Evaluating and Customizing User Interaction in an Adaptive Game Controller

Leonardo Torok(✉), Mateus Pelegrino, Jefferson Lessa, Daniela Gorski Trevisan, Cristina N. Vasconcelos, Esteban Clua, and Anselmo Montenegro

Computing Institute, Federal Fluminense University, Rua Passos Da Pátria 156 – E – 3rd Floor, São Domingos, Niterói, Brazil
{ltorok, daniela, crisnv, esteban, anselmo}@ic.uff.br, {mateuspelegrino, jefferson2459}@gmail.com

Abstract. When playing a game, the user expects an easy and intuitive interaction. While current game console controllers are physical pre-defined hardware components with a default number, size and position of buttons. Unfortunately, different games require different buttons and demand different interaction methods. Despite that, the play style of each player differs according to personal characteristics (like hand size) or past gaming experiences. To achieve an optimal controller configuration for each player, this work proposes a virtual controller based on a common touchscreen device, such as smartphone or tablet, that will be used as a joystick to control a game on a computer or console, collecting user input data and applying machine learning techniques to adapt the position and size of its virtual buttons, minimizing errors and providing an enjoyable experience. With the prototype controller, tests were performed with a set of users and the collected data showed considerable improvements in the precision and game performance of the players.

Keywords: Adaptive interfaces · Adaptive game control · Machine learning · User behavior · Mobile · Touchscreen

1 Introduction

Gameplay experience is one of the most important aspects for achieving the fun factor [1]. This process includes several characteristics that will determine the final impression and engagement of the user. Among many aspects, the challenges must be correctly balanced or the user can get frustrated by excessive roughness or become bored by the absence of challenge [2]. The gameplay experience is implemented by the game rules, which is executed by the player through its interface. In this sense, the interface must be engaged with the user experience, stimulating the user to interact.

While several games became infamous because of unresponsive controls or clunky and unintuitive interface solutions, others are remembered for the opposite: fast, responsive, intuitive, efficient and innovative control schemes, creating an enjoyable experience that made those games memorable. With the rise of smartphones, quickly followed by tablets, a gaming revolution was bound to happen. The computing power

A. Marcus (Ed.): DUXU 2015, Part III, LNCS 9188, pp. 315–326, 2015.
DOI: 10.1007/978-3-319-20889-3_30

on these mobile devices quickly improved, allowing the launch of several mobile games with complex visuals and refined interactions. A major difference was that these devices rely primarily on touch input, done on a large screen that normally covers the entire front of the device. These new technologies, both on mobile devices and in traditional gaming machines, were the starting point for a new wave of games using more intuitive control schemes, bringing millions of players that aren't interested in learning complex control schemes to electronic games. Although traditional controllers may be complex, they are reliable and precise. The new methods suffered with lack of precision in command input, slower reaction time and limitations when trying to map complex actions. The problem was clear in several mobile games, which started using virtual buttons displayed on its touchscreen to simulate a traditional controller and other input options. Among different problems of this solution, the lack of tactile feedback of a real controller is an important issue [2].

In this work we propose a new interface for electronic games, that has the large input options of a regular gamepad but with the flexibility of a virtual controller. This new input method aims to be an alternative to the traditional video game joystick with a touchscreen device with a virtual controller on its screen, creating a less cluttered and more intuitive interface to play a video game. In order to solve the lack of precision and consequently more mistakes, our solution introduces a novel control adaptation to the user's behavior. Based on an observation of basic interaction events of a specific user, such as button presses, speech input, or internal state changes, user preferences are derived. Different algorithms extract information from these basic events, such as preferences of the user or a prediction of the most likely following user action. The proposed adaptive interface employed machine learning algorithms to analyze user inputs and detect errors. After detecting how the user is missing the virtual buttons, the controller will try to smoothly fix the position and size of the buttons in order to eliminate or reduce touch errors.

