An Architecture for New Ways of Game User Interaction using Mobile Devices

Jose Ricardo da Silva Junior
Mark Joselli
Esteban Clua
Federal Fluminense University

Mateus Pelegrino
Evandro Mendonça
Federal Fluminense University

Abstract

One of the most important factor in games is the players' immersion. This immersion can come from different factories, like realistic graphics, realistic physics, artificial intelligence and from the user input. Nintendo Wii controller, Sony Playstation Move controller and Microsoft Xbox Kinect, to name a few, presents new forms of user interaction using the players' movements. Also mobile phones are getting more and more features, like accelerometer, touch screen and GPS, allowing some new forms of interaction on mobile games. With that, this work presents a multi-platform architecture used to allows desktop games being controlled by mobile phones, using its built in features like vibration, sound and gesture recognition to grows up player immersion during game play. Additionally, the framework allows three types of user data input, like gesture and motion that can be used according to user preferences and desires. These available schemes is proven to be very useful and attractive for players, giving them choices that best fit their preferences and abilities. In order to demonstrate our architecture effectiveness, a 2D desktop game is developed and a study of case is made with a group of users, where some usability factors are answered by these users, as will be presented in this paper.

Keywords:: Digital games, mobile phones, user interaction, network, desktop games

Author's Contact:

{jricardo,mjoselli,esteban}@ic.uff.br
mateuspelegrino,evandro.user@gmail.com

1 Introduction

The video game industry is growing rapidly, being considered one of the most remarkable business, even in relation to the movie industry. One of its success is the rich and realistic graphics that is presented to the user, provided by the advance of graphics card in the last years. Additionally to graphics, we can observe the realistic physics achieved by games, being almost impossible to be presented in real time before, that now gets even more player's immersion in the game world.

However, this immersion can be broken in the case the players input is used in a improper way. Due to this fact, game developers and designer have to start to look for new and a more reasonable ways for input devices. Knowing that many people avoid playing games because intimidating devices, as joysticks with many buttons, one recent trend is to provide news forms of users' interaction that avoids the usage of this kind of devices, acting as more natural as possible. It is the case of devices as guitars, drums, dance pads, Kinect and so on, which can be employed by some sort of games. As result, we can see that games that uses a more natural kind of devices, like the ones cited, is more common by people who never played games in their lives.

One of the first companies to play with these new kinds of devices was Nintendo through Wii console. For this, a remote pointing devices with motion sensing capabilities is used to capture their natural body movements, gestures and pointing [Wingrave et al. 2010]. This device, called Wiimote, recognize rotations in X, Y and Z device's axes, which is in player's hand. Additionally, there is the innovative accelerometer and optical sensor technology, employed by Virual Reality for decades [Sherman and Craig 2003; Burdea and Coiffet 2003]. Also Sony presents a similar approach by presenting the Move Motion controller, which is a controller similar to the Wii controller, but with a camera to track the controller positions.

Taking another approach, Microsoft presented a system called Kinect [Microsoft 2011], which can capture player's body movement and gesture recognition to be used as game input. This device employs cameras and sensors to best capture and process this information. With this approach, Microsoft eliminated completely the needs for devices, being the player's own body used as controller, which provides one of the best intuitive game control, allowing more people to play games.

As an additional motivation, the number of mobile phones in the world, specifically in Brazil, is been growing steadily, having a prediction that by the end of 2011 the number of mobile phones in this country will reach its total number of population [Zmoginski 2010].

Contributions: we propose and evaluate a framework that enables touch screen mobile phones to be used as a wireless game controller for multiplayer desktop-based games. The framework is highly customizable, allowing virtual keys to be proper positioned in the device, in order to give the best layout to fit the desires of the player, being almost impossible to archive in the ones that uses physical keys' device of mobile phones. Three different schemes, based on *Key Touch*, *Motion* and *Touch* are implemented and evaluated using a developed 2D shooting game and a case study using some participants with minor and major experience using these kind of devices.

Paper summary: The remainder of this paper is organized as follows. Section 2 presents some related works on using mobile device for user input applied in simulations like games and we describe the framework architecture, in Section 3. In Section 4 we describe our test case simulation and the game used for it. In Section 5 we show the results and, in Section 6, the conclusions are presented. Finally, Section 7 presents future topics to research.

2 Related Work

Mobile phones have specific hardware, lots of them different from traditional game platform, like the video games and PCs. With this, mobile devices present new forms of user interaction.

