# A Mobile Game Controller Adapted to the Gameplay and User's Behavior Using Machine Learning

Leonardo Torok[1], Mateus Pelegrino[1], Daniela G. Trevisan[1], Esteban Clua[1], and Anselmo Montenegro[1]

[1] Federal Fluminense University, Computing Institute
Rua Passos da Pátria 156 – E – 3rd floor, São Domingos, Niterói, Brazil
mateuspelegrino@gmail.com,
{ltorok,daniela,esteban,anselmo}@ic.uff.br

**Abstract.** When playing games, the user expects an easy and intuitive interaction. While current controllers are physical hardware components with a default configuration of buttons, different games use different buttons and demand different interaction methods. Besides, the player style varies according to personal characteristics or past gaming experiences. In previous works we proposed a novel virtual controller based on a common touchscreen device, such as smartphone or tablet, that is used as a gamepad to control a game on a computer or game console. In this work we include machine-learning techniques for an intelligent adaption of the layout and control elements distribution, minimizing errors and providing an enjoyable experience for individual users. We also present different usability tests and show considerable improvements in the precision and game performance of the user. We expect to open a new way of designing console and desktop games, allowing game designers to project individual controllers for each game.

**Keywords:** Touch surfaces and touch interaction, input and interaction technologies, machine learning and data mining, games and play.

## 1 Introduction

When playing a game, one of the main features that will define the perception that the user will have about its experience is the quality and fluency of the game controls, responsible for commanding the player's in-game avatar. Therefore, in the whole area of games, one factor that is very important for gameplay experience is the controller, or the control scheme. The most memorable gaming experiences created were generally based on good controls schemes. They are usually intuitive, allowing a new player to immediately start interacting without difficulties, and provide a deep interaction, with a vast array of possible actions in the easiest way. Similarly, an inadequate controller design is one of the first characteristics of games that are frequently remembered as unpleasant experiences with very few redeeming qualities. This direct relationship between controls and the overall quality of a game is easy to


K. Chorianopoulos et al. (Eds.): ICEC 2015, LNCS 9353, pp. 3–16, 2015.
DOI: 10.1007/978-3-319-24589-8_1

understand. The controls are the bridge between what the user want to do and what his avatar will actually do in the game. In the past, most games were simple and gaming controllers actually employed just a few buttons. Nowadays, controllers are presented with dozens of buttons, directional pads, analog sticks and even motion sensing and touch capabilities. While these interface devices may give a huge amount of interaction possibilities, they impose many constraints for the game designer, since he must use the same hardware for any kind of game, and increase the learning curve for players. A regular gamepad has a fixed size and shape, with the same buttons in the same position and with the same size, independently of who is using it or his ergonomic preferences. While this constraints provide a standardized interface that user may be more comfortable with and provide a simpler interface for the game designer, we would like to create an alternative for designers that are willing to explore different and fully customizable interfaces to extend their concepts from the game to the controller.

On the other hand smartphones and tablets opened a new paradigm of game interface, giving more freedom to game designers when projecting an interface. The new and simpler paradigm of control imposed by these mobile devices was responsible for bringing millions of casual players to the gaming world. The virtual controls allow mobile games to abandon older paradigms and design interfaces without regular buttons, creating a novel control scheme. Users can drag objects on the screen, perform gestures (such as pinch, rotate, etc) and still use regular virtual buttons. This flexibility allows game designer to design not only the game's visuals and gameplay element, but also the control interface, with any shape and interaction paradigm they wish and possessing only the buttons that are necessary for the game. While this already happens in most mobile games, their gameplay is usually simplified and the interfaces are created with different paradigms. The proposed approach tries to blend the vast capacity of input options and general game complexity of the more traditional console experience with the flexibility provided by touchscreen interfaces.

With our previous proposal, each game can have a custom interface, with the correct amount of buttons for its specific needs or even replacing buttons altogether with different interface elements, like regions in the screen where the user can input commands with specific multitouch gestures. The controller now becomes a part of the game design, fitting into the experience that the designer intends. The adaptive controller resulted in a patent application [17] and it is important to notice that it was created to fill a practical purpose and is intended to be released as a solution to end users.