2 Related Work

Mobile phones have specific hardware (camera, accelerometer, GPS, Bluetooth, Wifi) with lots of them being different from the ones found in traditional game platforms, like video games and PCs. For this reason, these devices bring new forms of user interaction however with the drawback of lacking tactile feedback [3]. On the other hand an adaptive user interface is an interactive software system that improves its ability to interact with a user based on partial experience with that user [4]. Adaptation of interactive systems describes changes of the interface that are performed to improve the usability or the user satisfaction.

Rogers et al. [5] developed models that treat uncertain input touch and use this to deal with the handover of control between both user and system. They demonstrate a finger map browser, which scrolls the map to a point of interest when the user input is uncertain. Keeping the same goal, but in a different way, Weir et al. [6] used Machine Learning approach for learning user-specific touch input models to increase touch accuracy on mobile devices. They proposed mapping data or touch location to the intended touch point, based on historical touch behavior of a specific user.

Bi et al. [7] conceptualized finger touch input as an uncertain process, and used statistical target selection criterion. They improve the touch accuracy using a Bayesian Touch Criterion and decrease considerably the error rate. However, even improving the accuracy in a higher level, the user keeps missing the buttons and the interface needs to calculate the intended target.

While personalized inputs have been commonly used, such as key-target resizing on soft keyboards [8], this type of adaptability has some disadvantages, as the needs of using only on meaning of process language, and it just adapt according to the model of language and of user typing behavior.

The use of touchscreen devices as gaming controls is relatively new, resulting in very few related works, with the most relevant ones covered in this section. When looking specifically at mobile phones with adaptive interfaces being used as gaming controls it is clear that this is an area that still remains unexplored.

3 Proposed Adaptive Interface

The proposed game control interface is composed of both a physical hardware (touchscreen device, in this case an Android smartphone) and software components that will observe the user behavior and adapt the interface components. As smartphones and tablets are common devices, they will be used as a physical component, allowing the player to use devices he already has to play games.

The touchscreen device, our first component, will represent the joystick. A mobile application was created, presenting an interface similar to a real controller and sending all button presses to the computer that runs the game. A second software, running on the PC, will receive the information about key presses and generate the corresponding keyboard events on the computer.

3.1 Control Adaptations to the User's Behavior

With our proposal, game developers may also project a specific interface for their game. However, due to time limitation, it is possible that the developer can't develop many usability tests in order to validate and adjust his interface. In our work we developed an automatic and interactive adaptation mechanism, based in machine learning concepts, in order to improve the usability or the user satisfaction and facilitate the developer process. Adaptations may be triggered by different adaptation causes, such as the context of the interaction, the experience of the user, or user behavior [9]. In this work, we focus on the adaptation of interactive game control to user behavior. From a philosophical point of view, [10] defines behavior as an internally produced movement and contrasts behavior with other movements that are produced externally. For instance, raising an arm is behavior, whereas having an arm raised by someone else is not. In this point of view, reflexive and other involuntary movements are considered as behavior. Thus, behavior does not necessarily have to be voluntary and intentional. Reflexives movements can become very common in some kinds of games where the user should perform movements of the virtual character many times quickly.

To achieve the desired adaptation to each user's play style we decided to perform two different changes in the controller's layout: size and position of buttons. These

adaptations are performed at the same time for each button. The size adaptations aims to facilitate the usage of the most important buttons for the game being played, increasing the size of the most used buttons in the controller and decreasing the size of the buttons that are less used. The controller retains the data for each button press and uses this data history to determine the optimal size for each button, following maximum and minimum limits on size (predetermined as two times the original size of the button and half of its original size). The controller is also capable of reacting to changing requirements. If a button that had minimal use start being used constantly, the controller will learn with the user interaction and start increasing the size of this button. This change will be done once per iteration, but limiting for this work, by a maximum of 3 pixels each time, in order to have a more organic adaption. The controller will perform one modification per second, so it does not change too fast.

