Real-Time Human Activity Recognition in Augmented Reality Games

Ryann Sullivan
Supervised by Gleb Beliakov & Tim Wilkin
Deakin University

Abstract

As mobile phones have become commonplace in our society and their capabilities continue to grow, research into activity recognition using these devices has also grown rapidly this millennium. The purpose of this research is by using a series of averaging techniques; identify which techniques (if any) are effective at enabling the real time analysis of data collected from a mobile phone accelerometer. Through this research it was found that some techniques are very similar in their robustness to noise. It was also found that some of the computationally simpler averaging techniques performed just as well as the more complex averaging techniques. This has important implications when analysing data in real time.

Introduction

As technology has advanced and mobile phones have become more and more powerful the opportunity for research into activity recognition using mobile devices has steadily grown throughout the last decade. The basis of this research is to investigate the effectiveness of using various averaging techniques to enable real time analysis of data collected by accelerometer sensors found in many mobile phones today. In this research the technique has been applied to the problem of trying to identify certain physical behaviours (such as skipping, running, jumping etc.) for use in an augmented reality mobile game. This research was part of a larger cross-disciplinary research team investigating the use of augmented reality games on smartphones and tablets, as a means of engaging children in physical exercise at school.

In the past decade there has been a lot of research in the field of activity recognition with regards to mobile devices. Much of this research has been directed to the field of healthcare and using activity recognition as a means of tracking elderly or injured patients. For example sending an alert message to staff in a nursing home when someone falls down or to track an injured athlete in a rehab facility to ensure they are getting the correct amount of exercise to make an optimal recovery. There have been

however limitations to this research as the movements that are being tracked for these purposes are often "non-rapid" movements such as walking or sitting. This creates a very different data set to data gathered for example from a skipping child. In previous research studies, sensors have often been mounted on the participant to ensure that the recorded data remains in a fixed frame of reference relative to the participant however for this research one of the fundamental aims was for participants not to be restricted by having to wear the sensor but instead be able to hold it freely in the hand. Obviously in the context of children's movement being tracked for an augmented reality game the types of movements we expect to see are vastly different from those exhibited by patients in a retirement home.

This necessitates new research to be done to investigate these more "rapid" movements and a new way of processing data. The processing method needs to be sensitive to the fact that the data is more erratic as the movements of children running around are much less fluid than the movements of retirees standing up and the data is noisier due to the fact that the sensor is not worn by the participant and therefore rotations and translations of the phone relative to the participant will corrupt signals. Another problem that needed to be overcome in the research was that the sensors in most mobile phones do not sample the environment at regular intervals and therefore many statistical techniques become considerably harder to implement and make the processing of the data more cumbersome.

Research in this field is relevant now more than ever in Australia as the obesity rates in children and teenagers are rising and this research will contribute to trying to get children in schools to be more physically active whilst keeping them entertained playing an augmented reality game.

Method

The research began by becoming familiar with the phone and how it collected data using the custom application loaded onto the device. Then some test data was collected using the phone and the grammatical structure of the data strings was analysed. An example data string is shown below in Figure 1. (NOTE: In the phone's output file the entire data string was stored in one line)

```
20140117144627495:Sensor:SensorsNO:
acc(0.031158447,-0.17416382,11.066971)
mag(23.878479,4.4387817,159.8999)
bearing(-1.2822051)
gyro(1.373291E-4,0.003036499,-1.8310547E-4)
light(11.0)
pressure(995.6727)
proximity(5.000305)
grav(0.003554366,-0.15101178,9.805487)
linacc(0.01807214,5.0239265E-4,0.0038967133)
rotvec(-0.0057282923,-0.0051482692,0.6507431,0.0,0.0)
humidity(0.0)
ambienttemp(0.0)
```

After analysing the structure of the programs output, a program was written in MATLAB to extract the useful information from the file and store each relevant segment of data in an appropriate variable. The next step was to identify exactly what each of the relevant data segments represented. This was achieved by researching online the phone's API (Application Programming Interface) and how its sensors worked. The relevant data and descriptions are shown below in Figure 2.

Data Segment	Description
20140117144627495	A timestamp recording the date/time that the record was taken. Formatted as YYYYMMDDHHMMSSSSS
acc(0.031158447,-0.17416382,11.066971)	The acceleration in (x,y,z) axis of the phone respectively. Measured in ms^{-2}.
mag(23.878479,4.4387817,159.8999)	Magnetometer readings in (x,y,z) axis of the phone respectively. Measured in T.
bearing(-1.2822051)	The radial bearing of the phone from true North.
gyro(1.373291E-4,0.003036499,-1.8310547E-4)	The gyroscopic rotations in (x,y,z) axis of the phone respectively. Measured in inertial units.
grav(0.003554366,-0.15101178,9.805487)	The acceleration due to gravity along the (x,y,z) axis of the phone respectively. Measured in ms^{-2}.
rotvec(-0.0057282923,-0.0051482692,0.6507431,0.0,0.0)	The rotation in radians around the (x,y,z) axis of the phone respectively. Rotation is measured by the change in orientation of the phone from the last sample to the current sample.

