Auto Gamepad: A dynamic multiplayer game touch control based on user behavior

Dênes Vargas Teixeira, Gabriel Tobias Fuhr, Cesar Tadeu Pozzer
Laboratório de Computação Aplicada (LaCA)
Universidade Federal de Santa Maria (UFSM)
Santa Maria, Brazil
{dvteixeira, gtfuhr, pozzer}@inf.ufsm.br

***Abstract*—Recently, smartphones have emerged as an alternative easily accessible virtual control for computer games, especially in multiplayer gaming. However, controls with touch interfaces can cause a poor or uncomfortable playing experience and can be even more impactful depending on the type of player. This work proposes a new virtual control for computer games, running on a smartphone, that automatically adjusts its interface, identifying the needs of its current user from data collected in the application, aiming for better comfort and user experience. It uses machine learning upon this user data and operates without the need for additional game data in real time over the network. To validate the usability of our proposed solution, we use a multiplatform multiplayer action game.**

***Keywords*—dynamic control; mobile control; Gamepad;**

I. Introduction

According to [1], the use of smartphones for gaming is in constant growth, much of this can be attributed to the advancement of mobile technology and its immense number of users. Recently, some proposals have suggested to use this mobile scenario to integrate computer games, merging its huge gaming market base, with the flexibility of mobile technology to be used as touch controls.

But the fact that touch screens do not offer physical buttons that can provide tactile feedback like a traditional video game controls or a keyboard can be a limiting factor for the smartphone experience as virtual controls. To mitigate this problem, the aim of this work is to present a way to make the mobile gaming experience more responsive to the player's actions, implementing solutions that seek to adapt the virtual buttons of the touch screen to the player's game style.

Differently from previous works, our proposed architecture explores different aspects of existing solutions by developing a multiplayer gameplay scenario and an innovative mobile control adaptive to user behavior. The test platform is designed to focus on virtual control for smartphones, which we call Auto Gamepad, and is designed to allow multiple players to connect their phones as virtual controls and test the deployed solutions, making each control suit the players in each match, thus increasing the number of tests that can be performed simultaneously.

This paper is organized as follows: Section II presents the previous works that proposed similar ideas and solutions. Section III describes the current scenario of desktop games with approaches to virtual controls. Section IV presents the architecture and technologies used in the project. Conclusions and future works are presented in section V.

II. Related Works

In this section we present works that focus on the use of smartphones as virtual controls in computer games and their proposed architectures, which are directly related to this present proposal.

The proposal of using virtual controls in smartphones was firstly presented by [2] with a framework based on Bluetooth communication, using a single player game and a computer as it gaming platform, capable of capturing different input data compared to other conventional devices. Similarly, the work of [3], for Symbian devices, allowed a multiplayer environment using Bluetooth communication.

Later, [4] advanced the architecture of virtual controls, introducing an architecture that supports Bluetooth and TCP/IP communication. Although with a very simple interface, it was able to use algorithmic solutions to adapt smartphone virtual controls to its users, but performing all the processing of the new interface on the server and sending it over the network.

To further facilitate the use of virtual controls, the work developed with the Unity Game Engine [5] has contributed to improving the usability and user experience of virtual controls. In this work they presented an efficient solution to adapt and improve control without maintaining a predominant visual aspect during execution, based on the analysis of user data, using machine learning techniques and doing their processing on the mobile device.

In contrast, the virtual control proposed by this research works in conjunction with a local database that stores user data on the device and server is responsible for sending a new interface through XML sequences, which can affect scalability depending on the application.

The architecture proposed by [6] describes a communication prototype developed in Java, to use smartphones as interfaces, but does not consider adaptive interfaces in its validation environment with computer games, ignoring users interactions with its prototype.

III. Current scenario of desktop games with support to virtual controls

The following are examples of virtual controls that complement computer games and the console/joystick experience, adding not only new features to the game, but also performing normal control tasks.

A. PS4 Remote Play

A great example of the integration of a gaming platform and mobile control application is the PS4 Remote Play, which has the streaming functionality of the Playstation 4 game for the mobile screen via wireless internet. Some versions of the application also allow the use the cell phone as a control and display at the same time.

B. Battlefield 4 - Commander Mode for Android and IOS

This app use the mobile interface to allow a player to enter in a Battlefield Multiplayer not as a human player, but as a Commander who can interact with the map giving supply drops and support to the human players on the ground giving an advantage to their team.

C. AirConsole - Multiplayer Games for friends

AirConsole is a private offline gaming initiative made for the multiplayer browser gaming experience. The purpose of the platform is to give players the ability to play together with other players anywhere that has a computer connected to the Internet. One of its great achievements is that it has support for multiple smartphones acting as game controls in the same game. This feature makes the multiplayer offline experience more accessible to players who do not have up to four different physical controls.