However, this flexibility brings some challenges: touchscreens cannot replicate the same precision of a traditional controller and the control interface projected by the game designer is not necessarily the best possible interface from an ergonomic standpoint and may need some tweaking to reach the optimal configurations. To solve this problems, our solution introduces an adaptation that derives the personal preferences of the user from a series of basic events, such as button presses, or internal state changes, adapting itself to the user and his ergonomic needs based on machine-learning approaches. To perform this task, different algorithms mine

information from game events, smoothly adapting the interface using machine-learning methods. The controller will try to smoothly fix the position and size of the buttons in order to eliminate or reduce errors, with the potential to improve the accuracy of a touch screen interface and correct a suboptimal default configuration, according to the users' needs and characteristics, such as hand size and mobility. In the end of this paper we show how our proposal increased the game experience for popular and commercial games, comparing a basic controller interface with and without our proposed adaptations.

## 2 Related Work

An adaptive user interface is an interactive software system that improves its ability to interact with a user based on its partial experience [15]. Rogers et al [19] 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 [24] used a 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 [4] conceptualized finger touch input as an uncertain process, and used statistical target selection criterion. They improved the touch accuracy using a Bayesian Touch Criterion and decreased 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 [2], this type of adaptability has some disadvantages. One that immediately comes to mind is the limitation of only working in of language processing, adapting the interface according to the language's model and the user's typing behavior. Our work is not just destined to fit the interface more comfortably to the users according to their type of usage, but also aims to avoid errors. In order to achieve that, we use the data about correct and incorrect virtual button presses to change different properties of the button.

Touchscreen devices were already used as gaming controllers, with commercial solutions already available, such as GestureWorks Gameplay [9] that allows the user to play using an Android smartphone as a controller and to customize the layout of his joystick. However, none of these products present a solution to the limited precision in touchscreen input and are not capable of determining the best ergonomic configuration for the user, which has to design the controller manually, resulting in a sub-optimal configuration. In a previous work [26], we developed a framework for building customizable controller based on a mobile device and in [17] we presented a mobile controller that is created and designed by the game developer specifically to his game, but does not include any kind of adaptation to improve the designed interface. This novel interface resulted in a patent filled at INPI [18], that includes the concept of a gaming controller that can be customized for each game and includes machine learning to improve its usability.

In [26] we made a first attempt for simple adjustments, moving buttons in accordance with heat maps generated by the touches. The biggest limitation in our previous attempts is that no intelligence was being used for the movements, so that a touch in the inner side of a quadrant will have the same weight of a touch in the outer area of the quadrant, creating a biased result. In the present work we propose a much more sophisticated and novel approach, based on a K-means classifier, treating all points without any bias issues. While the usability results in [26] were based on subjective opinions provided by the users, we now developed an approach capable of automatically monitoring the user behavior, collecting all input data during user interaction and creating a log file with each user's error rate when trying to touch a button and several other statistical output. With these data, it was possible to perform several statistical analyses and determine in a formal way if the controller's adaptations are increasing the player's performance and the gameplay experience. To determine the optimal configuration for the learning algorithm, this work included pilot tests that allowed us to perform several improvements and fine-tune the controller to achieve an even better adaptation to the user's needs.

## 3 Proposed Adaptive Interface

The adaptive controller is composed of a physical component (a smartphone or tablet) and a software component (client and a server). The mobile device will run the client software, presenting the user interface, collecting input data, performing the machine learning routines and adaptations and sending all inputs to the computer that runs the game and the server component, receiving the inputs and converting them to events that perform in-game actions. The mobile app is created with the Android SDK [1] in the Java language, using the Weka library [25] to perform the machine learning routines, with all input data being stored in an internal SQLite database. The desktop application is also programmed in Java and receives the commands from the controller..

To achieve the desired adaptation to each user's play style it is necessary to perform several adaptations. For this work, it was 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 current game, increasing the size of the most used buttons in the controller and decreasing the size of the buttons that are less used.

The second type of adaptation is the button's position. The basic concept here is to try to detect the position for a specific button that will guarantee that the majority of users' touches actually hit the button. A machine-learning algorithm will determine the position that maximizes the user's precision. In our implementation we included size and position redefinitions, but many other properties can be changed in the future, such as color, shape and force feedback.