The next step was to convert the acceleration vectors into a single initial frame of reference. This was done by using the rotation vector data to create a set of Tait-Bryan angles and apply them to the data at each current sample and every other sample after it. This in turn would step by step rotate the vectors back into their initial frame of reference. Once the acceleration vectors were all in a single frame of reference they had to be integrated to retrieve the velocity and position vectors to enable the reconstruction of a trajectory space. Unfortunately this proved too difficult to perform as the noise in the data was simply too much for the algorithms in MATLAB to overcome to give any meaningful output. After repeated various attempts to create a trajectory space representation the method was abandoned and instead the focus shifted to trying to implement a feature space representation of the various activities. A sample of each activity was recorded (hopping, jumping, skipping, running and walking). 10 feature space variables were decided upon to represent each activity. They were…

$$Given\ x = x_{\nearrow}\ and\ size(x) = n$$

1. **Median**

$$M = x_{\frac{n+1}{2}}\ if\ n\ is\ odd$$

$$M = \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2}\ if\ n\ is\ even$$

2. **Arithmetic Mean**

$$A = \frac{1}{n}\sum_{i=1}^{n} x_i$$

3. **Geometric Mean**

$$G = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}}$$

4. **Harmonic Mean**

$$H = \frac{n}{\sum_{i=1}^{n}\frac{1}{x_i}}$$

5. **Least Trimmed Squares (Penalty Form)**

$$LTS = \arg\min_{y} \sum_{i=1}^{\frac{n+1}{2}} (x_i - y)^2$$

6. **Least Median Squares (Penalty Form)**

$$LMS = \arg\min_{y} M(x_i - y)^2$$

7. **Shorth (Penalty Form)**

$$S = \arg\min_{y} \sum_{i=1}^{\frac{n+1}{2}} (x_i - y)^2$$

8. **Root Mean Squared**

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$$

9. **Standard Deviation**

$$SD = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (x_i - u)^2} \ where\ u = \frac{1}{n}\sum_{i=1}^{n} x_i$$

10. **Power Spectral Density**

$$P = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} x(t)^2 dt \ where\ T \in t$$

After identifying the 10 feature space variables to be used a program was written to take in a vector of acceleration values, a vector of time values (that match the acceleration values), a window size and a window offset. The program would take the acceleration values for a certain window of time and calculate the feature space variables for the window before moving the window along the sample by the window offset and then taking the acceleration values corresponding to that window and recalculating the feature space variables and so on until the window reaches the end of the window. Once the feature space variables were computed for each window the corresponding x, y and z components of the variables were combined to calculate the overall magnitude of the averaged acceleration over the window. This was used to demonstrate how resistant to noise the various averaging techniques were. The 10 feature space variables were then plotted against their corresponding window index number to show how the average value changed over the length of the sample.

