gRmobile: A Framework for Touch and Accelerometer Gesture Recognition for Mobile Games

Mark Joselli, Esteban Clua
MediaLab, IC-UFF
{mjoselli — esteban}@ic.uff.br

***Abstract*—Mobile phone games are usually design to be able to play using the traditional number pads of the handsets. This is stressfully difficult for the user interaction and consequently for the game design. Because of that, one of the most desired features of a mobile games is the usage of few buttons as possible. Nowadays, with the evolution of the mobile phones, more types of user interaction are appearing, like touch and accelerometer input. With these features, game developers have new forms of exploring the user input, being necessary to adapt or create new kinds of game play. With mobile phones equipped with 3D accelerometers, developers can use the simple motion of the device to control the game or use complex accelerated gestures. And with mobile phones equipped with the touch feature, they can use a simple touch or a complex touch gesture recognitions. For the gesture to be recognized one can use different methods like simple brute force gestures, that only works well on simple gestures, or more complex pattern recognition techniques like hidden Markov fields, fuzzy logic and neural networks. This work presents a novel framework for touch/accelerometer gesture recognition that uses hidden Markov model for recognition of the gestures. This framework can also be used for the development of mobile application with the use of gestures.**

***Index Terms*—Mobile Games, Gesture Recognition, Motion Sensors, Touch Phones,Tangible User Interfaces.**

I. Introduction

Digital games are defined as real-time multimedia applications that have time constraints to run their tasks. If the game is not able to execute its processing under some time threshold, it will fail [1]. Mobile games are also real-time multimedia application that runs on mobile phones that have time constraints and many others constraints [2], when compared to PC or console games, like: hardware constraints (processing power and screen size); user input, (buttons, voice, touch screen and accelerometers); and different operating systems, like Android, iPhone OS, Symbian and Windows Mobile. This makes streamilly difficult for the design and development of mobile games.

On the other hand, mobile games can have unique characteristics, making unique type of games: location based games [3], [4], voice based games [5], accelerometer based games [6], camera based games [7] and touch based games [8]. In order to develop good mobile games, they must be design to take advantages of such unique characteristics into gameplay [9].

Mobile game phones are a growing market [10] and in 2010 the sales of smartphones is expected to suppress the laptops sales [11]. More than 10 million of people worldwide play games on mobile phones and handheld devices [12] and the worldwide mobile gaming revenue is expected to reach $9.6 billion by 2011 [13]. These are important motivations for game developers and designer to create blockbusters games.

One special characteristic of most mobile phones is that the user interaction is made mostly through number input [6], [14]. Because of that, the design of games must deal with this fact and design the game to use as few buttons as possible, like just one button games [15] and no-buttons at all games [14].

The evolution of mobile phones increase the processing power of such devices and also new forms of input, like touch screen devices and devices equipped with accelerometers. With the development of touch phones, like Motorola a1200, Htc Diamond, Sony Ericson w960i, Samsung Ultra

978-0-7695-3963-8/09 $26.00 © 2009 IEEE
DOI 10.1109/SBGAMES.2009.24

Smart F520 and Nokia N810, new forms of user interaction has appeared through the use of the finger or pen. This innovation has led to change the way users interact with the operation system and with games.

With the popularization of the use of accelerometer by the Nintendo Wiimote [16], the major mobile phone manufactures had also equipped their hardware with accelerometer, like Nokia N95, Sony ericson F305, Samsung Omnia and Motorola W7, among others. But this new form of user interaction has not led to major change on the interaction. This is mostly because programs/games only uses the accelerometer data as orientation.

The iPhone was one of the first devices that is equipped with touch screen and accelerometer that has mostly of the user input made though touch or motion, soon others companies followed this tendency, like: RIM Blackberry Storm, Nokia 5800, LG Arena, and many others. They basically use touch for user interaction, and the use of the accelerometer data is restricted for orientation, just like the others phones equipped with accelerometer. This paper tries to fulfill a gap on user interaction by providing a framework for gesture recognition though touch input or motion input, that can be used for games or programs.

The gesture recognition is a type of pattern recognition and can be made by different ways like: brute force [17], fuzzy logic [18], Gabor wavelet transform [19], hidden Markov model [20], Support Vector Machine [21] and neural networks [22], [23]. This work has developed a framework that can be used for gesture recognition using hidden Makov model. In order to generate and recognize the gestures database the proposed framework is divided in two parts: one for database constriction and another for the gesture recognition.

