

Classification of Accelerometer Data as Input for User Interfaces

Ahmed Hassan
School of Computer Science
Carleton University
1125 Colonel By Drive
Ottawa, Ontario, Canada

December 20, 2007

Note

The work presented here is innovative and represents the summary for the honours project COMP 4900 during the summer term of 2007 at Carleton University.

Keywords

Accelerometer, Acceleration, Fuzzy logic, Real time, Training, Recognition,

1 Introduction

1.1 Context/Background

Gamers and developers are looking at new ways of interacting with games. The idea of this project is an extension of what Nintendo Wii has accomplished, which is recognizing hard coded motions. The outcome of this project aims to be used in gaming to give players the freedom to create their own motions.

1.2 Definition of the problem

The purpose of this project is to get more accurate movement recognition using a variable number of accelerometers for on the spot trained movements based on accelerations. There currently exists no published algorithm that solves this problem.

Other helpful work that has been completed regarding pattern matching and motion by Gabayan and Lansel [2], and also by Wu et al [3]. The relevance of their work is linked to the algorithms developed in this project.

1.3 Summary of the result

There are three algorithms that work together, DataStorage, Training and Recognition. DataStorage algorithm formats the data for Training algorithm. Training algorithm calculates representative data from recorded motions. Recognition algorithm compares the real time data against the representative data generated by the Training algorithm.

Test subjects had 97% successful recognition when performing motions that were different than other moves being tested. Accurate results are obtained when the time interval between readings is 25 ms. As the subjects attempt similar motions recognition between those movements produced false positives which reduced the accuracy.

1.4 Outline of the report

Background information about the accelerometers used in this project is provided in Section 2. The algorithms and the sensitivity variables are reviewed in Section 3. The outcome of this work is presented in Section 4. Future work is suggested in Section 5.

2 Detailed context/background information

We review in this section the tri-axis accelerometer circuit, which is shown in Figure 1.

The tri-axis accelerometers developed by Kionix can detect accelerations in three axis. Figure 2 below describes the tri-axis, pitch, roll and yaw. Pitch and roll are recognizable with one accelerometer, while yaw requires at least two accelerometers to be accurately measured. The area of the square accelerometers is 4.0 cm^2 .

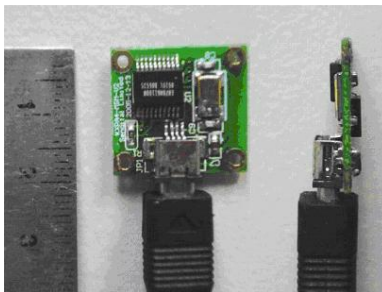


Figure 1: The accelerometers developed by Kionix.

The accelerometers connect to the computer through USB connections. Communication and powering of the device occur through this connection.

The accelerometer detects accelerations within the ranges of $+2.5g$. The accelerometers are capable of taking reading every one millisecond. The force of gravity causes constant one g acceleration towards the earth.

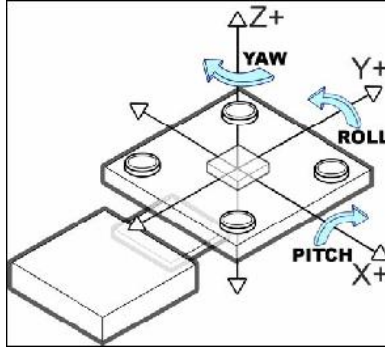


Figure 2: Tri-axis, Pitch, Roll and Yaw.

3 DataStorage, Training and Recognition Algorithms

There are three algorithms: DataStorage, Training and Recognition. DataStorage algorithm organizes the data for the Training algorithm. Training algorithm produces one set of representative data per move from the DataStorage algorithm. Recognition Algorithm checks conditions between the real time data and the representative data. To account for human variation in motions, sensitivity variables are used.

3.1 DataStorage Algorithm

This algorithm fills in the data in the format that the Training algorithm requires, referred to as Moves. Moves keep track of q moves, each move has s samples, each sample contains r accelerometers and each accelerometer stores t time. Time store the acceleration data for axis x , y and z .

The pseudocode for DataStorage is provided bellow.

```

1: for  $q = 0$  to  $i$  do
2:   for  $k = 0$  to  $s$  do
3:     for  $j = 0$  to  $r$  do
4:       for  $l = 0$  to  $t$  do
5:         store the X, Y and Z axis accelerations respectively into
            $Moves[q].Sample[k].Accelerometer[j].Time[l].X$ 
            $Moves[q].Sample[k].Accelerometer[j].Time[l].Y$ 
            $Moves[q].Sample[k].Accelerometer[j].Time[l].Z$ 
6:       end for
7:     end for
8:   end for
9: end for
  
```

DataStorage variables that yield highest accuracy are: s set to 5, $7 \leq t \leq 9$ and $a = 100$ ms. These variables are obtained through experimentation conducted during this study. See section 4 for more detailed justification.

