ADDRESSING RESPONSIVENESS IN INTERACTIVE, PERVASIVE GAMES

Davit Stepanyan†*, Ani Nahapetian**

†Jet Propulsion Laboratory (JPL), *California State University, Northridge (CSUN)
davit.stepanyan@jpl.nasa.gov, ani@csun.edu

ABSTRACT

This paper presents a mobile, infrastructure-based game, which coordinates the networked interaction between pairs of players on mobile devices and a server-based visualization and processing component. There Is No Ball (TINB) system is a web and mobile based game system, which allows users to play a game of paddle board without a physical ball. A common instance of the game involves two players using tablets as paddles, to gather and communicate sensor information and simulate the hitting of an imaginary ball back and forth between the two devices. A third device interfaces with the server and the WebGL-based ball visualization to display the calculated location of the ball and project its moving image.

The key challenge of game responsiveness in a highly-interactive game dependent on communication with a remote server and WebGL visualization is addressed. Specifically, local filtering strategies are applied to improve the responsiveness of actuation, and to deliver an enjoyable game experience.

Index Terms— Mobile devices; sensor fusion; pervasive games; motion visualization, WebGL

1. INTRODUCTION

The There Is No Ball (TINB) system is an interactive infrastructure-based pervasive game of paddle-board using Android-enabled smart phones or tablets. It provides a virtual environment for hitting a ball back and forth between two users. Each game session is associated with a web view that visualizes the motion of the ball, in real-time. Figure 1 summarizes the front-end system architecture, including the two tablets and the third machine which can project the moving ball.

The TINB system was developed and used as a platform for investigating responsiveness in pervasive games, which collect and process sensor data from the mobile device, and use that data for an interactive game among pairs of players. Our analysis of TINB and its paddle board game highlights the benefits of user-side data filtering to address this system-level challenge.

The realization of the game system contains several key challenges:

- The fusion of sensor data from the user tablet pairs, including pitch, tilt and acceleration data from each tablet;
- The real-time calculation of the ball position, and the transmission of select data to and from both tablets, in a timely manner;
- The development of an attractive game interface, whose responsiveness is deemed acceptable by system users.

The TINB system implementation interfaces with a variety of software and middleware frameworks. A native Android app was created for use on each of the user tablets. Sensor data is obtained from each of the devices and actuation was also carried out throughout the game. The continuous sensor data collected undergoes some local filtering, before being transmitted to a remote server that carries out the majority of the ball motion calculation. The data calculation on the server-side is tied in with a WebGL framework for a user-friendly and fun interface for displaying the imaginary ball with an HTML 5 enabled browser. The information about hits is relayed back to the tablet, also, for display on the user side.

Fig. 1. Front-end system architecture of the TINB system. Two players play a game of paddle board with respective tablets. The data is relayed between the tablets and a third device, which can be connected to a projector to project the ball motion.

The remainder of this paper is organized in the following manner. Section II presents the related work in this area, including mobile games, pervasive games, and interactive mobile games. Section III describes the TINB system with its tablet view and web view. Section IV addresses

implementation issues and describes how the ball motion is calculated from the collected sensor data. Section V discusses the data filtering that was carried on the user-side to address the challenge of responsiveness. Section VI presents the results of the user study of the system and the results obtained regarding its responsiveness. Finally, section VII concludes the paper.

2. RELATED WORK

Interestingly, the first computer game, developed in 1958, was Tennis for Two [1]. Gaming on mobile devices is ever popular and growing [2], with an important class of mobile games including pervasive games [3] that use on-board sensor technology. Examples of such games include those developed on the Poppet framework [4], which use sensors on mobile devices including accelerometers, camera, and RFID. Games on this framework include Mobi-tron (the classical Tron game played with a mobile phone) and TiltRacer (a 2D racing game) [4]. Other pervasive games include CatchBob! [5] and Human Pacman [6]. There are augmented reality (AR) games, as well, which interact with the environment, such as the Alchemists and TimeWarp [7].