Results

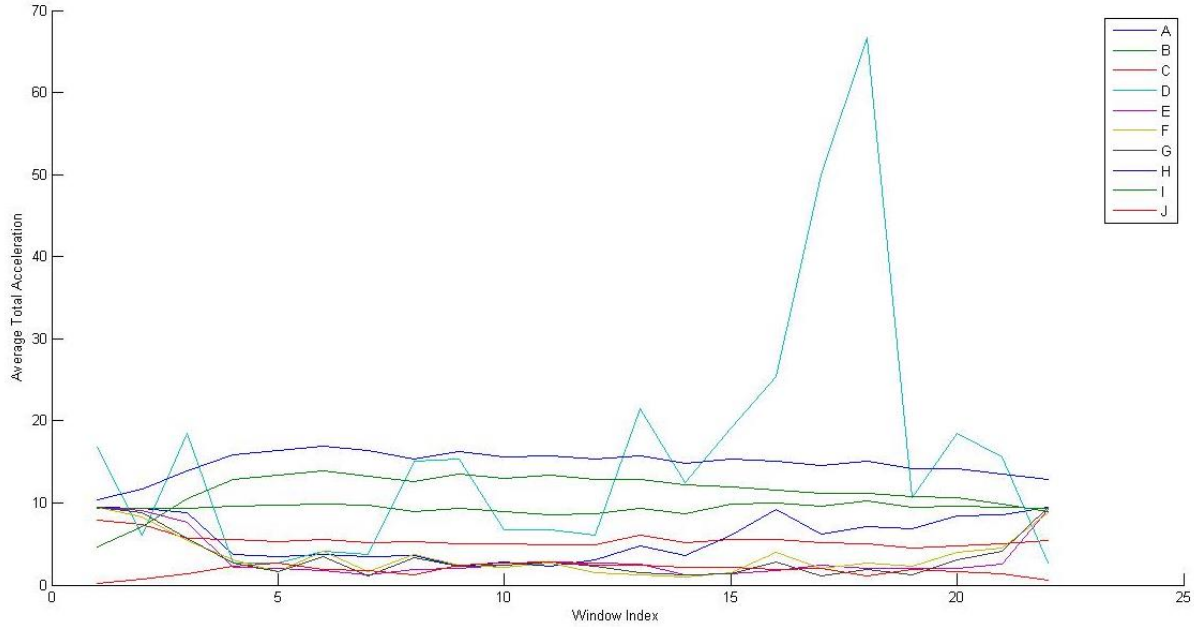

Figure 1 – Hopping

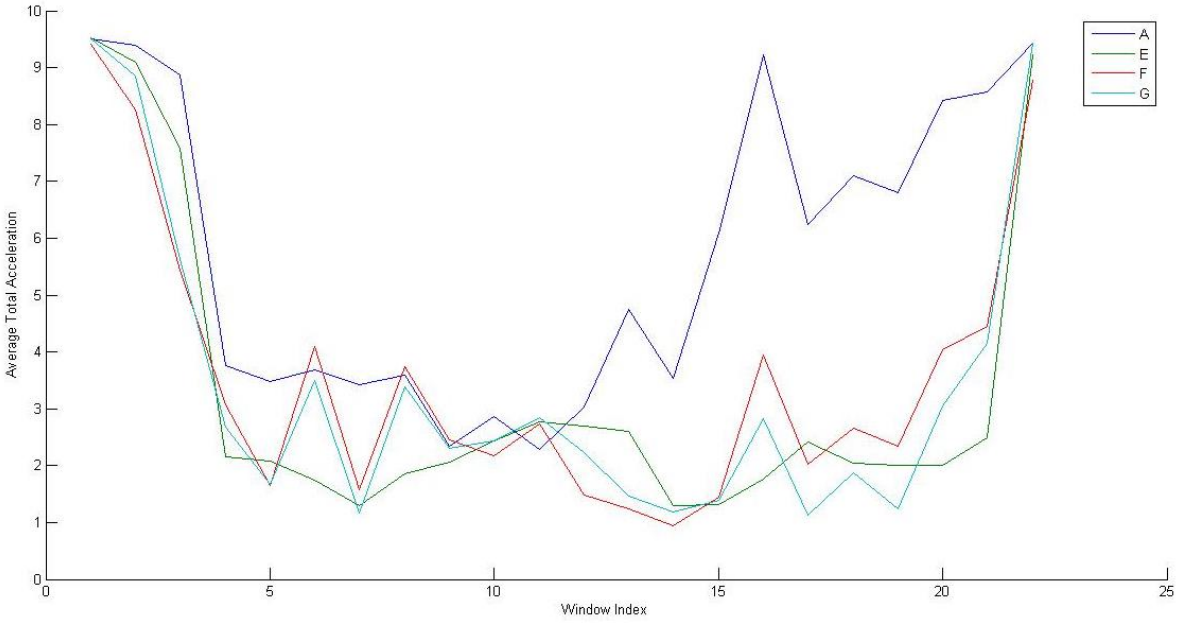

Figure 2 – Detailed Hopping

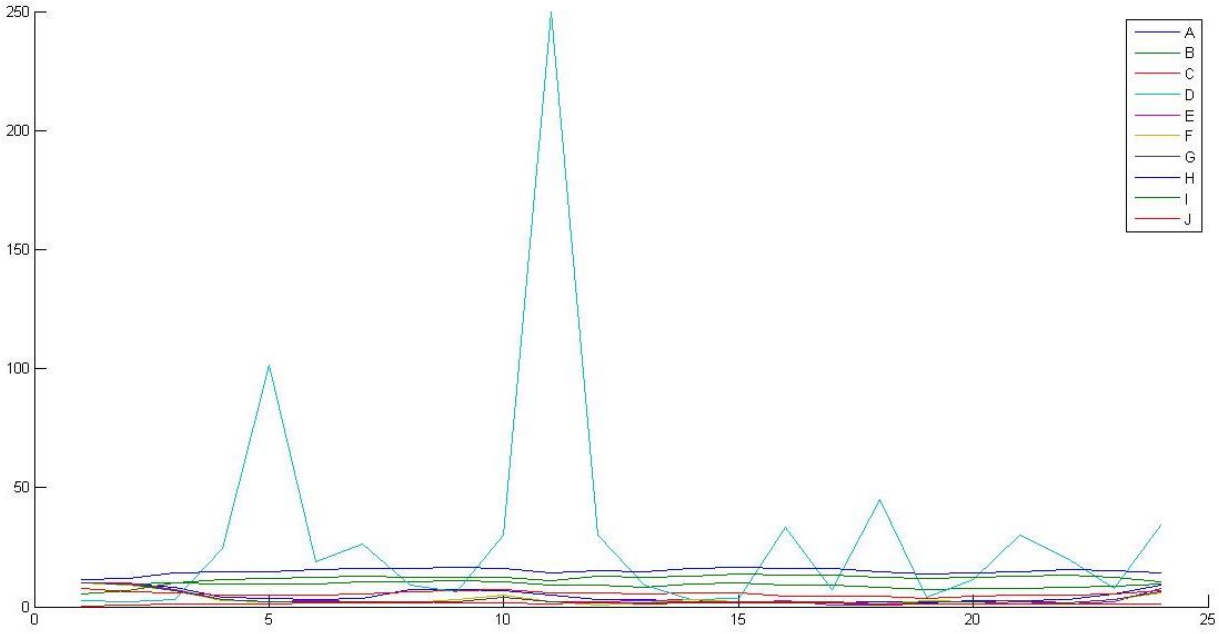

Figure 3 – Jumping

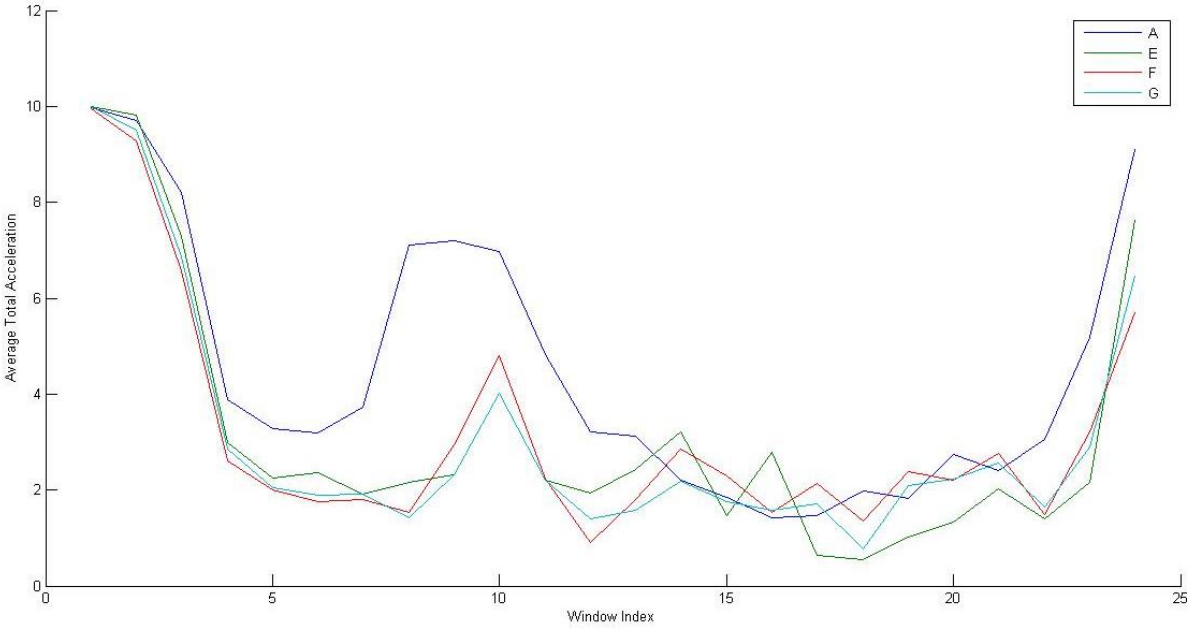

Figure 4 – Detailed Jumping

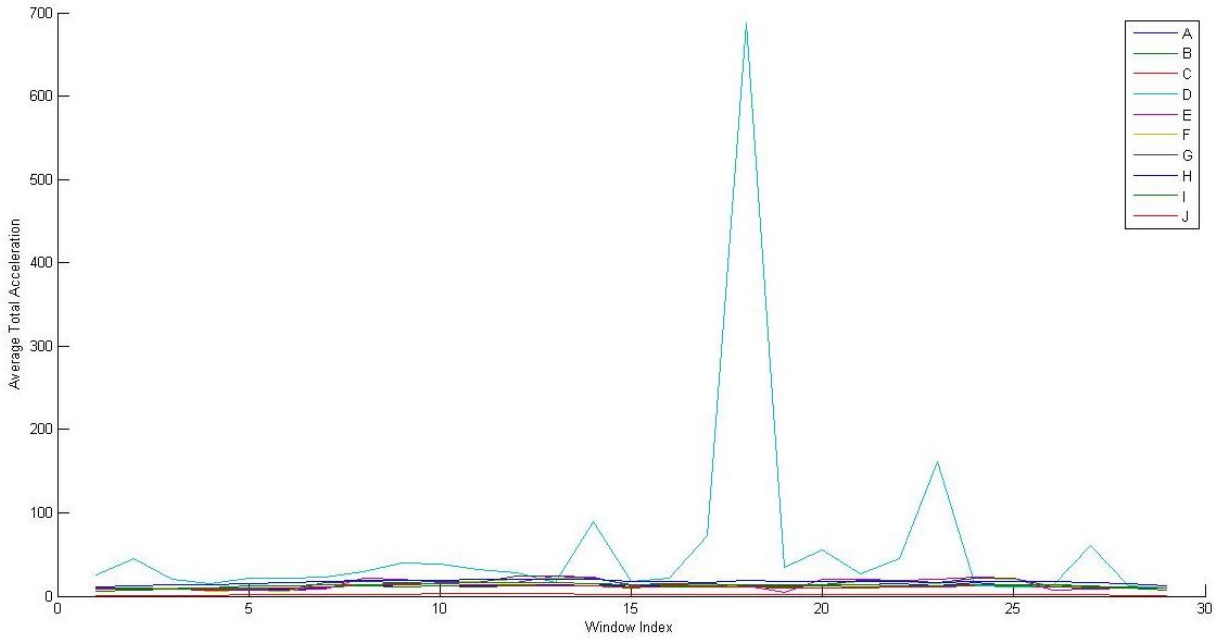

Figure 5 – Running

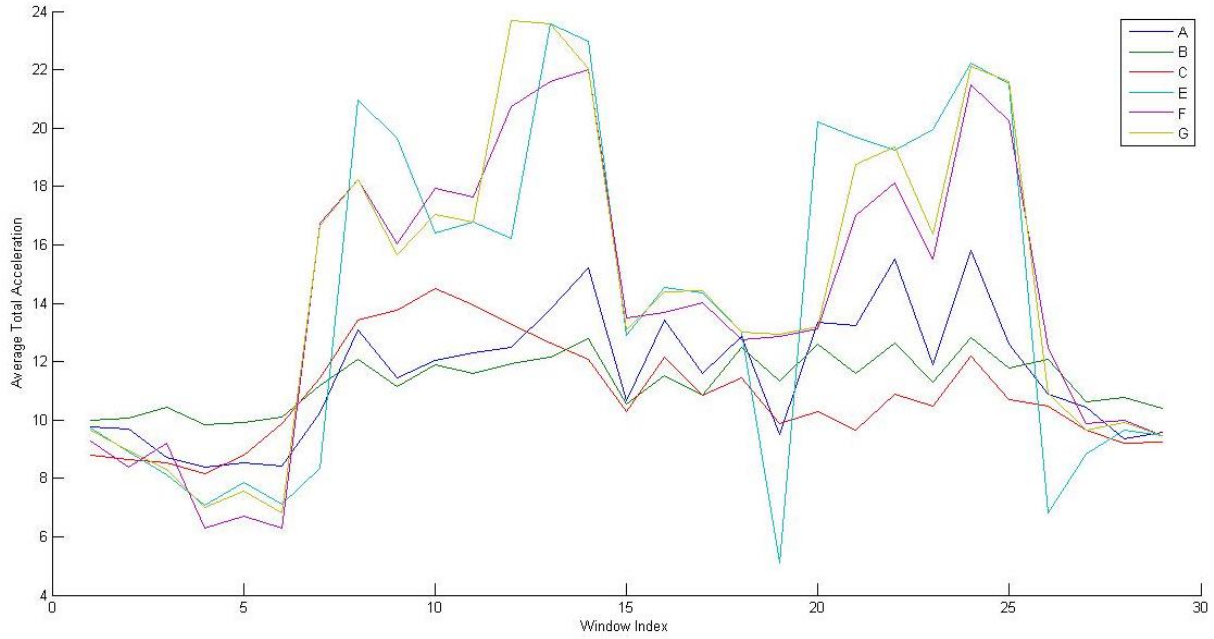

Figure 6 – Detailed Running

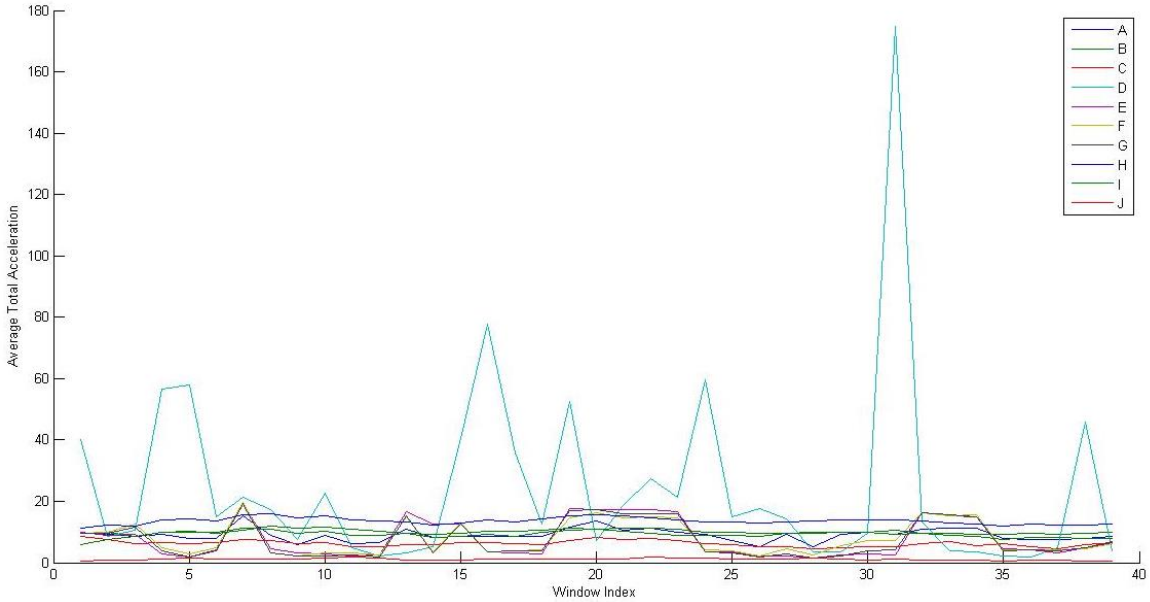

Figure 7 – Skipping

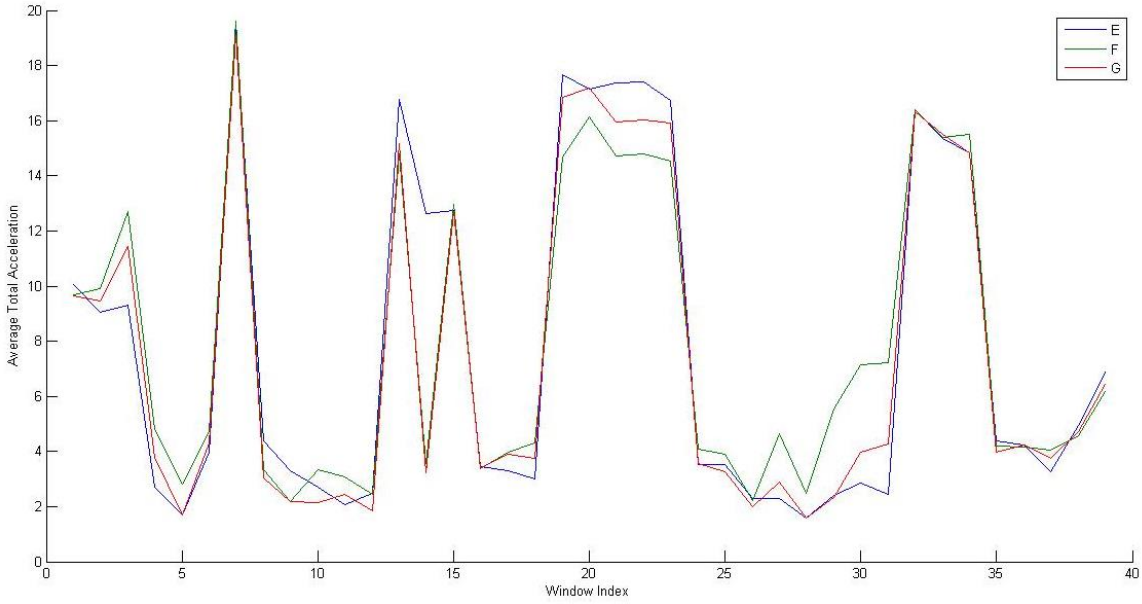

Figure 8 – Detailed Skipping

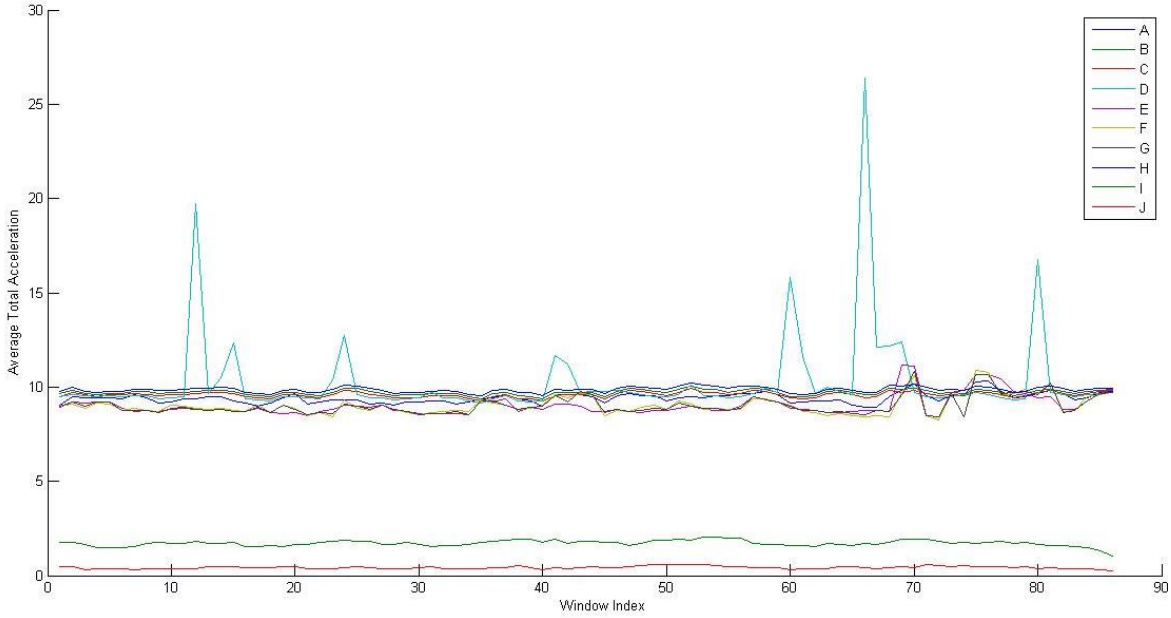

Figure 9 – Walking

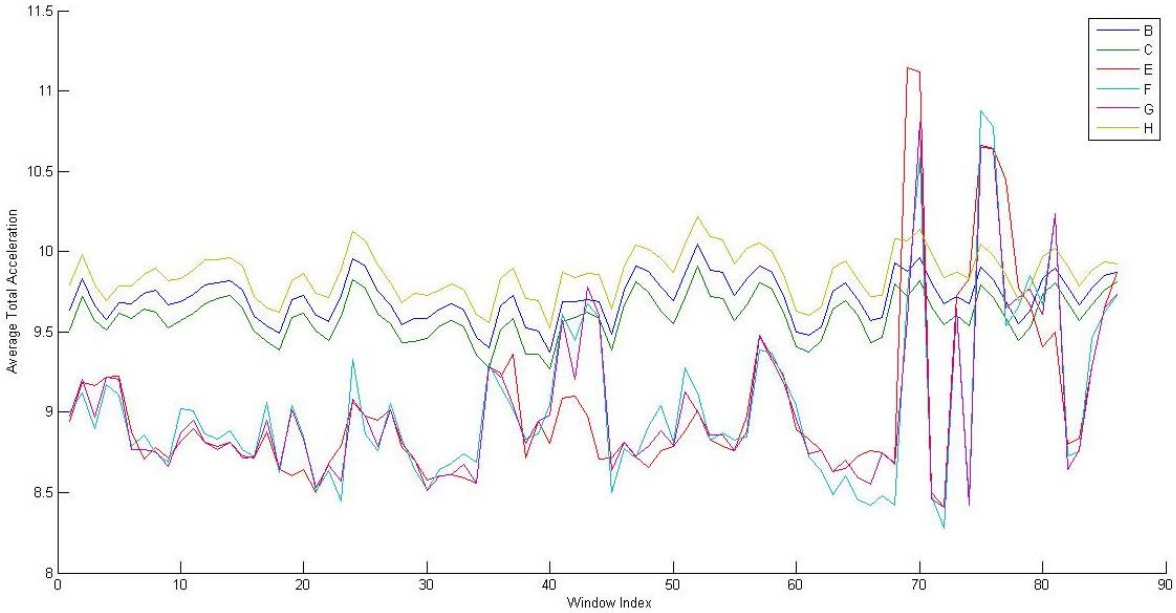

Figure 10 – Detailed Walking

Discussion

For the graphs above…

- A = Median
- B = Arithmetic Mean
- C = Geometric Mean
- D = Harmonic Mean
- E = Least Trimmed Squares
- F = Least Median of Squares
- G = Shorth
- H = Root Mean Squared
- I = Standard Deviation
- J = Power Spectral Density