Research in new way interaction devices is a hot topic for years. This is an important topic due the fact that some game applications are used to help rehabilitation or as a complementary therapy, instead of being used for funny [Golomb et al. 2010; Burke et al. 2009; Laikari 2009; Brandao et al. 2010]. In addition to conventional devices used to provide input information for the simulation, as joysticks, there is also an important topic being studied where input is made by gesture recognition, that can be divided in two broad approaches:

- sensor based, where a object are tracked by a sensor to provide data input. In this category, we can cite sensors such as gloves and motion trackers, that measure displacement in relation to a fixed coordinate system [Teichrieb and Figueiredo 2010].
- computer vision based, where tracking is done by vision algorithms. In this category, normally the user do not need anything to interact with the simulation [Stenger et al. 2010].

In [Figueiredo et al. 2009], the authors developed a framework for allowing playing a guitar game without any device, based only in gesture recognition. For this, each finger is tracked using a different color, that must contrast with the background color for best results. Unfortunately, these kind of interaction does not behave so well as playing guitar in the air, without any device to interact with, is unnatural for most players.

CamSpace [Tech 2011], a vision based project, can use almost any object for input during the simulation, using a computer vision algorithm. For its work, a pre-step calibration and recognition is necessary in order to configure the framework with the object that will be used for the simulation. Unfortunately, as many computer vision based approach, the player also needs to be in the focus of the camera, which could be a problem in cases where many peoples are going to play the game. In this case, the algorithm could get lost during the game or even takes much time for processing data. Another disadvantage of this approach is the the camera dependency for its proper work.

Mobile phones are already been proposed as input for simulations. The Poppet Framework [Vajk et al. 2008] can be used as a simulation controller that can send and receive data information to and from simulation, respectively. For this, the framework uses a bluetooth communication. Player's avatar movement is done by movement applied in mobile phone's accelerometer that is sent to a controller central server.

[Malfatti et al. 2010] propose BlueWave framework for using mobile phones as simulation input through bluetooth technology. BlueWave is implemented using Java technology, this way requiring a java enabled mobile phone. Unfortunately, the disadvantage of the proposed framework is that it uses a key's mobile phone device for control input, which can be completely different from a mobile phone to another. In this case, a layout key that could works well for a mobile phone could not work appropriately for another one, causing frustration for the player.

3 Framework Architecture

In most games, the tasks that need to be done for a real-time single player game can be divided in three general classes [Valente et al. 2005; Joselli et al. 2008; Joselli et al. 2010]:

- Data acquisition task is responsible for getting user commands from the various input devices, like: keyboard, mouse, joystick and mice;
- Data processing tasks, also referred as the update stage, are responsible for tasks that update the game state, for example: character animation, Physics simulation, Artificial Intelligence, game logic, and network data acquisition;
- Data presentation task is responsible for presenting the results to the user according to its input. In games, this corresponds usually to rendering graphics and playing audio.

This section presents our framework architecture general overview, which is separated in two main components.

3.1 General System Overview

The framework is based on two main components: the client and the server component. For this framework, cloud computing is used to allow for client and server to communicate.

To properly use our architecture, a set of steps need to be performed before the simulation actually starts. Following are a general overview of these steps performed:

1. The server component are initialized;
2. After the server initialization, it stays idle waiting connection request;
3. A client component is initialized by the user and make a connection request;
4. The server check for user authentication and establish a connection;
5. The client starts sending data to server, according to layout used in its application;
6. The server receives the data and store in its buffer;
7. The server send this data to the game application being running;
8. States are changed in the game according to data user input received.

After these input data are acquired, it is sent to the game host through network. The data received by the host is processed and game entities' state are update, according to the specific game logic. After update, these entities are finally rendered.

4 Test Case: An 2D space shotter and its unique control

In order to evaluate properly the framework an application was developed for this devices using the IOS SDK [Apple 2011]. The iPhone SDK is a development platform for the iPhones, iPads and for the iPod touch. For game development it uses smart C language, it has a 3D API based on OpenGL ES and it grant access for all the built in hardware resources. In our application we use cocoa touch to make the Key Touch mode interface. A screenshot of the application can be seen on Figure 1.

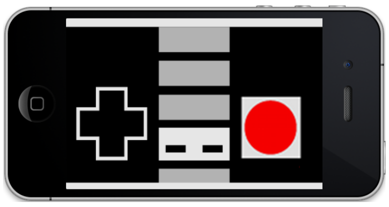

Figure 1: *Interface layout used for Key Touch mode.*

Also a game for desktop computers were made in order to evaluate the architecture. This game is a 2D prototype space shooter, where the main enhancement is that it uses a mobile device to gather the user input, allowing an unique new kind of iteration with a game.