The second type of adaptation is the button's position. The basic concept here is try to detect the position for a specific button that will guarantee that the majority of users touches actually hit the button (remembering that the user will not have the physical feedback of a real controller and will not be able to keep looking at the controller to check if his fingers are hitting the correct spot of the buttons during gameplay). To perform this adaptation, we will use all the data about users' touches on the screen (both correct touches and wrong ones) and try to detect the points that represent more accurately the center of the position where the user is trying to press. After determining the positions, each one will be correlated with the closest button. It is necessary to note that some buttons may not have a new center assigned, indicating that, for now, they will not need any adaptation (this may occur when a button did not had touches or when the amount of touches in its area is insignificant compared to the full dataset of inputs). With the correlation done, the application on the smart device will start to move the button for the new correct position. This movement is done moving the button at maximum 5 pixels (considering the distance in the x and y axis) per iteration, with an interval of one second. Figure 1 shows the adaptations correcting a button's position.

To keep the interface consistent, several rules and boundaries were specified. For the changes in size, it was determined that each button could reach a maximum size of two times its original size, with a minimum limit of half its size. This rule will avoid buttons with an exaggerated size or simply too small to use. Despite that, it is necessary to deal with the presence of other buttons in the surrounding areas. If a button moves or grows in conflict with a neighbor button, the user would not be able to use the interface. To avoid that, before each button size increase, the controller will verify if its new position and size is in conflict with any of the other buttons (considering that a conflict is defined as any intersection in the area of any pair of buttons). If a change results in conflict, it will not be done. This means that if the center for the input data assigned to a button is very close to another button, in a manner that moving the button until its center is located at that point would result in an intersection, the controller will move the button as close as possible to the correct position without intersecting both buttons. If the neighbor button that created the conflict moves out of the way (something that can happen because of the adaptations performed on the neighbor), then the algorithm will continue the adaptation since the conflict is not happening anymore. With this rules, the controller will always maintain a consistent interface.

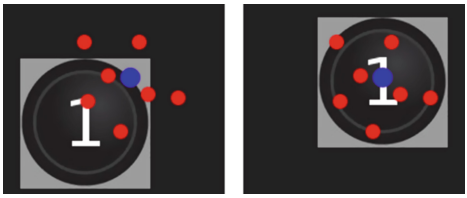

Fig. 1. The red dots represent the player's touches in the screen while the blue dots corresponds to the centroid for the cluster assigned to the button "1", the correct position. In the left side, the controller is on its default configuration. The controller will gradually move the button "1" until it reaches the position demonstrated in the right side of the figure.

3.2 Machine Learning

With the two adaptations approaches defined, it is necessary to define the method to gather user input data and translate it to meaningful results to improve the controller responding to the player's needs. The data used to evaluate the controllers correctness is a set with the entire touches log in the screen performed by the user. Each touch can be classified as correct or incorrect. A correct touch is defined as an interaction that hits any button in the screen and performs an action, while an incorrect one is defined as an action that does not hit any button, representing a situation where the user tried to perform an action and failed. To collect the necessary information about input, the mobile application uses an extra control over the interface, an invisible layer that will intercept the touches, collect data and redirect the touch for the buttons above. This layer covers the entire screen area and is mapped in Cartesian coordinates, with the y-axis representing the screen height and the x-axis representing the screen width, considering the device in landscape mode. The origin of the coordinate system is located at the upper left corner, following the convention used by the Android OS when mapping screen input. Each touch in the screen is saved in the internal SQLite database of the Android device (both the correct and the incorrect inputs) and the machine learning algorithm will use the most recent data from the database. In our tests, the algorithm always used the last 30 points. This approach allows the controller to change its behavior in response to a change in the player's needs. Both operations occur simultaneously, with the controller constantly inserting new input data in the database and the algorithm collecting the data and processing it at each iteration.