Summarizing, this work provides the following contributions:

- A novel architecture for touch/motion gesture recognition on mobile phones;
- Presentation of performance and tests of the framework showing that it can be used in real-time;
- Recognition test showing a high accuracy rate;

The paper is organized as follows: Section 2 presents some related works on the mobile development and gesture recognition on devices equipped with touch screen and devices equipped with accelerometers. Section 3 presents and explain the gesture recognition framework. Section 4 present and discuss some results of the use of the framework. Section 5 presents the conclusions and future works.

II. Related Work

Since the gRmobile has two kinds of gestures recognition (thought touch input or accelerometer motion input), this section is divided in two subsections: one for touch screen devices were user interaction and gesture recognition for this kind of device works; and another for accelerometer devices presenting user interaction works and gesture recognition approaches.

This work does not cover the related work on mobile game development. For this purpose the authors suggest the works [24], [2] which covers state of the art for this topic.

A. Touch Devices

Nowadays, more and more devices are coming with touch screen, and most of this is because of the decreased in the respective price [25]. Touch screen phone devices has the characteristics of having very few buttons and most of its users input interfaces are made through touch by finger or pen. For example the Blackberry Storm has about only 8 buttons and almost all of its user interaction is made by touch.

Most touch screen devices can have two kinds of input: dragged and pressed. The first is used when the user touches softly and can be used as a mouse being dragged. The second is when the user press hard on the device, and can be used as a mouse buttons pressed. Also modern devices has multi-touch screen devices like iPhone, Android T-G1 and Blackberry Storm, among others.

Some of these devices have used some of this types of input as gestures to enable friendly user interaction: like dragging for changing the web page, zooming options on photo view and many others features. But in third-party mobile software and games this use is very restrict.

Narayanaswamy et al [26] shows an implementation of handwrite recognition on PDAs using Hidden Markov Model. Wei et al. [27] presents a

study about using pen gestures instead of buttons in a mobile FPS game. They showed that users have little preference in using buttons over gestures and sometimes prefer the use of gestures. The gRmobile also could be used for handwrite recognition and gesture for games but a gesture database must be constructed in order to have the this functionality in the framework (in the case of handwrite recognition all the alphabet must be in the database).

Since the touch screens have similar behavior as the mouse, mouse gesture recognizer [28], [29], [30] could be adapted in order to be used by touch screen devices. But this could be very hard since they are not adapted for the low processing power and memory constraints of such devices. The gRmobile is very adapted to such devices having a very good performance as will be showed in the performance evaluation section.

Even tough touch devices enables much more freedom when compared with buttons based phones, the input error by those devices are higher, as shows by Hoggan et al. [31] using keyboard input tests.

B. Accelerometer Devices

Accelerometer devices are getting more and more popular. This allows the interaction though the form of gestures recognition, being the Nintendo Wii game console the most prominent example of this new form of interaction. This approach allows users to become more engaged to video games [32], whose experience is not only affected by button pressing and timing but also by movement. Also Sony's Playstation 3 has a controller that is equipped with accelerometers.

Nowadays, most smartphones, and also some mobile phones, come equipped with accelerometer. This allows the use of motion and gestures as user input, but very little has been done in the field. Most mobile operating systems [11] only uses the motion to choose the screen orientation. And also most games only uses the "tilt" of the screen and not gestures as input like the works [14], [6], [33].

There are many relevant work related to gestures recognition for accelerometer devices. With the usage of the Wiimote can be highlighted the works [34], [35]. These works use Hidden Markov Models (HMM) as the recognition algorithm and the [34] presents a lower recognition of 66 % and [35] shows a lower recognition rate of 84 %. This work also uses HMM as the recognition algorithm, but it is adapted for the lower processing power of mobile devices.

Accelerometer gesture recognition requires an intensive task to be achieved on a mobile phone. The work by Choi et al. [36] presents a Bayesian network algorithm with its computation performed in the PC to recognize numbers written on the air by accelerometer mobile phones with an average recognition rate of 97 %. Also [37] that shows a HMM recognition algorithm also implemented on the PC with the gestures done by mobile phones with a recognition rate of more that 99 %. There are also some work in development like [21], [38] shows two recognition algorithms, a HMM and a Support Vector Machine, which can have an average recognition rate of 96 %.

Without the use of the PC, MobiToss [39] presents the use of mobile phone to use simple gestures to interact with large public displays with all the processing made on the phone.

III. Framework Overview

The framework is build using Java language. The choice to use Java was to achieve a higher number of devices with the same framework, allowing its usage by any accelerated/touch device that can handle the acceleration data and/or touch data, and that has a Java virtual machine, like many devices from different manufacturers: Blackberries, Nokias, Sony Ericson, Motorola, Android, LG and Samsung. Also this framework can be used in a PC with a touch device like Microsoft surface or an accelerometer device like the Wiimote.