The game play is very simple: the player plays as a space ship inside the galactic space, and he needs to destroy (by shooting them) the enemies ships from a game. Every time the space ship make physical contact with a enemy, or with a enemy shot, it looses energy. The objective is to destroy the maximum number enemy in the smaller amount of time, without running out of energy. An screenshot of the game can be seen on Figure 2.

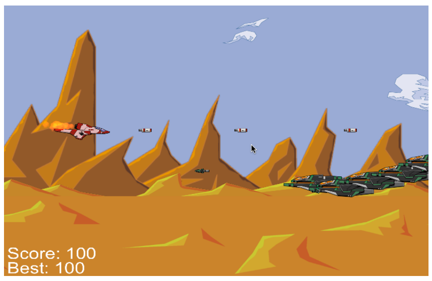

Figure 2: *Game prototype used for architecture evaluation.*

For this game, we have implemented the three possible types of input that our framework is able to process. These data input and its action in game are mapped in the following way:

- **Key Touch:** *using this scheme, the user can movement the spaceship to the right, left, up and down according to virtual keys layout presented in the device. Also, there is a key used to make the spaceship to shoot.*
- **Motion:** *motion input uses the built-in device's accelerometer. In this game, the user can movement the spaceship by moving the device in its X and Y axis. To shoot, the user needs to press a virtual key presented in the screen.*
- **Touch:** *using touch input, we allow the user to controls the spaceship through gesture that are made in the device. In this*

game, the user can movement the spaceship by using the Motion approach but shooting is made by swap gesture in device.

The game developed here shows how a unique game, in many situations, can use all the data input acquire method available, letting the user select the one that best fit its desires and abilities.

5 Results Evaluation

In order to evaluate our architecture, two mobile devices were used:

- an iPad first generation 16 GB model, which has an 1GHz Apple A4 custom-designed, high-performance, low-power system-on-a-chip CPU with 256 MB of RAM, touch screen, accelerometer, bluetooth, GPS, digital compass and wiFi ;
- and an iPod third generation, which has an ARM Cortex-A8 833 MHz (underclocked to 600 MHz) CPU, 256 MB of RAM, touch screen, accelerometer, bluetooth, and wiFi.

Two types of tests were made in order to validate the architecture. The first test mensure the performance, evaluating the impact that the architecture can have on the mobile phone, and another test for usability, to see how players react to this new kind of interaction. These tests are presented in the next two subsections.

5.1 Performance Evaluation

In most games, input data must be send almost in real time to avoid player frustration during the gameplay. This frustration may occur in cases where it press a button and this information does not arrive in time to be processed. For this case, some algorithms is used to avoid this communication latency in cases like networking games [de Carpentier and Bidarra 2005].

Due the fact that our architecture is entirely wireless based and information is sent over network, this latency is minimized by employing a fixed time rate update. Additionally, only a *keycode* is sent in order to avoid larger data being transferred over network. This *keycode* is composed using only 2 bytes, which is processed by the host game.

Table 1 shows the time, in milliseconds, spent after a data is send to it reaches its destination using all the input scheme allowed by our framework. For these tests, measure were performed by the average of running the simulation twenty times and removing the first and last measure. In this table it is possible to see that *Motion* scheme is the most expensive to be sent over network due the fact that instead of only sending one *keycode*, two data must be send, which is the current X and Y device's local axis.

Table 1: *Time spend to send data over network using all available frameworks' schemes.*

Input Type	**Time (milliseconds)**
Key Touch	18.54
Touch	17.82
Motion	22.98

5.2 Usability Evaluation

For the usability tests we have addressed the following characteristics to evaluate the framework and the new form of iteration:

- **Ergonomic (EG):** the player will test the device and give a grade in a scale ranging from 1 (very discomfortable) to 10 (very comfortable) of how he fells about its comfortable;
- **Fun factor (FF):** the player will test the game with the different device and will grade 1 (very boring) to 10 (very funny) the fun he had with it ;
- **Player choice (PC):** the player will grade this specific scheme in relation to others, ranging from 1 to 10;
- **Feedback (FB):** the user will grade the feedback he had with the device and will grade from 1 (bad) to 10 (great);
- **Time to learn (TL):** the observers will grade how difficult he had to learn the input, grading from 1 (very difficult) to 10 (very easy);
- **Player score (PS):** the game is scored accordingly to the points it made, summing all played session's points.