The results of the research present a few interesting features. It can be clearly seen from Figures 1,3,5,7 & 9 that the magnitude of the harmonic mean varies greatly throughout the samples. We can infer from this that the harmonic mean is greatly affected by noise in the signals. It is also seen across all figures that the Least Trimmed Squares, Least Median of Squares and the Shorth all give very similar results over the sliding windows. The median also gave similar results to the LTS, LMS and Shorth for the 5 activities. In Figure 6 the arithmetic mean and geometric mean both showed similar results to the median and in Figure 10 they behaved similarly to the root mean squared. As the LTS, LMS and Shorth all gave relatively similar results we can assume that they are the most resistant to noise as this is more probable than them all being equally corrupted by noise in the signals. However due to its computational efficiency along with its relative resistance to noise the median may be the best feature variable to use.

Conclusion

The purpose of this research was to investigate how effective various averaging techniques were at analysing data in real time that was collected by an accelerometer in a mobile phone. Previous research has been limited in this field as movements being tracked have been low energy movements such as walking, sitting and standing. Also in prior research sensors have often had to be worn by the participant making the gathering of data more intrusive and lacking the framework to manipulate the users data in real time. It is for these reasons that new research was to be done to investigate higher energy movements such as running and jumping. The results showed that using the harmonic mean of a signal is not a valid averaging technique for activity recognition as it is too heavily affected by noise in signals. It was also found that the Least Trimmed Squares, Least Median of Squares & Shorth all