Gesture recognition with touch/accelerated devices are represented by their patterns of the input data. The recognition is made by comparing the input pattern with the database pattern, checking if they match. In order to extract the pattern from the input data stream and comparing with the database pattern, this data must be prepared and analyzed. This work uses Hidden Markov Models in order to fulfill that need. In order to build a database, the framework needs to train and saves a set of gestures, which also used the mobile phone for this need.

The proposed framework has the following steps:

- Segmentation: is used to determine when the gesture begins and when it ends;
- Filtering: is used in order to eliminate some parts of the data stream that do not contribute to the gesture;
- Quantitizer: is used to approximate the stream of input data into a smaller set of values;
- Model: is used to compute likelihood of analyzed gestures.
- Classifier: is used in order to identify the input gesture accordingly to the database.

All the steps from the mobile phone to the user feedback are illustrated in figure 1. It is possible to notice that the framework has two distinct modes: one for gesture training, i.e, build the database, and one for gesture recognition, i.e, comparing the input gesture with the database.

A. Segmentation

Segmentation is used mainly to automatic determinate the begin and end of the gesture. For identifying the begin and end of touch gestures is very easy since the begin of the gesture is when the user first touch the screen and it ends when the user release the screen. For the accelerometer gestures the segmentation is more difficult since the data is continues in a frequency normally that varies from 20 Hz to 80 Hz. The data that comes from the accelerometer is 3 floats representing the acceleration in three axis, as figure 2 illustrate, that ranges from -3G to 3G (G meaning the gravity).

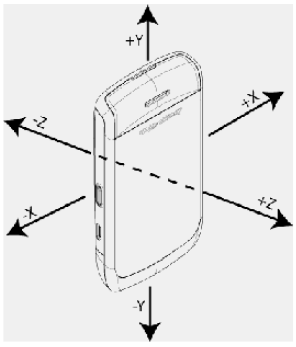

Fig. 2. The axis accelerometer

There are some accelerometer gesture recognition systems, like [35], that does the segmentation by using a button based segmentation, i.e, the user needs to press a button in order to sign the beginning and the end of the gesture. This seems like an easy approach, since it avoids computations to determine the begin and the end of a gesture like automatic segmentation must have. But on the other hand, the interaction is worst than no-button segmentation. In a mobile phones sometimes the pressing of buttons in a game is normally hard since it was designed for number dialing. This comes as another reason why this work has decided to do a no-button segmentation. This work does a similar approach as the works [40], [21].

In order to correct segmentate an accelerometer gesture, a definition of this kind of gesture is needed. This definition is made during the observation of the accelerated data and the movement of user during the recognition of different gestures. Normally gestures begins with a fast acceleration, a continuous direction change during the gesture, and it ends with a stop of the movement. In this work, the authors have observed that normally, a good gesture needs a duration of more than 0.6 seconds and less than 2 seconds.

To correct segmentate the accelerometer gesture based on the definitions some preprocessing of the accelerated data is needed. This work uses a simple mechanism. It checks the size of a vector made by the sums of the derivative, which is the difference between the axis float and the last axis float.

$$D = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2 + (z_k - z_{k-1})^2} \quad (1)$$

If the D value is bigger than 0.3, which was chosen during an extensively empirical study of gestures, the segmentation begins. And if the gesture is happening and this values drops to bellow 0.1 the gesture ends, i.e, it is assumed that the accelerometer device is in an idle position.

B. Filtering

This pass is used in order to eliminate some parts of the data stream that do not contribute to the gesture. This work uses two kinds of filters in order to eliminate noises and data that are very similar.

When a gesture is made, the data stream of the gesture may contain errors that if are sent to the HMM, some errors on the recognition may occur. In order to avoid such errors, a low pass filter is