Interactive mobile games, involve the interaction of several players on different mobile devices, including infrastructure-based or cloud-based games surveyed here [8]. Emerging classes of interactive mobile games are games on vehicular networks (VANETs) [9] and games for improving personal health [10] [11].

In this work, we use the TINB system to address system-level issues with pervasive and interactive mobile games, with an infrastructure-based model of communication. With our paddle board implementation, we develop a platform for accessing the validity of addressing responsiveness in this environment with data filtering.

3. SYSTEM DESCRIPTION

The TINB system is composed of mobile devices connected to simulate a game of paddle board. Each user takes a tablet, which acts a paddle in the game. The tablets are swung just as paddle would be, and the virtual ball is hit back and forth between the players. Currently, at most two players can play the game. In the case of individual play, the ball is hit against an imaginary wall.

Each game session is associated with a web view that visualizes the motion of the ball, in real-time. Ideally, the screen can be projected onto a wall, to simulate the appearance of a real ball being hit between the players.

3.1. Game Play

At the start of each game, users select single or two-person game play, which player they will be in the case of two-person play, and which level of difficulty they would like to play. Players hit the ball back and forth between each other. If the tablet swing misses the ball, the ball falls to the ground. The ball is then returned to the opposing player and a new round of play begins. If the game is restarted, then the ball returns to player 1's paddle.

3.2. Tablet Interface

During game play, users swing the tablet as they would a physical paddle. The timing of the hit, the orientation of the tablet, and the speed at which the ball is hit are all factors in determining the motion of the ball. When the ball is hit successfully, the tablet provides haptic feedback to the user and a "HIT" message is displayed on the screen. Figure 2 provides the screen shot of the tablet view of the game.

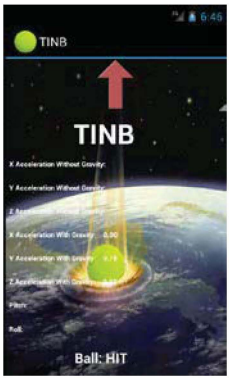
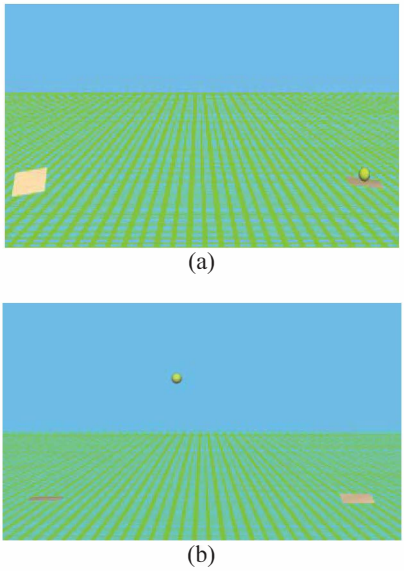


Fig. 2. The screen shot of the tablet interface during a hit



(a)

(b)

Fig. 3. The web-interface for a mobile device enabled paddle board game. Two tablets hit an imaginary ball back and forth, while the location of the ball is displayed on this web interface. Figure 3(a) shows the starting position of the ball, along with the tablet orientation; and Figure 3(b) shows the ball after a hit.

3.3. Web Interface

The web interface of the game is created ostensibly to display the motion of the ball. Players can access the web interface from a web browser, and even project the image to the screen during game play, further enhancing the experience.

The web interface is implemented using PHP and Javascript, interfacing with WebGL for a graphically attractive view. WebGL requires HMTL 5 for the WebGL processes to be rendered on the HTML canvas. (We have prepared a simpler alternative HTML page, for browsers not HTML 5 enabled.) Information on the screen is updated using AJAX calls, which allows for dynamic page updates upon data changes, instead of the more cumbersome page refreshes at set time intervals. Figure 3 shows two snapshots of the web view during game play.

4. IMPLEMENTATION DETAILS

The system back-end, summarized in Figure 3, involves filtered data being transmitted from each of the user tablets to a remote server, once a threshold value for a move is detected. The data is processed on the server, information is then transmitted to the tablets to provide feedback to users in the case of a hit of the ball. Furthermore, the ball motion data with the graphical interface is available through a web browser.