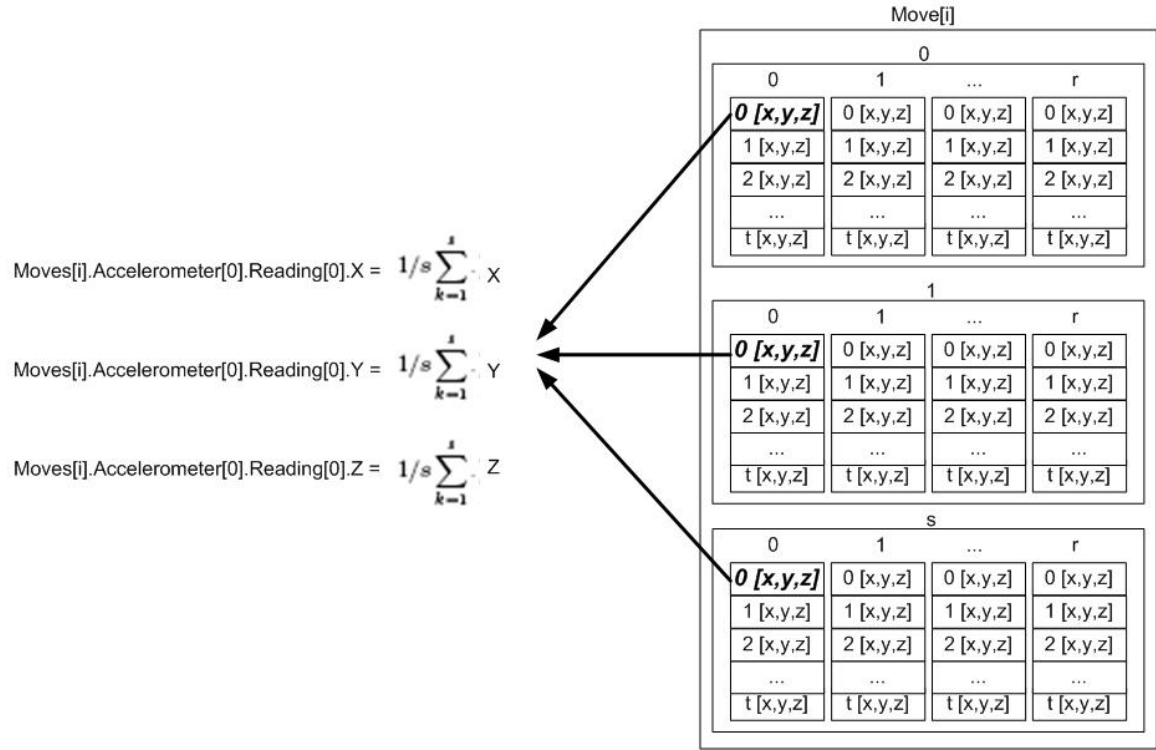


Figure 3: Moves, created by the DataStorage algorithm.

3.2 Training Algorithm

This algorithm averages recorded data and computes a representative set of data for each move, referred to as FinalReading. The diagram in Figure 3 shows a sample of traversing the data structure.

FinalReading contains the average from every time reading per sample for acceleration in the x , y and z . The following equation describes how FinalReading is calculated.

Let variable i denote the index of a move ($i = 1, \dots, q$), j denote the index of accelerometers ($j = 1, \dots, r$), k the index of samples ($k = 1, \dots, s$), and l denote the index of time readings ($l = 1, \dots, t$).

$$Moves[i].Accelerometer[j].Time[l].X = \frac{1}{s} \sum_{k=1}^s Moves[i].Sample[k].Accelerometer[j].Time[l].X$$

$$Moves[i].Accelerometer[j].Time[l].Y = \frac{1}{s} \sum_{k=1}^s Moves[i].Sample[k].Accelerometer[j].Time[l].Y$$

$$Moves[i].Accelerometer[j].Time[l].Z = \frac{1}{s} \sum_{k=1}^s Moves[i].Sample[k].Accelerometer[j].Time[l].Z$$

3.3 Recognition Algorithm

This algorithm obtains real time data and checks for a match against the FinalReading. This algorithm uses Fuzzy logic decision making. The rules used to verify a matched accommodate human variability in motion.