To keep the interface consistent, several rules and boundaries were specified: The maximum size of each button is defined and conflicting areas are analyzed, verifying

if the areas of any button intersects with a neighbor. The data used to evaluate the controller's correctness is a database with all the touches performed by the user. The screen is mapped in Cartesian coordinates and each touch in the screen is stored in a database. The machine-learning algorithm will use the most recent stored data (detailed in the section 4). Each touch will be classified as correct or incorrect. Correct touches are those that hit any button in the screen and are mapped to an action, while incorrect touches did not hit any button, representing a situation where the user tried to perform an action and failed. We defined that in our heuristic the size will increase for the most used buttons and decrease for the least used ones. A simple algorithm that tracks and counts the amount of times each button is used was included. A list of buttons is created, ordered by the amount of touches on each one. This list is divided in 3 parts, with the first one containing the most used buttons and the last one containing the least used ones. The controller will increase the size of the most used buttons while decreasing the least used ones to their original size. This list is update once per iteration, always following the current player's needs.

The position adaptation heuristics demands a more sophisticated process, that tries to find the points in the screen that represent the centers of the most used areas. A clustering algorithm presents a good solution for this case, classifying a set of points, or to touches, in classes that represent the buttons. Due to the characteristics of this problem, we decided by using K-means.

The input data, the most recent user touches, is passed to a K-means unsupervised learning algorithm, which is a vector quantization method for data mining [20]. This classifier receives a set of points and separates them in K classes of related entries. In our case, the algorithm is initialized with random points, a common approach for K-means. Although is a NP-hard problem [11], several heuristics allow a quicker solution that converges surprisingly fast to a local optimum. Given $_X = \{x_1, \ldots, x_m\}$ the goal of the classifier is to partition X into k clusters with similar values [20], defining prototype vectors $\mu_1, \ldots, \mu_k$ and an indicator vector $r_{ij}$ which is equal to 1 if, and only if, $x_i$ is assigned to the cluster j. The distortion measure and the distance of each point from the prototype vector will be minimized:

$$J(\mathrm{r},\mu)=\frac{1}{2}\sum_{\mathrm{i}=1}^{m}\sum_{\mathrm{j}=1}^{k} r_{\mathrm{ij}}\left\|x_i-\mu_j\right\|^2, \tag{1}$$

where r = { $r_{ij}$ } and $\mu = \mu_j$. In order to find r and μ a two stage strategy is used. First, μ is fixed and the objective is to determine r. The solution for the i-th data point $x_i$ can be found by setting:

$$r_{ij}=1 \; if \; j=argmin\|x_i-\mu_{j'}\|^2, \tag{2}$$

and 0 otherwise. In this case, r is fixed and μ will be determined. J will be a quadratic function of μ that can be minimized by setting the derivative of $\mu_j$ to 0:

$$\sum_{i=1}^{m} r_{ij}(x_i-\mu_j)=0 \; for all \; j \quad \text{, rearranged as} \quad \mu_j=\frac{\sum_i r_{ij}x_i}{\sum_i r_{ij}}. \tag{3}$$

Since $\sum_i r_{ij}$ counts the number of points assigned to cluster j, the algorithm sets $\mu_j$ to be the sample mean of the points assigned to cluster j. The algorithm runs several iterations until it stops when the cluster assignments do not change significantly anymore. Originally, K would be the number of virtual buttons, but the algorithm was adjusted to use a more reliable metric, defining K as the number of buttons that were used in the current gaming section. This approach avoids the creation of redundant classes and unnecessary centroids that represent the same button.

The K-means algorithm will return several subsets of the input dataset grouping the closest points in the same class and returning the respective centroid for each class. The users' touches will be located in the area of a button or at least close to it, since the use is trying to hit the buttons. After several interactions, we will observe a pattern of touches close to each button, allowing the K-means clustering to separate inputs in classes that represent each buttons area and part of the surrounded space. After finding the centroids, each one will be paired with the closest button. In cases where there are two centroids and the closest button is the same, the correct pair will be created based on the minimal distance, leaving the other centroid without assigned buttons. The centroid paired with the button will be considered as the its optimal position, representing the mean 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 corresponding centroid. These changes on the buttons are visual and the user can observe the adaptations being performed in real time.