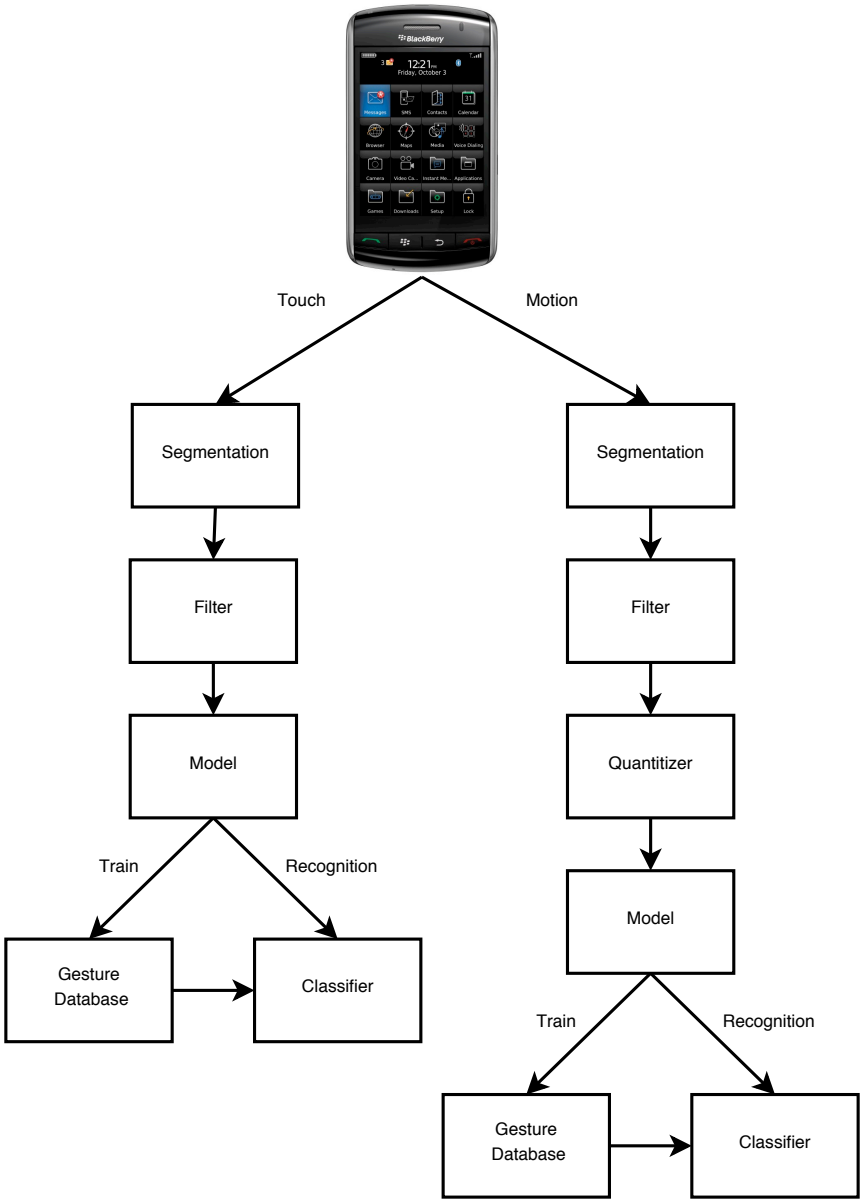

Fig. 1. System block diagram with major components of the framework.

applied which is a very common filter used for noise remove.

When the gesture is made, there are a lot of data on the data stream that does not contribute to the overall characteristic of the gesture. In order to diminished the data passed to the HMM, this work uses an idle threshold filter. This uses the same equation 1, and if D value is less than 0.2, it is not included in the gesture data stream.

C. *Quantitizer*

This step is only used for accelerated gestures. Because the accelerometer continues sends its data to the processor, the amount of data may be to big to for handling into a single HMM. Also, since the amount of RAM memory of mobile phones are not so big, the use of a quantitizer keeps less information in the gesture database. This work uses a k-mean algorithm which is a method of cluster analyzer. The algorithm aims to partition n observation

into k clusters in which each observation belongs to the cluster with the nearest mean. This is a similar approach as the work [35].

For this approach, a k value must be chosen. For this work we selected $k = 14$ since experiments from [35] shows that this values is optimal for the Wiimote, which can be extended to the phone accelerometers, since they have a similar architecture.

D. Hidden Markov Model

A hidden Markov model (HMM) is a popular statistical tool for gesture/pattern recognition. This work has based its implementation in an open source HMM implementation [41] and in an open source HMM Wii gesture recognition [42].

This work uses left-to-right HMM with 8 states for each gesture. The reason why this work choose this configuration is because it is a very efficient for accelerometer recognition, following the tests made in [35]. For the training process, the HMMs with the iterative Baum-Welch algorithm was used. And for recognition, the forward-backward algorithm was used. More information on this algorithms can be obtained in [43], [44], [45].

E. Classifier

The classifier is used in order to identify gesture selecting the gesture with more likehood between the input gesture and the database gesture. This work uses a naive Bayes classifier also called simple Bayesian classifier [46]. Naive Bayes is simple probabilistic classifier based on the so-called Bayesian theorem and it is a well known algorithm both in statistics and machine learning.