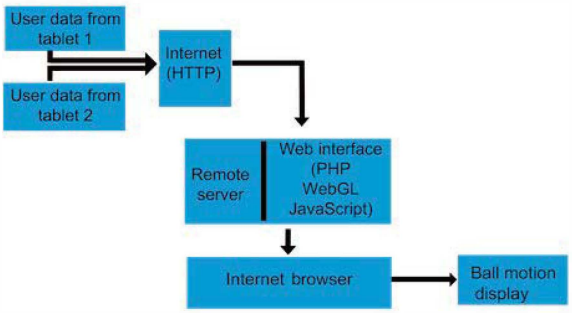


Fig. 4. Back-end system architecture of the TINB system. Data is filtered locally and transmitted to a remote server where calculations of the ball motion are carried out.

4.1. Sensor Fusion

Sensor data, namely pitch, roll, and x-, y-, and z-acceleration, is collected form the individual tablets. The identification of the various potential movements of the tablet are summarized in Figure 5. We disregard the y_{aw} values, since the rotation of the tablet is not relevant to the ball motion calculation. However, pitch and rolls values are used to determine the tablet position when a swing is carried out. The absolute position of the tablet is not used in the game play.

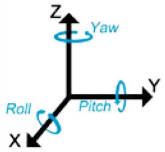


Fig. 5. Tablet movement parameters: Roll, Pitch, and Y_{aw}.

4.2. Ball Motion Calculation

Projectile motion calculations are used to determine the trajectory of the ball throughout the game play. Figure 6 demonstrates how the ball would move in relation to the tablet if the ball were physically being hit by the tablet. The tablet and the ball move together at the beginning of the swing (point A and point B). At some point C they are separated.

Fig. 6. Model of tablet hitting ball over time.

The initial velocity of the ball at point C is given by Equation (1).

$$\overrightarrow{V_{PointC}} = \overrightarrow{V_{PointA}} + \vec{a}t \quad (1)$$

Assuming that the tablet starts from a resting position, i.e. its initial velocity at point A is 0, then $\overrightarrow{V_{PointA}} = 0$, Equation (1) can be re-written as Equation (2).

$$\overrightarrow{V_{PointC}} = \vec{a}t \quad (2)$$

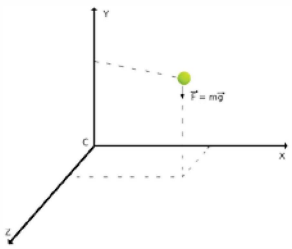


Fig. 7. Ball coordinates with origin at point C.

As illustrated in Figure 7, the position (x, y, z) of the ball is calculated relative to the point C. For the x-coordinate:

$$x = \vec{V_0}T = V_{0x}tT = V_{Point\,C\,x}tT = a_x tT \quad (4)$$

Where a_x is the x-component of the tablet acceleration at point C (collected from the tablet's accelerometer). Similarly for the z-coordinate:

$$z = \vec{V_0}T = V_{0z}\,tT = V_{Pointc\,z}\,tT = a_z tT \quad (5)$$

We ignore the impact of air friction on the ball, as it accounts for a negligible difference. Therefore, the velocity in x- and z-directions is constant. The velocity in the y-direction of the ball, however, changes due to gravity. Thus

$$y = \vec{V_0}T + \frac{\vec{a}T^2}{2} = V_{0y}\,tT + \frac{a_y T^2}{2} = V_{oy}\,tT + \frac{g_y T^2}{2} = V_{0y}tT - \frac{gT^2}{2} \quad (6)$$

Using equation (2),

$$y = V_{Pointc\,y}\,tT - \frac{gT^2}{2} = a_y tT - \frac{gT^2}{2} \quad (7)$$

Thus, at any given time T, ball will have the following position:

$$(x_0 + a_x tT,\ \ y_0 + a_y tT - \frac{gT^2}{2},\ \ z_0 + a_z tT) \quad (8)$$

4.3 Handling Difficulty Levels with Precision Loss

The different difficulty levels of the game are created by varying the precision of the tablet position and swing intensity necessary to land the ball on the other player's tablet. The point location of the ball at the end of the projectile motion is expanded and converted into an area, whose radius is determined by the difficulty setting of the game. If an overlap between the generated area and the opponents tablet exists, then the ball is assumed to have landed on the opponent's tablet. This calculation is done before the motion of the ball is displayed, thus preserving the quality of graphical interface.