## 4 Usability Tests and Results

In order to validate how our proposed adaptation behaves with the final user and his gameplay experience, we conducted a usability test, observing the effectiveness and user satisfaction regarding our adaptations. To perform this evaluation, we divided it in two stages: the pilot tests and the final user tests. The pilot test was realized to set and define important adaptation parameters to the final test, determining an optimal configuration to the adaptation.

The evaluation used two different controllers, an adaptive and a non-adaptive, both with the same functionality and layout. It was not told to the users that our controller is adaptive and were just informed that they would test two different joystick prototypes with two game genres: platform with Super Mario Bros. (Nintendo) and the 2D shooter Sonic Wings (Hamster Corporation), totalizing four evaluation sessions per user.

**Fig. 1.** The setup used for testing.

Our comparison sections were focused in comparing the adaptive controller with a regular version, that represents a game controller created by a game designer with a layout that is intended to provide the best experience for these specific games. However, it is hard to determine if an interface is optimal, requiring several usability tests. This approach is costly and demands a precious time that will not be available for any game developer. With these usability tests, we expect to show how an adaptive controller can improve a customized game controller and optimize it to a more ergonomic configuration. Comparisons with current physical controllers were discarded, since the objective of this work is to provide a fully customizable interface that is also capable of adapting itself to improve its usability. As such, we are trying to provide a functionality that does not exist on traditional controllers and our tests are focused in demonstrating the benefits that these adaptations can bring to an initial unoptimized interface created by a game designer.

Our group of volunteers consisted of 16 users, 9 male and 7 female, with ages ranging from 18 to more than 60 years old. The user group was selected and separated in two different groups: 8 expert users and 8 novice users. The first group was composed basically by users that play games on consoles or PC regularly, are used to play with a joystick and play videogames for more than 10 years. The second group was made by users with less experience with games and mostly consumers of casual and simple games. An Asus Nexus 7 tablet (2012 version) running the Android OS 4.4.4 was used in all usability tests. The capacitive touchscreen has a diagonal of 7.0 inches with an aspect ratio of 16:9 and a resolution of 1280 x 800 pixels. Figure 1 shows the interaction setup for our tests.

The adaptive controller must be configured with two parameters that will interfere in the final adaptation process: the movement limit of the button per iteration, measured in pixels, and the amount of the most recent points sent to the K-means algorithm. For the first one, a higher value will make the controller change its layout faster. In the second parameter, a lower value will make the controller consider only recent points, adapting faster to the current conditions of the interaction. In the pilot tests, three different configurations were tested with several games from different genres: conservative, intermediary and aggressive, presented on this order on Table 1.

The results showed some interesting conclusions. The faster adaptation allowed the controller to answer quickly to changes in the user play style, such as situations where the user changed the position of his hands without noticing, or changes in the own game, like bonus stages with different level designs. In a subjective evaluation, the faster adaptation provides a controller that adapts better to the game and has a quick response to correct the user's mistakes. The empiric feeling that this adaptation was better was also confirmed by our success rates.

**Table 1.** Results of the pilot tests.

| Setup | Sonic (success rate) | Lifeforce (success rate) |
|---|---|---|
| 5 pixels, 100 points | 97.56 | 96.62 |
| 10 pixels, 30 points | 97.86 | 97.63 |
| 100 pixels, 10 points | 98.58 | 97.71 |

Once the adaptation parameters were defined we started the usability tests with the users. Each evaluation session was limited to 5 minutes of gameplay and approximately one minute of training before starting the test. The training session consisted of a free user interaction while the evaluator read the test script describing the function of each button as well as the game goals. The evaluation comprises both a subjective survey collected by a questionnaire and an objective investigation registered by the collected log data during the user's interactions. The subjective evaluation includes perceived performance and usability issues.

All events caused by the user-control device interaction were written to a log file, registering the touches, the controller's success rate for each period of time, the success rate of each button, the final success rate, score and quantity of lives spent in the game. The success rate is a value between 0 and 100, which represents the percentage of correct touches. The success rate per buttons is calculated in a similar way, but only taking in account the touches destined to that specific button. Each correct touch is associated to the button the user pressed and each incorrect touch is associated with the closest button.