Because the accelerometer data input has some noise movements, i.e., movements that are not gestures. Because of that, a probability threshold is included, so that any gesture probability below that threshold is considered as a noise. The value of this threshold was determined according to empirical values.

IV. Results Evaluation

All tests of this work were made in a BlackBerry Storm 9530 [47] which has a 528 MHz Qualcomm processor with 128 MB of RAM, touch screen and accelerometer. In order to evaluate properly the gRmobile framework an application to train, recognize and save gestures were developed for the BlackBerry Storm, a screenshot of the application can be seen on figure 3.

Fig. 3. An Screenshot of the application.

Two types of tests were made in order to validate the architecture performance, one for evaluating the impact that the architecture can have on the mobile phone, and another for recognition, in order to evaluate the accuracy of the gRmobile. These tests are presented in the next two subsections.

A. Performance Evaluation

There are some available frameworks for gesture recognition but most of them cannot be used in mobile phones since the processing power is very low

when compared to a PC. gRmobile was designed to be used with constraint hardware of mobile phones.

The authors of this work have observed that the size of gesture database influenciates on the total time of the recognition. Another observation is that the time for touch gesture is lesser than motion gestures, since it has much less data. Table I shows the numerical results in average time with gesture different database sizes.

The results shows that with a database with ten or less gestures, the gRmobile can be used in real-time applications, such as games, without major impact on the overall performance of the application. Since more than ten gestures becomes inpratical due to the fact that user must learn all different gestures, like [37] argue and the authors of this work agreed. These results shows that gRmobile provides a good rate for real-time gesture recognition.

Also this architecture were tested in a PC with an 3500+ Athlon64 processor with 2GB RAM memory with Wiimote as the accelerometer device and the mouse simulating the touch device. This tests shows that resource consumption is insignificant in a PC even with a gesture database of twenty gestures.

B. Recognition Evaluation

In order to test the accuracy of the recognition, a dataset of gestures have been created. These work have defined ten different gestures for motion gestures, and ten gestures for touch gestures similar to the motion gestures. These gestures can be seen on figure 4.

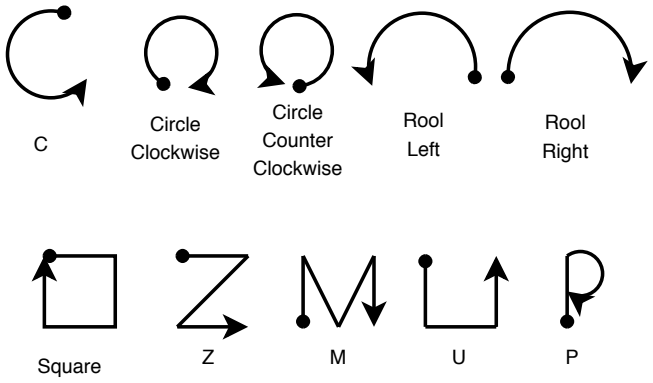

Fig. 4. The gesture database used in tests.

All gestures were repeated ten times by a group of four different users, which provided a four hundred examples overall. The group consists of three men users (participants A, B and D) and one women user (participant C) with age ranging between 21 and 42. None of the participants was physically disabled. One participant have major experience with touch/accelerometer device (participant A), two have minor experience with such devices (participant B and C) and one have none experience with the touch and accelerometer mobile phone devices (participant D). The results can be seen on table II.

These results show a high fidelity recognition rate of the gRmobile framework with an average recognition rate of 89% in motion gestures and 98 % for touch gestures.

Also the results shows that the specialist user obtained 99.9 % of recognition in motion gestures and 100 % in touch gestures. The low experience users have an 88.5 % in motion gestures and 98 % in touch gestures. The no experience user have 79 % in motion gestures and 96 % in touch gestures. This results show that touch gestures are easily to be performed and recognized. The tests also shows that gestures can be used with both user with big expertise and no expertise on the subject.

Figure 5 shows the average recognition in % rate of motion gestures. This results shows that the P gesture has the lowest recognition rate, an average rate of 75%, and $RollLeft$ gesture has the highest recognition rate, an average rate of 97.5%.

Fig. 5. Average Recognition Rate of the Motion Gestures.

Figure 6 shows the average recognition in % rate of touch gestures. This results shows that the

TABLE I
NUMERICAL RESULTS FROM PERFORMANCE EVALUATION WITH DIFFERENT DATABASE SIZES IN AVERAGE TIME IN MILLISECONDS.

# of gesture in the database	time accelerometer (ms)	time touch (ms)
5	52	40
10	87	71
15	125	107
20	186	154

TABLE II
RECOGNITION TESTS RESULTS IN % OF ACCURACY BETWEEN ALL PARTICIPANTS.