In order to improve and study the technologies and techniques used in these examples, especially the latter, in this paper we present an example of computer game that has been developed to support up to 16 offline players. Our solution, a multiplayer game and Auto Gamepad, offers a complete multiplayer experience.

With this game of tests and the Auto Gamepad, we can offer a complete game experience to validate the applicability of our adaptive control.

IV. The Proposed Architecture

The test scenario consists of multiple users with Auto Gamepad who operate as clients in conjunction with the computer game that acts as the server (Fig. 1). Auto Gamepad has machine learning subroutines so that the interface adapts itself to the player that uses it in real time.

It is not necessary to send information from the server to the real-time controls to make adjustments to the interface. Differently from [5], our approach focuses on the performance as well as the simplicity of development, since the new button layout is generated upon data available on the smartphone.

In the action game used as a server, each player connected to the server controls a car inside a battle arena and each car is able to score points eliminating other players.

The Unity was used in the development of the application because it has libraries that help in the communication of the devices, besides being one of the engines most used currently for the development of games [7].

In the following subsections, we will go into details about the implementation and operational architecture of the control and the game. Features such as the overall game server

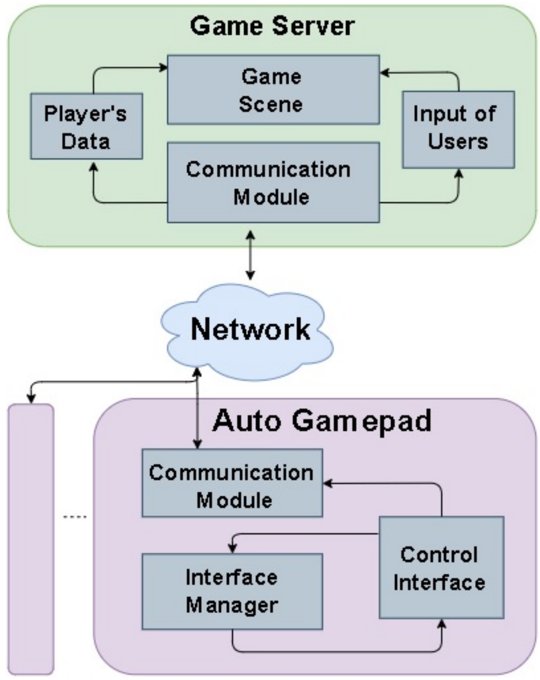

Figure 1. The Architecture modules. The vertical empty purple box indicates that the architecture supports multiples clients connected at same time, using the same scheme of Auto Gamepad.

architecture that will host the multiplayer test scenario and virtual control network technology were based on the Unity platform documentation and related work [3] [4] [6]. The next subsections presents a detailed architecture.

A. The Auto Gamepad

The Auto Gamepad interface was developed specifically for the test game, providing specific controls for the player's car movement. For other games, a new interface must be produced, but the adaptive structure of the adaptive buttons remains the same.

The interface has a joystick to drive the car and four action buttons to shoot, jump, increase speed for a limited time, and reset.

We defined a design that is in accordance with the proposal of the game and with a self-explanatory appearance in relation to its operation. A screenshot of the application can be seen in Figure 2.

Two important aspects were considered during the automatic adaptation of the control interface:

1. Exploit ease and efficiency of intelligent adaptation methods with the current user's control. As reviewed in related works, gains in the user's experience with the game are obtained, as well as better initial adaptation to the application and can improve the player's learning curve.

2. Keep the layout of the buttons as close as possible to the original layout.

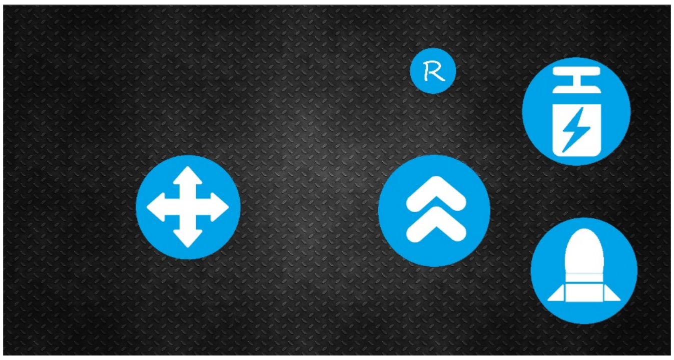

Figure 2. Control interface initial state.

The Auto Gamepad uses the k-means clustering algorithm [8] to adapt the real-time interface, as well as a subroutine that delimits its operation.

The k-means clustering receives a centroid for each button that must be adapted when the application starts, and each centroid is respective to the current position of each button.

A new interface layout is created after 10 clicks on the screen; after that process, the control self-adjusts smoothly and at intervals of 200 ms, until it reaches the position inferred by the k-means generated centroids.