All subjects used both controller versions with both games in an alternated order to eliminate any training effect caused by the evaluation order. Thus, each user had his unique sequence of test. To select the sequence, which the user would test, we performed a permutation between both controllers' version and game genres. The adaptive version will be referred as KA (K-means adaptive) while the non-adaptive version will be referred as NA.

After determining the most suitable design for our tests, we created hypotheses to validate the proposed interface:

H1: Adaptive controller will increase the user success rate for the novice users.

H2: Adaptive controller will increase the user success rate for the expert users.

H3: Adaptive controller will increase the user success rate independently of his experience.

H4: Adaptive controller will increase the user success rate independently of game genre.

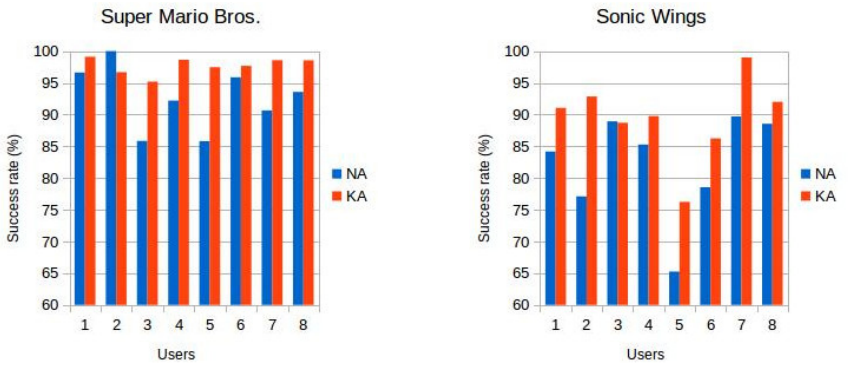


**Fig. 2.** Success rate for the novice users on Super Mario Bros (left) and Sonic Wings (right) using the K-means adaptive controller (KA) and the non-adaptive controller (NA).

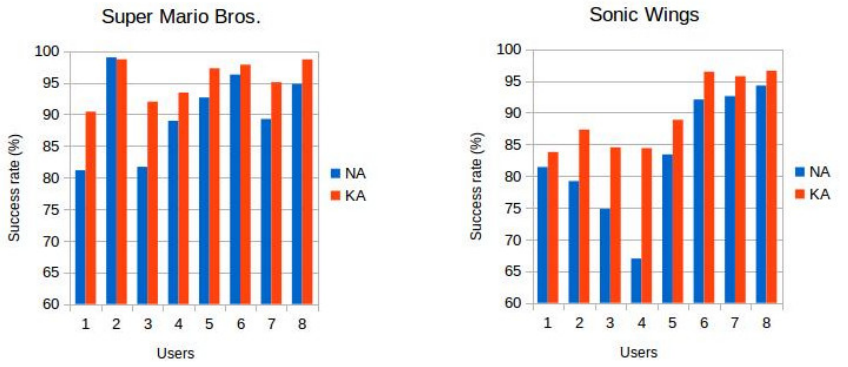


**Fig. 3.** Success rate for the expert users on Super Mario Bros (left) and Sonic Wings (right) using the K-means adaptive controller (KA) and the non-adaptive controller (NA).

To perform a comparison between both versions of the controller in our objective analysis, we will compare the mean final success rate of all users in one group with the same metric for the other group. After that, a Wilcoxon signed-rank test, a non-parametric statistical hypothesis test, is performed (significance level of 0.05 and the two-tailed hypothesis defined), returning the p-value, that indicates if the difference between the results achieved for both groups is significant. The decision of using a non-parametric test was motivated by the unknown distribution of the test results, since a parametric test requires a previous knowledge about the data distribution, that must be normal. For our groups' sizes, a p-value smaller than 0.05 represents a significant difference. Figure 2 and 3 presents the success rates for all users in both groups for all evaluation sessions. . Only two users do not present a better success rate with the adaptive version compared with the non-adaptive one. Figure 4 shows the initial and final configuration achieved by the adaptive controller for a user. Figures 5 presents the success rate over time for each user group, while figure 6 presents the success rate over time for both combined user groups.