Gesture	User A		User B		User C		User D	
	Motion	Touch	Motion	Touch	Motion	Touch	Motion	Touch
C	100	100	90	100	100	90	90	100
Circle Clockwise	100	100	100	100	80	100	80	90
Circle counter Clockwise	100	100	90	90	100	100	80	100
Roll Left	100	100	100	100	100	100	90	100
Roll Right	100	100	100	100	90	100	100	100
Square	100	100	80	100	90	100	70	80
Z	100	100	80	100	90	100	70	90
M	100	100	90	100	80	90	70	100
U	100	100	80	100	80	100	80	100
P	90	100	70	100	80	90	60	100

$Square$ gesture has the lowest recognition rate, with an average rate of 95% and $RollLeft$, $RollRight$ and U gestures have the highest recognition rate, with an average rate of 100%.

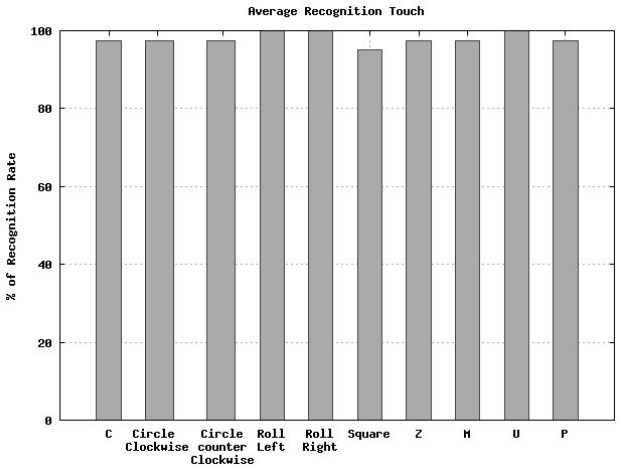

Fig. 6. Average Recognition Rate of the Touch Gestures.

V. CONCLUSION

The mobile gaming market is a growing market, motivating game developers to have more focus on mobile development. Also mobile phones now have much more processing power allowing mobile games to have more complexity.

Even touch screens and accelerometers devices becoming available in mostly devices, current mobile games almost do not explore these features. This work has presented a novel framework, the gRmobile, that can recognize touch and motion gestures in real-time using the mobile phone limited hardware.

The gRmobile was designed to help developer to better explore this new kind of input through gestures. This framework was built in Java in order to address most mobile phones (Symbian, Android, BlackBerry and others phones that have JVM).

There are others frameworks for gesture recognition on accelerometer and touch devices, but none of them provide the unique characteristics of gRmobile like: providing both touch and accelerometer recognition on the same platform; run in real time on the restricted mobile phone hardware; developed in a way that can run in any system with JVM.

This work also presented many performance tests with the framework, showing that the solution can run in real-time on mobile phone devices. Also recognition test were made, showing that the gRmobile has a high recognition rate of 89% in motion gestures and 98 % for touch gestures.

In the future work topics, it is included to porting the framework to iPhone and Windows Mobile

platforms. The authors also point as future work, the study of how to substitute the key press process for gestures in traditional mobile games and how this affects the gameplay.

REFERENCES

[1] M. Joselli, M. Zamith, L. Valente, E. W. G. Clua, A. Montenegro, A. Conci, and P. Feij, Pagliosa, "An adaptative game loop architecture with automatic distribution of tasks between cpu and gpu," *Proceedings of the VII Brazilian Symposium on Computer Games and Digital Entertainment*, pp. 115–120, 2008.

[2] F. Chehimi, P. Coulton, and R. Edwards, "Evolution of 3d mobile games development," *Personal Ubiquitous Comput.*, vol. 12, no. 1, pp. 19–25, 2008.

[3] M1ndLab, "Alien revolt: Location-based massive-multiplayer online rpg." Avalible at: `http://www.alienrevolt.com`, 2007.

[4] A. De Souza E Silva, "Hybrid reality and location-based gaming: Redefining mobility and game spaces in urban environments," *Simul. Gaming*, vol. 40, no. 3, pp. 404–424, 2009.

[5] M. J. Zyda, D. Thukral, J. C. Ferrans, J. Engelsma, and M. Hans, "Enabling a voice modality in mobile games through voicexml," in *Sandbox '08: Proceedings of the 2008 ACM SIGGRAPH symposium on Video games*, Sandbox. New York, NY, USA: ACM, 2008, pp. 143–147.

[6] F. Chehimi and P. Coulton, "Motion controlled mobile 3d multiplayer gaming," in *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, ACE. New York, NY, USA: ACM, 2008, pp. 267–270.