The game was tested with each type of controller ten times by a group of four different users. 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 mobile device (participant A), two have minor experience with such mobile devices (participant B and C) and one have none experience with the touch and accelerometer mobile phone devices (participant D). All subjects were trained for five minutes by watching the observer play the game and show the input type.

The first test consists in playing the game using the *Key Touch* scheme. Table 2 shows the results of all participants in this first phase. According to this table, it is possible to see that players who have more experience with key touch input achieved a high score in relation to others players, who have minor experience with it. Also, in this table it is possible to see how difficult it can be for a woman (participant C) playing using key touch scheme, according to it score.

Table 2: *Key Touch scheme usability evaluation.*

User	EG	FF	FB	TL	PS	PC
User A	8	10	7	10	8.200	3
User B	7	9	8	10	4.000	2
User C	9	10	7	10	1.500	5
User D	7	9	5	10	2.500	4

For the second phase, the scheme were changed to allow data input by Motion. Table 3 presents the result of this scheme. According to results presented in this table it is possible to see that the discrepancy in the score between participants was minimized, due the fact that the game input used by this scheme became more natural to players, even for the one without any experience, like the *User D*. Additionally, it is possible to see that player's score is higher than the previous scheme for all the participants, having a higher choice among all of them.

Table 3: *Motion scheme usability evaluation.*

User	EG	FF	FB	TL	PS	PC
User A	9	10	8	10	5.100	10
User B	9	10	8	10	7.400	10
User C	8	9	9	10	2.700	9
User D	8	10	9	10	3.000	10

Finally, in the third phase, Touch scheme was employed to allow data input. The results are presented in Table 4, where it is possible to see a high discrepancy in the player's score over all participants. According to this scheme, the user needs a certain ability to make the spaceship shoots, which is made by gesture. *User A*, which has a major experience among the others, achieved a high score in relation to others participants. Exception from *User B*, who achieved the minor score, this scheme also has a good *Player Choice* among the participants.

Table 4: *Gesture (Touch) scheme usability evaluation.*

User	EG	FF	FB	TL	PS	PC
User A	10	10	10	10	10.800	9
User B	4	5	6	10	1.300	3
User C	9	10	8	10	2.600	9
User D	9	10	7	10	3.000	8

Observing all results presented by each input scheme, it is possible to see that using *Motion* scheme benefits all players, ranging from the ones who have minor experience with this kind of devices to ones with major experience. In this case, a fair option during a game competition could be set to use *Motion* scheme in order to benefit the experienced ones.

6 Conclusion

New forms of user's input are being researched by industry in order to attract more players and grows up the immersion during the game play. Many times, people avoid playing games due complexity observed in many input devices, pushing them away from video games. Using a mobile phone as a game data input gives the opportunity to people that are resistant to these complexity input devices to try and possibly enjoy playing games using a device that they are accustomed to use daily-by-daily, minimizing the *Time to Learn* necessary to start enjoying a game. This could, in many situations, eliminate the enter barrier found in novice game players.

In order to demonstrate this fact, we developed a simple game to be played using the three schemes allowed by our framework. With these schemes, the user could select an approach that used virtual buttons, as the case of Key Touch scheme to one that uses any button, as the Motion scheme. This scheme selection allows player to has some control over its device usage, fact that is not possible in many others related works or even in conventional devices. Observing the results, we could confirm that users with almost any experience prefer the more natural schemes, as the case of *User D*, where its *Fun Factor* is the greater when using the Motion scheme.

Additionally, using mobile phones for game data input can eliminate the use of proprietary game controllers, which can encompass more players, requiring only a mobile phone to play a game.

7 Future Works

One important aspect for future work is allowing image data to be sent from the host game to device. This feature allied with Motion scheme could allow compelling game play, as the screen device could be seen as an extension of the game word, presenting vital information during the game. One drawback that must be solved is the fact that images have large amount of data, which could grows up the network latency. The authors of this work also pretend to further develop the client in order to allows others kind of user input, like, camera, voice and geo-location.

References

APPLE, 2011. ios development center. [Online; accessed 02-August-2011].

BRANDAO, A., TREVISAN, D., BRANDAO, L., MOREIRA, B., NASCIMENTO, G., VASCONCELOS, C., CLUA, E., AND MOURAO, P. 2010. Semiotic inspection of a game for children with down syndrome. In *Games and Digital Entertainment (SBGAMES), 2010 Brazilian Symposium on*, 199 –210.

BURDEA, G. C., AND COIFFET, P. 2003. *Virtual Reality Technology*. Wiley-IEEE Press.