**Table 2.** Results of the Wilcoxon test and mean success rate for both controllers for all groups.

| Group | Game | NA mean (%) | KA mean (%) | p-value |
|---|---|---|---|---|
| Novice | Super Mario | 92.89 | 98.11 | 0.03906 |
| | Sonic Wings | 84.73 | 90.42 | 0.01563 |
| Expert | Super Mario | 90.95 | 96.17 | 0.01563 |
| | Sonic Wings | 82.41 | 88.10 | 0.007813 |
| All | Super Mario | 92.43 | 97.57 | 0.0004272 |
| | Sonic Wings | 83.79 | 89.33 | 0.0001 |

**Fig. 4.** The initial and final configurations for an user.

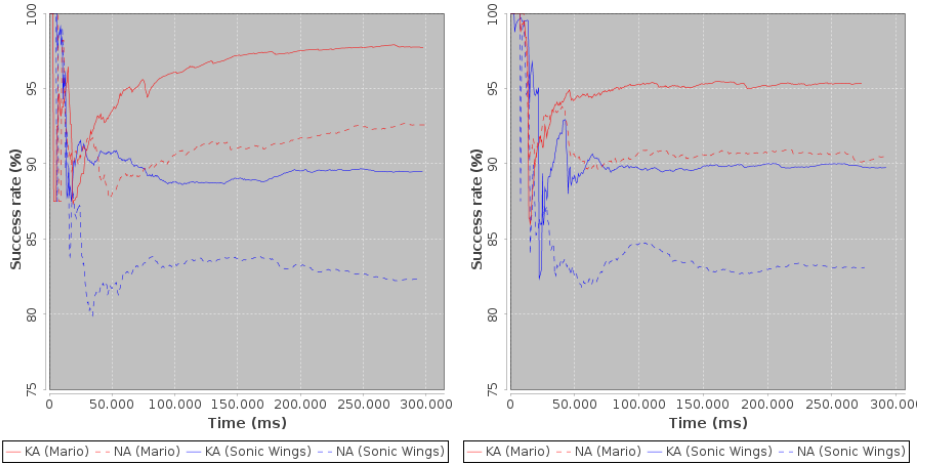


**Fig. 5.** Average success rate for all novice (left) and expert (right) users on Super Mario Bros and Sonic Wings for both controllers.

Table 2 shows the means for all users separated by group and for all users combined in a single group. We can see that for both games, the novice users had a higher success rate with the adaptive controller. The p-value was also less than 0.05, showing that the difference is significant and we can reject the null hypothesis. With these results, we can accept the hypothesis H1 that adaptive controller increases significantly the success rate for novice users. In a similar way, the expert users had a higher success rate in all cases. The p-value allows us to reject the null hypothesis and to consider the difference significant, validating the hypothesis H2.

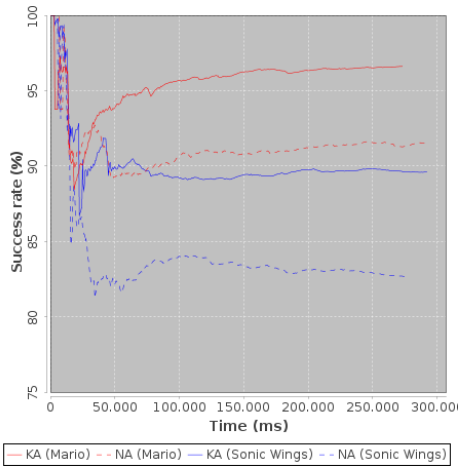


**Fig. 6.** Average success rate for all users on Super Mario Bros and Sonic Wings for both controllers.

However, during our evaluations of the log files, we noticed that the success rate for users of both groups were similar, leading us to believe that maybe our initial assumption that novice and expert users would use a game controller differently could be incorrect. In this case, all users would actually represent a single group and this would be an indication that the interaction with the gamepad could be more affected by other factor, such as ergonomic factors. Hence, we compared both groups to conclude if there is a significant difference between them. First we set the game Super Mario Bros. and confront the non-adaptive version for both groups. The p-value is 0.3828, so the result is not significant at $p <= 0.05$. With the same game, but confronting both adaptive results of groups the p-value is 0.1484, which is higher than the significance level, and then the difference is not significant. Setting the game to Sonic Wings and confronting the results of the non-adaptive version for both groups, we obtained a p-value of 0.8438, that shows that the difference is not significant. Lastly, with the adaptive in this game, the p-value of 1 and we concluded that this result is also not significant.