gave similar results meaning that two of these features would most likely be made redundant in future calculations. The arithmetic mean seemed to be the best averaging technique to use for activity recognition as it is fairly resistant to noise and is very computationally cheap to implement which is important when data is being analysed in real time. Further research is required in this field to fully realise whether these averaging techniques are a valid method for performing activity recognition in real time.

References

- Beliakov G, Pradera A, Calvo T, 2007, *Aggregation Functions: A Guide for Practitioners*, Springer
- Hanselman D, Littlefield B, 1998, *Mastering MATLAB® 5: A Comprehensive Tutorial and Reference*, Prentice-Hall International Inc.
- Android, 2014, *SensorEvent | Android Developers*, 3/1/14, <http://developer.android.com/reference/android/hardware/SensorEvent.html>
- Brezmes T, Gorricho JL, Cotrina J, 2009, 'Activity Recognition from Accelerometer Data on a Mobile Phone', *Lecture Notes in Computer Science*, 5518, pp796-799
- Kwapisz JR, Weiss GM, Moore SA, 2010, 'Activity Recognition using Cell Phone Accelerometers', *Knowledge Discovery from Sensor Data*, pp10-18
- Ravi N, Dandekar N, Mysore P, Littman ML, 2005, 'Activity Recognition from Accelerometer Data', *American Association for Artificial Intelligence*
- Casale P, Pujol O, Radeva P, 2011, 'Human Activity Recognition from Accelerometer Data using a Wearable Device', *Lecture Notes in Computer Science*, 6669, pp289-296
- Krishnan NC, Colbry D, Juillard C, Panchanathan S, 2008, 'Real Time Human Activity Recognition using Tri-axial Accelerometers', *Sensors, Signals and Information Processing*
- Rousseeuw P, 1984, 'Least Median of Squares Regression', *Journal of the American Statistical Association*, 79, pp871-880
- Rousseeuw P, and Leeroy A, 1987, 'Robust Regression and Outlier Detection', *John Wiley and Sons*, pp380

