maintained like an ordinary portable media center.

* Match between system and the real world

As said before, our system will be based on the portable media center. So after user selects a plan it will behave like that with the usually pause, stop, play, forward, rewind, volume control, etc.

* User control and freedom

As far as this is concerned the user has the freedom to do what he is used to do. So if he doesn’t like the current music he can skip to the next one or if he wants to hear the same music again he can just rewind. The application itself has to determine if the playlist has to be changed in according to the user actions.

* Consistency and standards

Portable media centers are very common nowadays, so it has also standard’s that work seamlessly between them as far as sound file types and playlist are concerned.

* Error prevention

We can avoid users to insert a runner plan if there is not enough songs to fulfill all plan, but we can skip this error by repeating the songs that users have in that moment on the portable media center. If there is no music files, the user won’t be able to play an empty playlist, but this is also something that is implemented in common portables media center.

* Recognition rather than recall

Our application only interacts with the user when inserting time and distance. This fields are really identified and users use recognition to insert the data correctly. All other features are using system itself: playlists, music control, sound control, etc.

* Flexibility and efficiency of use

One way of providing efficiency is reducing the number of possible functions. For example, if user wants to change the plan the application only needs to go to the home screen. Here users insert new data and the procedure is equal to what he has done before.

* Aesthetic and minimalist design

The user interface is so small that we almost can skip this step. It is based in two fields and one button: time, distance and ok button. During the runner the playlist is changes depending on the goal achievement. This runs in background and user doesn’t realize if the playlist was changed or not.

# Constrains

The main constrains encountered in the development of this project where mainly the difficulty in the interaction with the API’s and Wii devices. As said before, these problems constrained us from developing a number of other features and validations in order to have a more robust system.

The Wiimote connection toke us a lot of time, and then the beats per minute measuring also ended up being really tough to accomplish.

We've tried a lot of libraries and we opt for one that is far from perfect.

With time we’re sure a better approach would be to do a FFT analysis and implement our own beat measure library.

Last but not least we got stuck with the media library. This Library needs windows media player 10 installed. Once that we had the windows media library 9 it didn’t show any error, it simply don't work.

# Future

The future steps should be interacting with a real media center that can be connected to a device like the one Nike and Apple developed for the IPod[2].

In order to achieve this we think that the IPhone should be the elected device to implement it once it has an SDK that allows you to interact with their system, and there is the precedent Nike device that already communicates with an apple device and the system probably are compatible.

Another approach is to implement this concept for other activities like for house cleaning or even on Job work. An example can be linking music beats with the typing speed. We can apply similar research methods but instead of using an accelerometer we can measure time between key presses.

# Conclusion

This work has covered a number of areas from the communication with Bluetooth devices to the evaluation of acceleration diagrams, and of course passing by activity recognition.

All the areas were interestingly challenging, but what has really stuck till the end, was the interesting way in which you might influence a wide range of everyday activities with a simple ordinary everyday thing like sound.

We’ve come to conclude that the use of a system like this one will bring true benefits to your everyday activities in ways such as the following:

* Cadence control
* Bounce in runner step
* Avoiding injuries and illness
* Helping the breathing pattern
* Keeping cadence correct and reinforcement the aspect of ideal running/activity
* Better capillary blood flow
* Less muscle fatigue
* Makes you feel good ☺

We think this was a great idea and could end up being a great application. We fell really sorry for not achieved all the implementation that we wanted.

In resume, we have made all the really important stuff: receiving data from the Wiimote, measuring the music beats and music length and building a playlist but the tight schedule didn't allow us to combine all the functionality and provide a complete implementation.

# REFERENCES

1. Run2rhythm

<http://www.run2r.com>

1. Nike and iPod

<http://www.apple.com/ipod/nike/run.html>

1. BPM Detection

<http://www.mmartins.com/mmartins/bpmdetection/bpmdetection.asp>

1. Adion’s BPM Detection Library

<http://adionsoft.net/bpm>