Surprisingly, the difference between both groups is not significant, showing that the level of experience probably is not an important factor in the usability of a game controller. With this result, we can combine all users in a single group. Now, it is necessary to test if, for this larger group of users, the adaptive joystick still is better than the non-adaptive version. Table 2 indicates the means and p-values for the unified group containing all users. The adaptive controller success rate was higher than the non-adaptive version, as indicated by the means. The p-values, smaller than 0.05, indicates that the difference is significant and we can reject the null hypothesis. With these last tests, we can conclude that the hypothesis H3 is valid and the adaptive controller increases significantly the success rate independently of the users' experience. Finally, we tested the hypothesis H4 comparing the adaptive version in Super Mario Bros with the adaptive version on Sonic Wings, including all sixteen

users. The obtained p-value was 0.0001526. This result is significant and we can conclude that H4 is rejected and the adaptation can achieve different results in different game genres. This indicates that the controller can adapt itself better in some games and is influenced by the genre.

The second part of the analysis corresponds to the subjective results. In our survey most users believed they played Super Mario Bros. and Sonic Wings better with KA (respectively, 69% and 81% of the users) and would use the KA controller in a second play session (88% of the users in both cases). The users also rated the ease of use for both controllers, with the results in table 3 showing a clear advantage for the KA controller.

**Table 3.** User's opinion about the ease of use of the controllers according to a Likert scale, where 1 is "Very hard to use" and 5 is "Very easy to use".

| | Super Mario Bros. | | Sonic Wings | |
|---|---|---|---|---|
| Difficulty | NA | KA | NA | KA |
| Very hard | 6% | 0% | 6% | 0% |
| Hard | 44% | 19% | 38% | 0% |
| Normal | 31% | 25% | 38% | 31% |
| Easy | 19% | 38% | 13% | 50% |
| Very easy | 0% | 19% | 6% | 19% |

## 5 Conclusions and Future Works

New forms of video game controllers are being proposed by the game industry to attract more players and enhance the immersion during 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 device as a game controller gives the opportunity for designing specific interfaces for each game, making the interaction more accurate and adapted for individual gameplay mechanic.

Our work not only investigated whether the proposed adaptation improves the user-control interaction, but also gave insights 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.

A great surprise was to discover that the adaptation helps experts, casuals and even non-players. More than that, the interface showed similar results for both groups. In both game genres, our version of the controller showed an average improvement of more than 5%. Although in terms of percentage this is not a huge difference, when realizing that a user touches more than 2500 times when playing Sonic Wings and more than 500 when playing Super Mario Bros, we can consider that we had a great improvement in accuracy. The similar results for both groups also creates the possibility of evaluating other factors, like hand size and mobility. We would also want to evaluate the relation between how many times a button is used and its real importance for the game, determining if reducing the least used buttons really is the best approach for all game genres.

Our system is based on games with no source code, which means that we don't have access to the gameplay implementation. In future works we intend to include our system to commercial game engines, so that the control can change in accordance to the game context. The authors also pointed the study of other ways of adaptability, changing not only the button size and position, but also the shape. We also intend to investigate how this kind of user-control adaptation could improve the game interaction of different user groups, such as older users or despaired children, which usually have some motor skills limitations. The machine-learning algorithm is another topic for further studies. Other clustering algorithms can be evaluated, like K-medoids and K-medians.