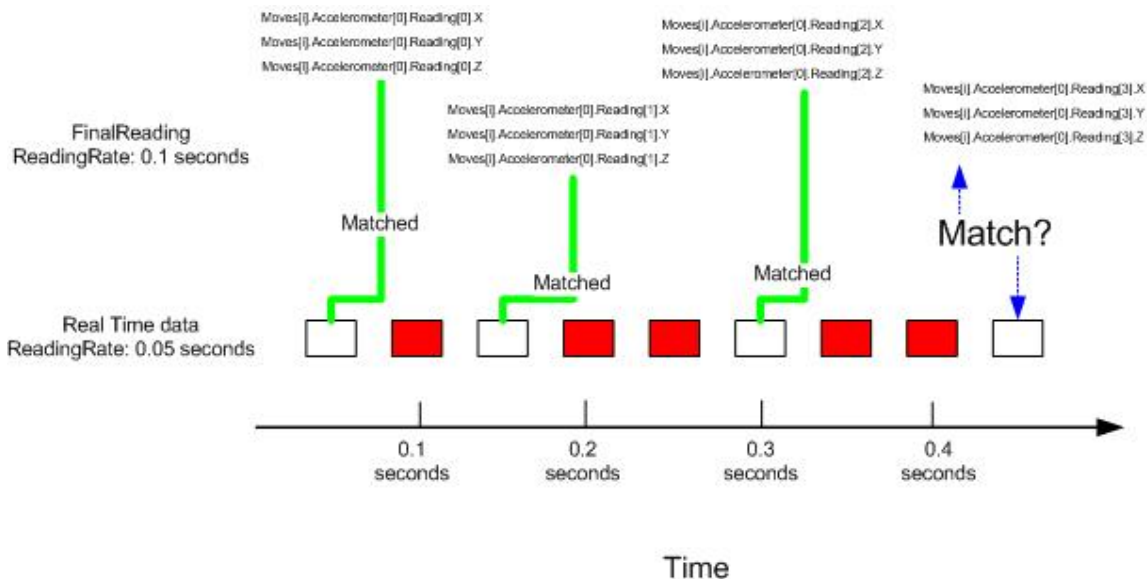


Figure 4: Graphical representation of realtime data storage being recognized against the trained data.

Figure 4 shows how the recognition process works. Real time data is sequentially compared against the data in FinalReadings. Variables used to determine successful matching are described in further detail in the following subsections.

3.3.1 Duration of a motion

The longer a motion takes to complete the less likely a human is to get a successful movement recognized.

The opposite is also true where the motion is short in duration then the Recognition algorithm has a higher probability of producing false positives. Short movement durations cause random motions to be recognized as a movement. This occurs because the acceptance range fall within the sensitivity variables. The best recognition is from motions that were roughly between 0.7 second and 0.8 second in duration with a time reading interval of 100 ms.

3.3.2 Acceleration of a motion

The accelerations have to be of the same order because the higher magnitude accelerations would contain the lower magnitude accelerations within their range of errors. This fact would lead to many false positives.

The following discussions cover the sensitivity variables used to filter out some of the problems.

3.3.3 Time-Reading-Interval

Time-reading-Interval represents the milliseconds between each reading that is required. The recognition algorithm does not produce correct results when the time-reading-interval is very small or very large. During the experimentation the time-reading-interval was set to 100 milliseconds.

If the intervals are too close in time then many false positives are produced. Long intervals in time make recognition very difficult.

Having a higher time reading rate for input data in relation to the recorded data is discussed by Wu et al. [3]. The result of implementing a similar process showed a great increase in recognition when more readings were taken for the recognition algorithm.

3.3.4 Movement, Time and Acceptable accuracy

There are variables used to modify the acceptance levels of Moves.

Having a faster time reading rate only accounted for faster movements but not for slower movements. This problem is approached by allowing the recognition to have a slightly longer time than the training data to reach a positive recognition.

The accuracy rates in each unit in time is referring to the number of correct comparison values obtained from all the axis from each accelerometer.

The sensitivity values used during the algorithm design require further research in order to get a clearer understanding of the effect, and in turn obtain better results.

4 Results

The accuracy of Training and Recognition algorithms and the sensitivity variables are tested. Some movements performed in the experimentation are similar to others, while the remaining movements are different. This is to test the accuracy of the recognition algorithm and to test the adjustment variables. Movements two and three are similar and movements five and six are similar. The similarity between movements is a qualitative attribute defined by the initial accelerations in the three axis for each accelerometer.

The Training and Recognition algorithms are tested by comparing at the ratio between movements attempted and recognized movements. Figure 5 shows six movements that subjects performed. The acceleration data for each movement is stored with variable lengths and variable time reading intervals. The movements are recorded with seven intervals and with ten readings ($t = 7$ readings and $t = 10$ readings respectively) ; each of those groups of data are also tested with different time reading rates ($a = 10ms$ and $a = 25ms$).

The results of this experiment are summarized in Figure 6. The results clarify the ranges for the sensitivity variables, $t = 7$ readings and $a = 25ms$. When a move is approximately 0.7 second long then the success rate is highest. Movements one and five have 97% and 93 % accuracy, respectively. However the same movements have 68% and 71% accuracy when the Recognize algorithm has a longer set of data from FinalReading to compare with (movement duration of 1.0 second). Most subjects were reaching six and seven correct readings out of the ten readings taken to describe each move.