With the data being collected, the next step was defining when and how to perform adaptations. The size adaptation is the simplest case, detailed in the last section. The size will increase for the most used buttons and decrease in the least used ones. In this first case, a simple algorithm that tracked and counted the amount of times each button was used was enough, allowing the controller to easily determine which buttons would need to increase and which would need to decrease its size. The algorithm uses the list of buttons ordered by the amount of presses. This list is divided in 3 parts, with equal size. The first one contains the most used buttons while the last one contains the least used. The middle set will not suffer any size change. The buttons in the first set will start to increase at each iteration, while the buttons in the last group will start decreasing, respecting the boundaries detailed in the last section. At each iteration, the

list will be updated with the user inputs and the order may change. As an example, during one iteration, the button A can be the most used, starting to increase on size. After a set of iterations, it can fall to the middle group, halting it's increase. If, after a defined delta time, the A button becomes less necessary and ends in the last group, it will start to decrease.

While the size adaptation only demanded a simple algorithm, the position correction needs a more sophisticated approach. The main objective is to find the points in the screen that represent the center of the most used areas. The objective of using the input data as points in the screen space and group the data according to its positions, finding the center for each group can be correctly modeled by the K-means unsupervised learning algorithm, which is a vector quantization method used in data mining that receives an entry set and separates these entries in K classes of co-related entries. The algorithm is NP-hard, but with several heuristics that allow a quicker solution that converges surprisingly fast to a local optimum. Given a set $X = \{x_1, \ldots, x_m\}$ the goal of K-means is to partition it into k clusters such that each point in a cluster is more similar to points from its own cluster than others from different clusters [11].

In our case, the entry set is composed by the most recent user inputs, represented as Cartesian coordinates. The objective was to separate it in K classes, with K being the number of virtual buttons in the screen. The algorithm will return several subsets of the full input dataset containing related points grouping the closest points in the same class. For each class, the K-means clustering will also detect the class centroid, which is the nearest point to all other touches that corresponds to that cluster, since it corresponds to the optimal position for a button. The users' touches will be located in the area of a button or at least close to it, since the player's objective is to hit the correct position of the button to perform actions in the game. After several interactions it is possible to observe a pattern of inputs close to each button, allowing the K-means clustering to separate inputs in classes that represent each buttons area and some of the space around it. After finding the centroids, each one will be paired with the closest button. In the case where there are two centroids where the closest button is the same, the correct pair will be created based on the minimal distance, with the other centroid remaining without an assigned button, which is a more common situation in the initial usage, based on a low amount of touch data. The paired centroid related to the button will be considered as the optimal position for that element, since it represents the middle point of all inputs directed to that button. With this data, the controller will start to move the button gradually towards the centroid of its class, until the center of the button is located precisely in the centroid for that class.

4 Usability Tests

In order to investigate our adaptation approach and the user gameplay experience, we conducted a user interaction evaluation for the adaptations presented before. These users' tests examine the effectiveness and satisfaction of the adaptations and provide empirical evidence for the application of the adaptations in the domain of game control.

4.1 Participants and Apparatus

The evaluation used two different interactive systems (one adaptive and other not adaptive) and two games genre: a platform game (Super Mario Bros) and an arcade game (Streets of Rage).

Platform games involve traveling from one direction to another, avoiding obstacles, enemies and jumping platforms (very occasionally other means are substituted for jumping, like swinging or bouncing, but these are considered variations on the same mechanic). They are most often associated with iconic video game mascots like Donkey Kong, Sonic the Hedgehog, Mario, Megaman, Samus, Crash Bandicoot and Rayman, though platform games may have any theme. Arcade games can involve several variations, being common in the 1990 s. Usually, the player must travel from one screen to another, defeating all enemies presented. Streets of Rage and Final Fight are two examples of this genre.

Fig. 2. The user interaction setup used during the evaluation sessions

A total of 16 users tested two different version of the control with two different games, totalizing four evaluation sessions for each user. Our group of volunteer participants consisted of 13 male users and 3 female users with ages ranging from 18 to 50. The user group was selected among expert computer and smartphone players. The experiment was conducted on a Motorola Moto X phone running Android OS 4.4. The capacitive touchscreen was 4.7 inches in diagonal with an aspect ratio of 9:16 and a resolution of 720 × 1280 pixels. When a finger touched the screen, the approximate centroid of contact area between the finger and the screen was reported as the touch point. Figure 2 is showing the user's interaction set up.