[7] A. Park and K. Jung, "Flying cake: Augmented game on mobile devices," *Comput. Entertain.*, vol. 7, no. 1, pp. 1–19, 2009.

[8] M. Rohs, "Marker-Based Embodied Interaction for Handheld Augmented Reality Games," *Journal of Virtual Reality and Broadcasting*, vol. 4, no. 5, Mar. 2007, `urn:nbn:de:0009-6-7939`, ISSN 1860-2037.

[9] M. Zyda, D. Thukral, S. Jakatdar, J. Engelsma, J. Ferrans, M. Hans, L. Shi, F. Kitson, and V. Vasudevan, "Educating the next generation of mobile game developers," *IEEE Computer Graphics and Applications*, vol. 27, no. 2, pp. 96, 92–95, 2007.

[10] E. M. I. Koivisto, "Mobile games 2010," in *CyberGames '06: Proceedings of the 2006 international conference on Game research and development*, CyberGames. Murdoch University, Australia, Australia: Murdoch University, 2006, pp. 1–2.

[11] E. Oliver, "A survey of platforms for mobile networks research," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 12, no. 4, pp. 56–63, 2008.

[12] J. O. B. Soh and B. C. Y. Tan, "Mobile gaming," *Commun. ACM*, vol. 51, no. 3, pp. 35–39, 2008.

[13] Gartner, "Gartner says worldwide mobile gaming revenue to grow 50 percent in 2007," Avalible at: `http://www.gartner.com/it/page.jsp?id=507467`, 2007.

[14] P. Gilbertson, P. Coulton, F. Chehimi, and T. Vajk, "Using "tilt" as an interface to control "no-button" 3-d mobile games," *Comput. Entertain.*, vol. 6, no. 3, pp. 1–13, 2008.

[15] F. Nokia, "Turn limitation into strength: Design one-button games," Nokia, Tech. Rep., 2006. [Online]. Available: http://sw.nokia.com/id/8dff4326-3979-4149-96c0-5fa95a14a3cb/Turn_Limitation_into_Strength_Design_One-Button_Games_v1_0_en.pdf

[16] J. Delgado, M. Joselli, S. Stanzani, S. M. Sadjadi, E. Clua, and H. Alvarez, "A learning and collaboration platform based on sage," in *WCCCE '09: Proceedings of the 14th Western Canadian Conference on Computing Education*. New York, NY, USA: ACM, 2009, pp. 70–76.

[17] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes," in *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, UIST. New York, NY, USA: ACM, 2007, pp. 159–168.

[18] L. Anderson, D. J. Purdy, and W. Viant, "Variations on a fuzzy logic gesture recognition algorithm," in *ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, ACE. New York, NY, USA: ACM, 2004, pp. 280–283.

[19] J. Mena-Chalco, H. Carrer, Y. Zana, and R. M. Cesar Jr., "Identification of protein coding regions using the modified gabor-wavelet transform," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 5, no. 2, pp. 198–207, 2008.

[20] T. Westeyn, H. Brashear, A. Atrash, and T. Starner, "Georgia tech gesture toolkit: supporting experiments in gesture recognition," in *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, ICMI. New York, NY, USA: ACM, 2003, pp. 85–92.

[21] Z. Prekopcsák, P. Halácsy, and C. Gáspár-Papanek, "Accelerometer based real-time gesture recognition," in *Proceedings of the 12th International Student Conference on Electrical Engineering*. International Student Conference on Electrical Engineering, 2008.

[22] D. King, W. B. Lyons, C. Flanagan, and E. Lewis, "Signal processing technique utilising fourier transform methods and artificial neural network pattern recognition for interpreting complex data from a multipoint optical fibre sensor system," in *WISICT '04: Proceedings of the winter international synposium on Information and communication technologies*, WISICT. Trinity College Dublin, 2004, pp. 1–6.

[23] G. Bailador, D. Roggen, G. Tröster, and G. Trivi no, "Real time gesture recognition using continuous time recurrent neural networks," in *BodyNets '07: Proceedings of the ICST 2nd international conference on Body area networks*, BodyNets. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–8.

[24] T. Capin, K. Pulli, and T. Akenine-Möller, "The state of the art in mobile graphics research," *IEEE Comput. Graph. Appl.*, vol. 28, no. 4, pp. 74–84, 2008.

[25] S. J. Vaughan Nichols, "New interfaces at the touch of a fingertip," *Computer*, vol. 40, no. 8, pp. 12–15, 2007.