For none of the difficulty levels is the necessary timing modified. In the easy level, if the user hits the ball at the right time then the ball lands on the opponent's tablet, as the radius of the area at that level is infinite. For progressively harder levels, the radius of the area is proportionally decreased.

5. DATA FILTERING FOR RESPONSIVENESS

The most significant implementation challenge involves overcoming issues of responsiveness, specifically the delay in displaying ball motion using data from the swing of the tablets. This delay is addressed by filtering the swing data on the user's device.

To carry out the filtering, the acceleration values in the x-, y-, and z-directions are evaluated against threshold values. If the threshold is met in either case, a session is started with the server and all relevant filter sensor data is transferred. Figure 8 demonstrates the quantity of sensor data collected from the tablet before and during a swing, and Figure 9 provides the accelerometer data used to signal that the server code should start computing the ball motion.

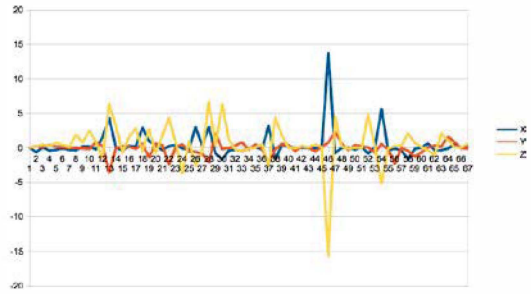


Fig. 8. Accelerometer values over time for swing of the tablet, as obtained from the tablet.

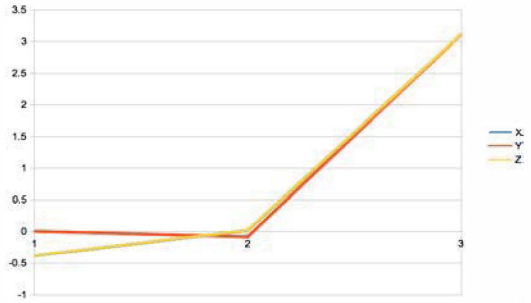


Fig. 9. Accelerometer values over time for swing of the tablet, transmitted to the server to calculate the ball motion.

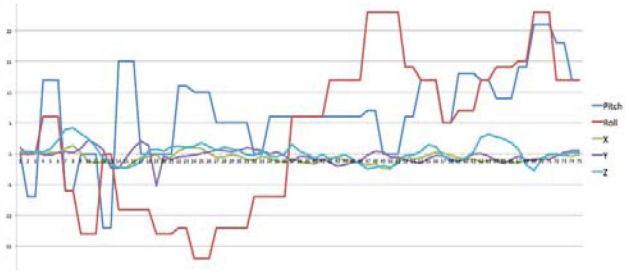


Fig. 10. All sensors values during a swing of the tablet, once a threshold value is reached, as obtained from the tablet.

Fig. 11. Accelerometer values during a swing of the tablet, processed at the server to calculate the ball motion, once a threshold value of change is reached.

Figure 10 demonstrates the quantity of sensor data collected from the tablet during a swing, and Figure 11 provides the filtered data transmitted to the server to compute the ball motion.

The small variations in sensor data collected from the tablet, if transmitted to the server, would result in small movement calculations for the web interface display, unnecessarily slowing down the game play. Instead, as shown in Figure 11, accelerometer, pitch, and roll data is transmitted to the server, if the accelerometer values in any of the three axes are above a threshold. Otherwise, the ball motion and the table movement are extrapolated from the previously recorded sensor values.