The effect of the sensitivity variables for Recognition is clearer when comparing the results for each move. This comparison shows that having readings separated by 25 ms produces a higher success rate than time readings separated by 10 ms, this is likely because there are many incorrect positive recognitions. This happens because the algorithm receives incoming readings at a higher rate and their proximity in time

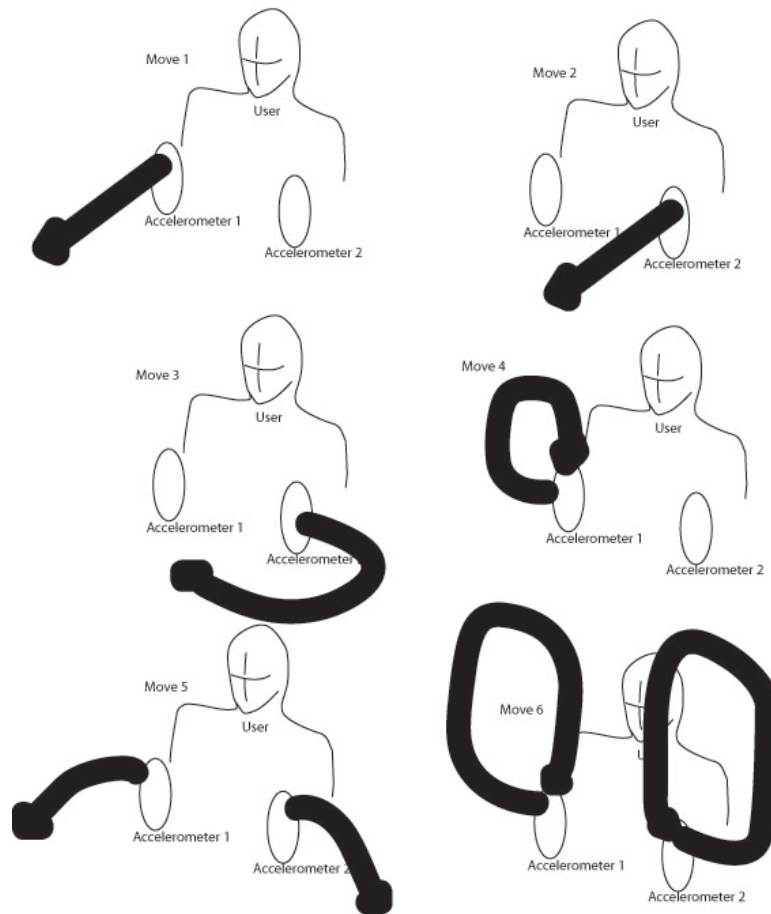


Figure 5: Movements designed to test Training and Recognition algorithms.

means their accelerations will also be close; this proximity causes false positives because the magnitude of the majority of the accelerations are very similar and thus many movements are incorrectly matched.

Recognition accuracy produced false positives when movements being tested are similar to each other. Looking at movements two and three the lowered accuracy was a result of false positives (movement two is often recognized as movement three). This was also true for movements five and six respectively.

5 Conclusions and future work

DataStorage, Training and Recognition algorithms produce the most accurate results when Accelerationthreshold is set to 1.1 g, readinginterval for training set to 100 ms, readinginterval for recognition set to 25 ms and when the motion lengths are approximately 0.7 seconds long. This information can be reliably used to duplicate this system and can therefore be tested for game play and perhaps for athletic training.

Movement	7 Training readings		10 Training readings	
	Recognition reading interval = 10 ms	Recognition reading interval = 25 ms	Recognition reading interval = 10 ms	Recognition reading interval = 25 ms
1	59	97	15	68
2	5	69	8	50
3	30	94	18	68
4	46	75	15	48
5	54	93	20	71
6	22	60	6	32

Figure 6: All Training algorithm data recorded at 100 ms time reading interval. All results are percentages of accurately recognized movements.

Further testing and development is required to modify the algorithms in order to increase the variability, tolerance and consistency. A self-modifying algorithm that bases its next sample reading on the first reading would be ideal. If the algorithm is capable of alter its own variables for the Training and Recognition algorithms then more reliable and accurate results should be obtainable..

6 References

- [1]Kionix,Inc. USB Demo Board Kit User’s Manual. June 7, 2006.
- [2]K.Gabayan, S. Lansel. Programming-By-Example Gesture Recognition. Stanford HCI, 2006.
- [3]S. Wu, U. Manber, G. Myers. An $O(NP)$ Sequence Comparison Algorithm. University of Arizona. W. Miller, The Pennsylvania State University, 1989.