4.2 Procedure and Study Design

Each evaluation session was limited to 5 min of playing and approximately one minute and half of training before starting the test. The training session consisted of a free user interaction while the evaluator read a script describing the function of each button as well as the game goals. The evaluations comprise both a subjective survey by means of

a questionnaire and an objective investigation by means of log data collected during the users' interactions. The subjective measures include perceived performance and usability issues. The four evaluation sessions per user took approximately 30 min.

All events caused by the user-control device interaction were written to a log file to enable an assessment of the user's behavior. The log file registered the following data: screen coordinates where the users touched for each button, error rate, score and quantity of lives spent in the game. The success rate is committed when the user performed a touch inside the area of the button. All subjects used both an adaptive and a non-adaptive version of each game genre to facilitate a comparison of the two conditions. The order in which the subjects tested the individual systems was altered to eliminate an effect of the order, for instance through a training effect. Likewise, the adaptive and the not-adaptive version were used first alternately. The same procedure was adopted for alternating the game genres order. Basically with this study design we intend to verify two main design issues: firstly how the adaptation is improving the user interaction, and secondly if the achieved adaptation pattern suffers influence from the game genre. In the next section we discuss our results.

5 Results

In this section we described our obtained results divided into two kinds of achievements: the objective results including the data log analysis and the subjective results involving the post questionnaire users' answers.

5.1 Objective Results

In order to verify how the adaptation to the user behavior is improving the user-control interaction we observed two variables: the success rate and the average of score per life. Observing Fig. 4 we can conclude the adaptive control reached better success rates than the not-adaptive version in both genres of games. For the platform game (Super Mario Bros) the adaptive control augments the precision in 6.67 %. For the arcade game (Streets of Rage) the adaptive control augments the precision in 13.7 %. In Fig. 5 we can observe the adaptive control reached better score per life than the not-adaptive version in both genres of games.

Observing the graphs of Figs. 4 and 5, we evaluated the difference between both prototypes. Since our samples could not be assumed to be normally distributed we use the Wilcoxon Signed-Rank non-parametric statistical two-tailed hypothesis test with a threshold of 0,05 for significance. Table 1 presents the results for each variable in game and the results are statistically significant.

Figure 3 (a) and (b) shows different layouts corresponding to the intersection of all users achievements at the end of the interaction for each game genre. It is possible to observe that depending on the genre of the game, the final layout may have some similarities. For instance, the platform games, which the main objective of the game is to reach the rightmost part and avoid to be hit by enemy either passing or jumping, may present the Right and the Jump buttons bigger than others. However, for arcade games,

where the player must walk through scenario and hit enemies, the configuration of the controller changes slightly, where the punch button probably corresponds to the largest one, the directional buttons may vary and the Right button tends to be the largest button of the directional.

Table 1. Results of wilcoxon test

Game	Variable	P-value
Super Mario Bros.	Success Rate	0.003357
Super Mario Bros.	Score/Life	0.0155
Streets of Rage	Success Rate	3.052e-05
Streets of Rage	Score/Life	0.01443

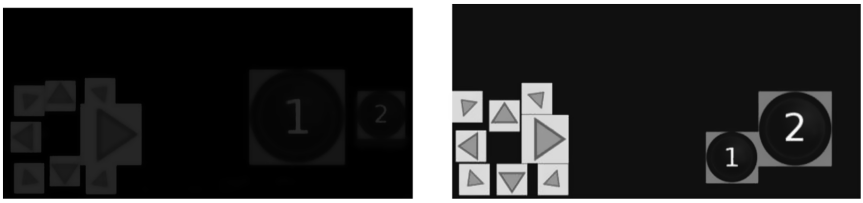

Fig. 3. Intersection of all users achievements at the end of the interaction for (a) Streets of Rage and (b) Super Mario Bros.