BURKE, J. W., MCNEILL, M., CHARLES, D., MORROW, P., CROSBIE, J., AND MCDONOUGH, S. 2009. Serious games for upper limb rehabilitation following stroke. In *Proceedings of the 2009 Conference in Games and Virtual Worlds for Serious Applications*, IEEE Computer Society, Washington, DC, USA, VS-GAMES '09, 103–110.

DE CARPENTIER, G. J. P., AND BIDARRA, R. 2005. Behavioral assumption-based prediction for high-latency hiding in mobile games. In *Proceedings of the 7th International Conference on Computer Games*, IEEE Computer Society, Angoulme, France, CGAMES'05, 83–88.

FIGUEIREDO, L. S., TEIXEIRA, J. A. M. X. N., CAVALCANTI, A. S., TEICHRIEB, V., AND KELNER, J. 2009. An open-source framework for air guitar games. In *Proceedings of the 2009 VIII Brazilian Symposium on Games and Digital Entertainment*, IEEE Computer Society, Washington, DC, USA, SBGAMES '09, 74–82.

GOLOMB, M. R., MCDONALD, B. C., WARDEN, S. J., YONKMAN, J., SAYKIN, A. J., SHIRLEY, B., HUBER, M., RABIN, B., ABDELBAKY, M., NWOSU, M. E., BARKAT-MASIH, M., AND BURDEA, G. C. 2010. In-home virtual reality videogame telerehabilitation in adolescents with hemiplegic cerebral palsy. *Archives of Physical Medicine and Rehabilitation 91*, 1, 1 – 8.e1.

JOSELLI, M., ZAMITH, M., VALENTE, L., CLUA, E. W. G., MONTENEGRO, A., CONCI, A., FEIJ, B., DORNELLAS, M., LEAL, R., AND POZZER, C. 2008. Automatic dynamic task distribution between cpu and gpu for real-time systems. *IEEE Proceedings of the 11th International Conference on Computational Science and Engineering*, 48–55.

JOSELLI, M., ZAMITH, M., CLUA, E., LEAL-TOLEDO, R., MONTENEGRO, A., VALENTE, L., FEIJO, B., AND PAGLIOSA, P. 2010. An architeture with automatic load balancing for real-time simulation and visualization systems. *JCIS - Journal of Computational Interdisciplinary Sciences*, 207–224.

LAIKARI, A. 2009. Exergaming - gaming for health: A bridge between real world and virtual communities. In *Consumer Electronics, 2009. ISCE '09. IEEE 13th International Symposium on*, 665 –668.

MALFATTI, S. M., DOS SANTOS, F. F., AND DOS SANTOS, S. R. 2010. Using mobile phones to control desktop multiplayer games. In *Proceedings of the 2010 VIII Brazilian Symposium on Games and Digital Entertainment*, IEEE Computer Society, Washington, DC, USA, SBGAMES '10, 74–82.

MICROSOFT, C., 2011. Introducing kinect for xbox 360. [Online; accessed 15-July-2011].

SHERMAN, W. R., AND CRAIG, A. B. 2003. *Understanding Virtual RealityInterface, Application, and Design*. Morgan Kaufman.

STENGER, B., WOODLEY, T., AND CIPOLLA, R. 2010. A vision-based remote control. In *Computer Vision: Detection, Recognition and Reconstruction*. 233–262.

TECH, C.-T., 2011. Camspace from cam-trax technologies. [Online; accessed 15-July-2011].

TEICHRIEB, V., AND FIGUEIREDO, L. S. 2010. Interao em realidade virtual e realidade aumentada. In *Livros de minicursos do Brazilian Symposium on Virtual and Augmented Reality 2010*. 9–22.

VAJK, T., COULTON, P., BAMFORD, W., AND EDWARDS, R. 2008. Using a mobile phone as a wii-like controller for playing games on a large public display. *Int. J. Comput. Games Technol. 2008* (January), 4:1–4:6.

VALENTE, L., CONCI, A., AND FEIJÓ, B. 2005. Real time game loop models for single-player computer games. In *Proceedings of the IV Brazilian Symposium on Computer Games and Digital Entertainment*, 89–99.

WINGRAVE, C. A., WILLIAMSON, B., VARCHOLIK, P. D., ROSE, J., MILLER, A., CHARBONNEAU, E., BOTT, J., AND JR., J. J. L. 2010. The wiimote and beyond: Spatially convenient devices for 3d user interfaces. *IEEE Computer Graphics and Applications 30*, 71–85.

ZMOGINSKI, F., 2010. Brasil fica perto de 1 celular por pessoa. info online. [Online; accessed 15-July-2011].