[26] S. Narayanaswamy, J. Hu, and R. Kashi, "User interface for a pcs smart phone," in *ICMCS '99: Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, ICMCS. Washington, DC, USA: IEEE Computer Society, 1999, p. 9777.

[27] C. Wei, G. Marsden, and J. Gain, "Novel interface for first person shooting games on pdas," in *OZCHI '08: Proceedings of the 20th Australasian Conference on Computer-Human Interaction*, OZCHI. New York, NY, USA: ACM, 2008, pp. 113–121.

[28] M. Moyle and A. Cockburn, "Analysing mouse and pen flick gestures," in *In Proc. of the SIGCHI-NZ Symposium On Computer-Human Interaction*. SIGCHI, 2002, pp. 266–267.

[29] L. Lombardi and M. Porta, "Adding gestures to ordinary mouse

use: a new input modality for improved human-computer interaction," in *ICIAP '07: Proceedings of the 14th International Conference on Image Analysis and Processing*, ICIAP. Washington, DC, USA: IEEE Computer Society, 2007, pp. 461–466.

[30] M. Buckland, *AI Techniques for Game Programming (The Premier Press Game Development Series)*. Course Technology PTR, October 2002. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/193184108X

[31] E. Hoggan, S. A. Brewster, and J. Johnston, "Investigating the effectiveness of tactile feedback for mobile touchscreens," in *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI. New York, NY, USA: ACM, 2008, pp. 1573–1582.

[32] N. Crampton, K. Fox, H. Johnston, and A. Whitehead, "Dance dance evolution: Accelerometer sensor networks as input to video games," in *IEEE HAVE 2007*. IEEE HAVE, 2007, pp. 74–84.

[33] L. Valente, C. S. de Souza, and B. Feijó, "An exploratory study on non-visual mobile phone interfaces for games," in *IHC '08: Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems*, SBC. Porto Alegre, Brazil, Brazil: Sociedade Brasileira de Computaç ao, 2008, pp. 31–39.

[34] J. Mlch, "Wiimote gesture recognition," in *Proceedings of the 15th Conference and Competition STUDENT EEICT 2009 Volume 4*, Faculty of Electrical Engineering and Communication BUT. Faculty of Electrical Engineering and Communication BUT, 2009, pp. 344–349. [Online]. Available: http://www.fit.vutbr.cz/research/view_pub.php?id=8933

[35] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a wii controller," in *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, TEI. New York, NY, USA: ACM, 2008, pp. 11–14.

[36] E. S. Choi, W. C. Bang, S. J. Cho, J. Yang, D. Y. Kim, and S. R. Kim, "Beatbox music phone: gesture-based interactive mobile phone using a tri-axis accelerometer," in *Industrial Technology, 2005. ICIT 2005. IEEE International Conference on*. ICIT, 2005, pp. 97–102.

[37] T. Pylvänäinen, "Accelerometer based gesture recognition using continuous hmms," *Pattern Recognition and Image Analysis*, pp. 639–646, 2005. [Online]. Available: http://dx.doi.org/10.1007/11492429_77

[38] Z. Prekopcsák, P. Halácsy, and C. Gáspár-Papanek, "Design and development of an everyday hand gesture interface," in *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, MobileHCI. New York, NY, USA: ACM, 2008, pp. 479–480.

[39] J. Scheible, T. Ojala, and P. Coulton, "Mobitoss: a novel gesture based interface for creating and sharing mobile multimedia art on large public displays," in *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, MM. New York, NY, USA: ACM, 2008, pp. 957–960.

[40] F. G. Hofmann, P. Heyer, and G. Hommel, "Velocity profile based recognition of dynamic gestures with discrete hidden markov models," in *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, International Gesture Workshop. London, UK: Springer-Verlag, 1998, pp. 81–95.

[41] J.-M. Franois, "Jahmm: Java implementation of hidden markov model (hmm) related algorithms," Avalible at: `http://code.google.com/p/jahmm/`, 2009.

[42] B. Poppinga, "wiigee:a java-based gesture recognition library for the wii remote," Avalible at: `http://www.wiigee.org`, 2009.

[43] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ASSP Magazine, IEEE [see also IEEE Signal Processing Magazine]*, vol. 3, no. 1, pp. 4–16, 1986. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1165342

[44] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Readings in speech recognition*, pp. 267–296, 1990.

[45] J. A. Bilmes, "What hmms can do," *IEICE - Trans. Inf. Syst.*, vol. E89-D, no. 3, pp. 869–891, 2006.

[46] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, no. 2-3, pp. 131–163, 1997.

[47] R. Kao and D. Sarigumba, *BlackBerry Storm For Dummies*. For Dummies, 2009.