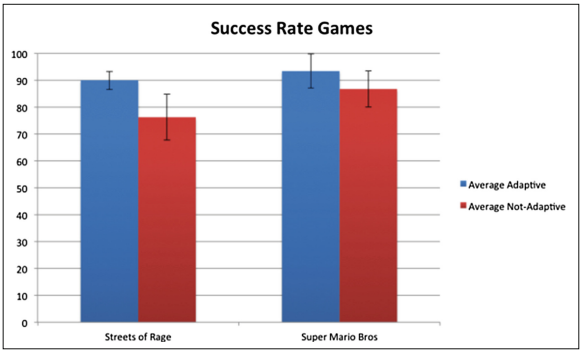

Fig. 4. Touch buttons precision measured as Success Rate

5.2 Subjective Results

After the tests, users answered a questionnaire about the usability of the adaptive control, with questions about which controller the user believes he played better, which prototype he would prefer to use to play again, the general ease of use for both controllers and if he is interested in using an adaptive controller. All users receives two identical questionnaires, one for each game played.

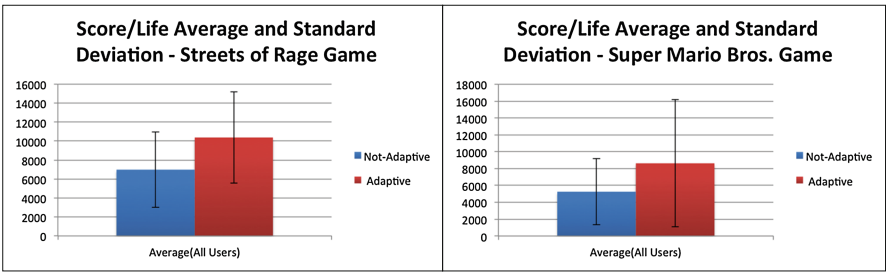

Fig. 5. Performance measured as Score/Life, where the y-axis displays the average score reached in the game for each life (a)Streets of Rage (b) Super Mario Bros.

The statistics were favorable to use this type of adaptive control. We observed that 88 % of the users prefer to play Streets of Rage using a control with adaptation and 94 % of these users prefer to use this adaptive control to play Super Mario Bros. Regarding the ease of use, the median score was 4 (where 1 corresponds to very hard and 5 to very easy, with the upper quartile equals to 5 chosen by four users and one unique minimum value equals to 2. The other question asks if the user would use the adaptive control in other possibility. The scale is again from 1 to 5, which 1 represents I would not like to use and 5 represents I would love to use again. The median of this question was 4, with the upper quartile equals to 5 and the minimum value equals to 3. From the subjective evaluations, we can conclude that the adaptive prototype was well accepted for most users. According to this subjective evaluation, the users appeared motivated to accept and change to this new way to play, even with the large experience from users over the conventional joystick.

6 Conclusion and Future Work

New forms of user's input are being proposed by the game industry in order to attract more players and enhance the immersion during the gameplay. Many users avoid playing games due to the complexity observed in input devices, pushing them away from the video games community. Using a mobile phone as a game data input gives the opportunity for new users that are resistant to these complex input devices to try and possibly enjoy playing games using a device that they already have. With that, we hope to minimize the time to learn necessary to start enjoying a game.

The evaluations not only investigated whether the proposed adaptation improves the user-control interaction, but also give insight into the general conditions under which the adaptations perform well. We can conclude that the adaptive controller brings benefits to great part of the users and from this brief evaluation we can start pointing an ideal interface. The advantages showed by the test results indicate that the participants achieved a configuration for the controller that allowed a very low standard deviation, principally playing the game Streets of Rage. Another observation comparing the user's final interface was the disposal of the buttons. Some users changed the original positions of the buttons 1 and 2 (originally placed with the button 2 slightly

above the button 1) until both buttons were aligned in a position more similar the one found on controllers for video game consoles.

Another interesting finding was related to the user attention focus while playing. It was observable through video capture that in the non-adaptive version the users frequently alternate their attention focus between the control and the computer screen while in the adaptive version this frequency decreased considerably. However in the next work we hope to proof this finding by capturing more valuable and objective data of the user's attention focus for instance by using an eye tracking system.