## References


1. Android SDK and Eclipse plugin. http://developer.android.com/sdk/index.html?hl=sk
2. Baldwin, T., Chai, J.: Towards online adaptation and personalization of key-target resizing for mobile devices. In: IUI 2012, pp. 11–20 (2012)
3. Bezold, M., Minker, W.: Adaptive multimodal interactive systems. Springer, 5 (2011)
4. Bi, X., Zhai, S.: Bayesian Touch – A Statistical Criterion of Target Selection with Finger Touch. ACM UIST 2013, 51–60 (2013)
5. BlueCove. https://code.google.com/p/bluecove/
6. Brandao, A., Trevisan, D., Brandao, L., Moreira, B., Nascimento, G., Vasconcelos, C., Clua, E., Mourao, P.: Semiotic inspection of a game for children with down syndrome. In: 2010 Brazilian Symposium Games and Digital Entertainment (SBGAMES), pp. 199–210 (November 2010)
7. Burke, J.W., McNeill, M., Charles, D., Morrow, P., Crosbie, J., McDonough, S.: Serious games for upper limb rehabilitation following stroke. In: Proceedings of the 2009 Conference in Games and Virtual Worlds for Serious Applications, VS-GAMES 2009, pp. 103–110. IEEE Computer Society, Washington, DC (2009)
8. Dretske.: Explaining Behavior. Reasons in a World of Causes. MIT Press, Cambridge (1988)
9. GestureWorks Gameplay on Steam. http://store.steampowered.com/app/296610
10. 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., Burdea, G.C.: In-home virtual reality videogame telerehabilitation in adolescents with hemiplegic cerebral palsy. Archives of Physical Medicine and Rehabilitation, e1 91(1), 1–8 (2010)
11. Gonzalez, T.F.: On the computational complexity of clustering and related problems. In: Drenick, R.F., Kozin, F. (eds.) System Modeling and Optimization. Lecture Notes in Control and Information Sciences, vol. 38, pp. 174–182. Springer, Heidelberg (1982)
12. Joselli, M., Junior, J.R.S., Zamith, M., Clua, E., Soluri, E.: A content adaptation architecture for games. In: SBGames, SBC (2012)
13. Joselli, M., Silva Junior, J.R., Zamith, M., Soluri, E., Mendonca, E., Pelegrino, M., Clua, E.W.G.: An architecture for game interaction using mobile. In: 2012 IEEE International Games Innovation Conference (IGIC), pp. 73–77 (August 2012)

14. Laikari, A.: Exergaming - gaming for health: A bridge between real world and virtual communities. In: IEEE 13th International Symposium on Consumer Electronics, ISCE 2009, pp. 665–668 (May 2009)
15. Langley, P.: Machine learning for adaptive user interfaces. In: Brewka, G., Habel, C., Nebel, B. (eds.) KI 1997. LNCS, vol. 1303, pp. 53–62. Springer, Heidelberg (1997)
16. 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, SBGAMES 2010, pp. 74–82. IEEE Computer Society, Washington, DC (2010)
17. Pelegrino, M., Torok, L., Trevisan, D., Clua, E.: Creating and Designing Customized and Dynamic Game Interfaces Using Smartphones and Touchscreen. In: 2014 Brazilian Symposium on Computer Games and Digital Entertainment (SBGAMES), pp. 133–139. IEEE (2014)
18. Pelegrino, M., Zamith, M., Mendonça, E., Joselli, M., Torok, L., Clua, E.: Controle orgânico virtual para jogos por dispositivos eletrônicos com tela sensível ao toque configurável e adaptável ao uso. Patent INPI BR 1020130300039 (filled in November 2013). http://www.inpi.gov.br/portal/artigo/busca_patentes (Accessed: February 2013)
19. 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)
20. Smola, A., Vishwanathan, S.: Introduction to Machine Learning, pp. 32–34. Cambridge University (2008)
21. Stenger, B., Woodley, T., Cipolla, R.: A vision-based remote control. In: Computer Vision: Detection, Recognition and Reconstruction, pp. 233–262 (2010)
22. 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., 4:1–4:6 (January 2008)
23. 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 (2008)
24. 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)
25. Weka: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka/
26. Zamith, M., Joselli, M., Siva Junior, J., Pelegrino, M., Mendonça, E., Clua, E.: AdaptControl: An adaptive mobile touch control for games. In: SBGames, pp. 137–145 (2013)
27. 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 Computer Graphics and Applications 27(2), 96, 92–95 (2007)