6. USER STUDY

To determine the attractiveness of the game to potential users a small user study was carried out, with specific attention being given to game responsiveness. The study included 10 users, three 3 women and 7 men, all between the ages of 20 and 35 years old. Each of the participants was given a 1 minute introduction on how to play the game. They, then, were paired together to play the game. Immediately following the game play, they were given an anonymous survey to complete.

TINB USER SURVEY	(1 VERY LOW : 5 VERY HIGH)
Likelihood of recommending TINB to others	1 2 3 4 5
Likelihood of using TINB in the future	1 2 3 4 5
Level of fun while using TINB	1 2 3 4 5
Level of responsiveness during game play	1 2 3 4 5
Sense of playing a real paddle game	1 2 3 4 5
Overall satisfaction with game	1 2 3 4 5

Fig. 12. Survey questions used in user study.

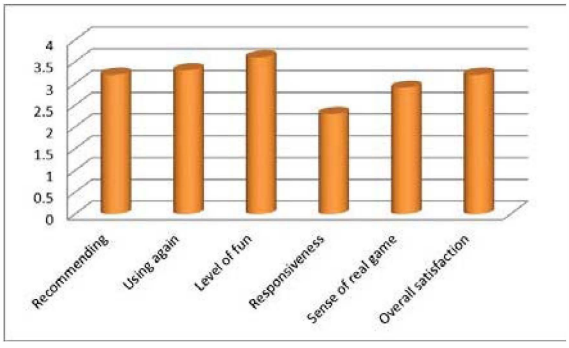


Fig. 13. Results of the survey administered to 10 participants.

The survey was composed of the 6 questions provided in Figure 12, with participants rating features of the game including responsiveness, similarity to physical game play, and overall satisfaction. The results of the survey are given in Figure 13. The results are promising, especially in terms of fun with game play, which averaged to 3.6 out of 5.

7. CONCLUSION

The TINB mobile game system presented in the paper is composed of various mobile devices wirelessly connected to simulate a game of paddle board. Users swing tablets to move a virtual ball is hit back and forth between the players. The approach to filtering the data on the user side, to improve the game's responsiveness during game play, is highlighted, along with the encouraging results of the user study.

8. REFERENCES

[1] J. Anderson, "Tennis for two: the story of an early computer game," Accessed on 15 April 2012. http://www.pong-story.com/1958.htm.

[2] T. Fritsch, et al, "Mobile phone gaming (a follow-up survey of the mobile phone gaming sector and its users)," *ICEC'06*, pp. 292-297, 2006.

[3] D. Schuster, et al, "Pervasive social context-taxonomy and survey," *ACM Trans. Intell. Syst. Tech.*, Vol. 9, No. 4, Art. 39, pp. 1-22.

[4] Y. Lee, et al, "MobiCon: a mobile context-monitoring platform," *Comm. of ACM*, vol. 55 no. 3, pp. 54-65, March 2012.

[5] O. Akribopoulos, et al, "Developing multiplayer pervasive games and networked interactive installations using ad hoc mobile sensor nets," *ACM ACE '09*, pp. 174-181, 2009.

[6] A. D. Cheok, et al, "Human pacman: a sensing-based mobile entertainment system with ubiquitous computing and tangible interaction," *ACM NetGames '03*, pp. 106–117, 2003.

[7] W. Broll, et al, "Toward Next-Gen Mobile AR Games," *IEEE Comp. Graphics and Appl.*, vol. 28, no.4, pp. 40-48, July-Aug. 2008.

[8] M. Gerla, et al, "A survey on interactive games over mobile networks," *Wiley Wireless Comm. Mob. Comp.*, vol. 13, no. 3, Feb. 2013.

[9] O. K. Tonguz and M. Boban. "Multiplayer games over vehicular ad hoc networks: A new application," *ACM Ad Hoc Networks*, vol. 8, no. 5, pp.531-54, July 2010.

[10] J. Woodbridge, et al, HIP: Health Integration Platform. *IEEE PerHealth '10,* March 2010.

[11] M.-K. Suh, et al, "Bayesian networks-based interval training guidance system for cancer rehabilitation," *MobiCASE '09*, Oct. 2009.