The machine learning algorithm is another aspect where improvements can be made. The K-means implementation in use considers that the number of classes is equal to the number of buttons in the controller. However, when a game doesn't use all buttons this kind of assumption will lead to a higher than needed number of classes and centroids, affecting the adaptation process. In this direction a more flexible implementation of the algorithm that considers only the buttons in use will be investigated.

Regarding the user's experience evaluation other parameters involved in the adaptation process (such as speed adaptation, maximum change in a button position and so on) could be further investigated to fine tune the controller adaptive behavior which could lead to an optimal configuration to the controller.

Acknowledgments. We thank all participants who have contributed in our experimental studies and all colleagues at Universidade Federal Fluminense for their valuable comments to improve this work.

References

1. Koster, R.: Theory of fun for game design. O'Reilly Media Inc., Sebastopol (2013)
2. Schell, J.: The Art of Game Design: A book of lenses. CRC Press, Boca Raton (2008)
3. Joselli, M., Junior, J.R.S., Zamith, M., Clua, E., Soluri, E.: A content adaptation architecture for games. In: SBGames. SBC (2012)
4. Langley, P.: Machine learning for adaptive user interfaces. In: Brewka, G., Habel, C., Nebel, B. (eds.) KI 1997. LNCS, vol. 1303. Springer, Heidelberg (1997)
5. Rogers, S., Williamson, J., Stewart, C., Murray-Smith, R.: Fingercloud: uncertainty and autonomy handover in capacitive sensing. In: ACM CHI 2010, pp. 577–580 (2010)
6. Weir, D., Rogers, S., Murray-Smith, R., Löchtefeld, M.: A user-specific machine learning approach for improving touch accuracy on mobile devices. In: ACM UIST 2012, pp. 465–476 (2012)
7. Bi, X., Zhai, S.: Bayesian Touch – a statistical criterion of target selection with finger touch. In: ACM UIST 2013, pp. 51–60 (2013)
8. Baldwin, T., Chai, J.: Towards online adaptation and personalization of key-target resizing for mobile devices. In: IUI 2012, pp. 11–20 (2012)
9. Bezold, M., Minker, W.: Adaptive multimodal interactive systems. Springer, Boston (2011)
10. Dretske: Explaining Behavior. Reasons in a World of Causes. MIT Press, Cambridge, MA, USA (1988)
11. Smola, A., Vishwanathan, S.: Introduction to Machine Learning. Cambridge University, Cambridge (2008)

12. Malfatti, S.M., dos Santos, F.F., dos Santos, S.R.: Using mobile phones to control desktop multiplayer games. In: Proceedings of the 2010 VIII Brazilian Symposium on Games and Digital Entertainment, ser. SBGAMES 2010, pp. 74–82. IEEE Computer Society, Washington, DC, USA (2010)
13. Stenger, B., Woodley, T., Cipolla, R.: A vision-based remote control. In: Cipolla, R., Battitato, S., Farinella, G.M. (eds.) ICISTM 2012. SCI, vol. 285, pp. 233–262. Springer, Heidelberg (2010)
14. Vajk, T., Coulton, P., Bamford, W., Edwards, R.: Using a mobile phone as a wii-like controller for playing games on a large public display. Int. J. Comput. Games Technol. **2008**, 4:1–4:6 (2008). http://dx.doi.org/10.1155/2008/539078
15. Wei, C., Marsden, G., Gain, J.: Novel interface for first person shooting games on pdas. In: OZCHI 2008: Proceedings of the 20th Australasian Conference on Computer-Human Interaction, OZCHI, pp. 113–121. ACM, New York, NY, USA (2008)
16. Zyda, M., Thkral, D., Jakatdar, S., Engelsma, J., Ferrans, J., Hans, M., Shi, L., Kitson, F., Vasudevan, V.: Educating the next generation of mobile game developers. IEEE Comput. Graph. Appl. **27**(2), 96, 92–95 (2007)