Appendix

ImportData.m

```
function [A] = ImportData(filename)

if (~exist(filename,'file'))
    error(['Unable to open file ' filename]);
end

fid = fopen(filename,'r');
lines = textscan(fid,'%s','delimiter','\n');
fclose(fid);

numlines = length(lines{1});

j = 0;

hour = [];
minute = [];
second = [];
tstamp = [];
acc = [];
mag = [];
bearing = [];
gyro = [];
grav = [];
linacc = [];
rotvec = [];

for i = 1:numlines

    line_i = lines{1}{i};
    textscan(line_i,'%d','delimiter',':');
    colonpos = strfind(line_i,':');

    if length(colonpos) == 3
        if strcmp(line_i(colonpos(2)+1:colonpos(3)-
1),'SensorsNO') == 1

            j = j + 1;

            timestr = line_i(1:colonpos(1)-1);
            hour(j) = str2num(timestr(9:10));
            minute(j) = str2num(timestr(11:12));
            second(j) = str2num(timestr(13:17))/1000;
            tstamp(j) = (hour(j) * 60^2) + (minute(j) * 60) +
second(j);

            data = line_i(colonpos(3)+2:end);

            spacepos = strfind(data,' ');
```

```
            accdata = data(5:spacepos(1)-2);
            magdata = data(spacepos(1)+5:spacepos(2)-2);
            gyrodata = data(spacepos(3)+6:spacepos(4)-2);
            gravdata = data(spacepos(7)+6:spacepos(8)-2);
            linaccdata = data(spacepos(8)+8:spacepos(9)-2);
            rotvecdata = data(spacepos(9)+8:spacepos(10)-2);

            commapos = strfind(accdata,',');

            acc(j,1) = str2num(accdata(1:commapos(1)-1));
            acc(j,2) =
str2num(accdata(commapos(1)+1:commapos(2)-1));
            acc(j,3) = str2num(accdata(commapos(2)+1:end));

            commapos = strfind(magdata,',');

            mag(j,1) = str2num(magdata(1:commapos(1)-1));
            mag(j,2) =
str2num(magdata(commapos(1)+1:commapos(2)-1));
            mag(j,3) = str2num(magdata(commapos(2)+1:end));

            bearing(j) = str2num(data(spacepos(2)+9:spacepos(3)-
2));

            commapos = strfind(gyrodata,',');

            gyro(j,1) = str2num(gyrodata(1:commapos(1)-1));
            gyro(j,2) =
str2num(gyrodata(commapos(1)+1:commapos(2)-1));
            gyro(j,3) = str2num(gyrodata(commapos(2)+1:end));

            commapos = strfind(gravdata,',');

            grav(j,1) = str2num(gravdata(1:commapos(1)-1));
            grav(j,2) =
str2num(gravdata(commapos(1)+1:commapos(2)-1));
            grav(j,3) = str2num(gravdata(commapos(2)+1:end));

            commapos = strfind(linaccdata,',');

            linacc(j,1) = str2num(linaccdata(1:commapos(1)-1));
            linacc(j,2) =
str2num(linaccdata(commapos(1)+1:commapos(2)-1));
            linacc(j,3) =
str2num(linaccdata(commapos(2)+1:end));

            commapos = strfind(rotvecdata,',');

            rotvec(j,1) = str2num(rotvecdata(1:commapos(1)-1));
            rotvec(j,2) =
str2num(rotvecdata(commapos(1)+1:commapos(2)-1));
```

```
            rotvec(j,3) =
str2num(rotvecdata(commapos(2)+1:commapos(3)-1));
            rotvec(j,4) =
str2num(rotvecdata(commapos(3)+1:commapos(4)-1));
            rotvec(j,5) =
str2num(rotvecdata(commapos(4)+1:end));


        end
    end
end

save([filename '.mat'],'tstamp', 'acc', 'mag', 'bearing',
'gyro', 'grav', 'linacc', 'rotvec');

A.time = tstamp';
A.accel = acc;
A.mag = mag;
A.bearing = bearing';
A.gyro = gyro;
A.grav = grav;
A.linaccel = linacc;
A.rotation = rotvec;

end
```

FeatureVector.m

```
function A = FeatureVector(x,t,w,o)

A = zeros(10,size(x,2));
i = 1;
ts = t(i);
k = 0;
while ((t(end)-t(i)) >=w )
    k = k+1;
    j = (i-1) + find(t(i:end) <= ts + w, 1, 'last');

    if ~isempty(j)
        I(k,:) = [i j];
        i = (i-1) + find(t(i:j) <= ts + o, 1, 'last');
        ts = t(i);
        if isempty(i)
            break;
        end
    end
end


for i = 1:length(I)

    k = I(i,1):I(i,2);

    A(1,:,i) = median(x(k,:));
    A(2,:,i) = mean(x(k,:));
    A(3,:,i) = geomean(abs(x(k,:)));
    A(4,:,i) = harmmean(x(k,:));
    A(5,:,i) = ltsq(x(k,:));
    A(6,:,i) = lmsq(x(k,:));
    A(7,:,i) = shorth(x(k,:));
    A(8,:,i) = rms(x(k,:));
    A(9,:,i) = std(x(k,:));
    A(10,1,i) = trapz(pwelch(fft(x(k,1))));
    A(10,2,i) = trapz(pwelch(fft(x(k,2))));
    A(10,3,i) = trapz(pwelch(fft(x(k,3))));
end

end
```