The subroutine that delimits the positions of the centroids of the buttons has its fixed position when the application starts and the new positions generated by Interface Manager does not exceed the areas imposed by the delimiters. Fig. 3 shows a demonstration of how these delimiters acts internally in the control.

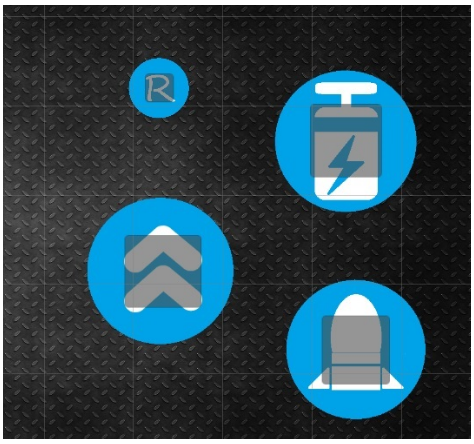

Figure 3. Buttons delimiters highlighted in gray.

In this way, the control can switch to another player, providing a facility for it to quickly adapt to its current interface and does not have the discomfort with a different interface that the game proposes.

This also causes that the proposed interface design is not affected and remains functionally stable, avoiding button overlaps, and reducing the learning curve for player to adapt themselves to use the interface between different sections of the same game.

B. Desktop Game Server

In our multiplayer action game, we try to put elements present in current games to make it fun and challenging at the same time, to allow us to analyze how the controller works in a more modern game.

The game attributes a different name and color to each character that is connecting to the game and has its IP saved by the server. Even when a player disconnects, the server have saved information about the character so that he can reconnect again without losing information on his character.

The objective of the game is to kill more players in a match, which may be by time or by limiting score. The game scenario is a battle arena and whenever a player dies, he respawn in a random place within the arena. Each car has the ability to fire missiles against other players, jump with the car to avoid missiles, activate a higher speed for a short time and also the ability to respawn.

Every player who eliminates another player with his missiles gains a point and the game displays in real time the ranking of players. In this way the game causes the user to use all the buttons to win and with high frequency depending on the player skills. A screenshot of the game can be seen in Figure 4.

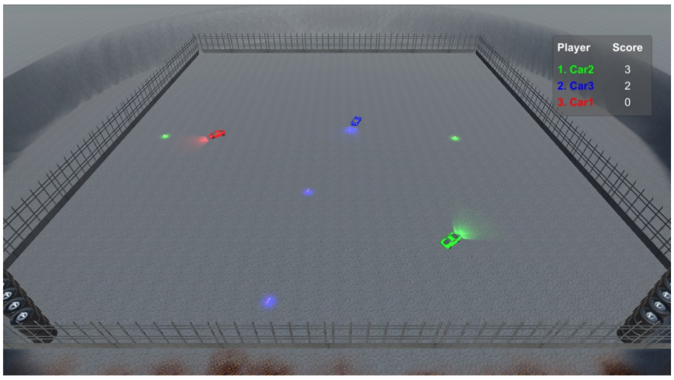

Figure 4. Our test game with 3 players connected.

C. Architecture

Communication between the server and clients is performed by sockets managed by the Unity NetworkManager library, and the controls' inputs are sent over the network as messages.

The server has a communication module only for receiving messages and forwarding them to their corresponding characters.

This module is responsible for saving the data of the different players that connect with their characters in the data structure of each player.

In the Auto Gamepad the central module of communication with the server is who receives the information of the interface to be sent over the network. The automatic control adjustment system is performed by the Interface Manager, which saves in memory the collected data from the user and then performs the processing.

V. Conclusions and future work

With this work, we develop and evaluate the operation of our adaptive control to the user, which is scalable with multiplayer games and maintains the efficiency of work already done in the literature.

It also has a low-cost implementation for projects due to its simple and reusable modular structure. It has the ability to maintain a friendly interface and without losing its visual identity, to provide greater convenience to its users.

Although it is still just an initial prototype, from this we will be able to apply efficiency tests with Auto Gamepad and really evaluate the effectiveness impacts, both in terms of latency by the number of players and the rate of errors in the use of the user interface.

It will also be possible to evaluate its use for different groups of players so that the control can identify patterns in the types of users to provide better results. In addition, it is intended to create an efficient way to optimize button size to better use the available interface.

Although we only have a simulated environment in Local Area Network (LAN), for the operation of the internet it is expected that the environment maintains the same basis of operation and communication, but with an additional impact on the latency and mode of operation of the game server.

One should be careful in choosing a number of inputs that the k-means clustering algorithm collects before performing the processing. A very small number can cause very sudden and imprecise visual changes whereas if it is too large it can be more costly and with little predictability of user actions. To better address this approach, we intend to use the Elbow algorithm for clusters analysis to validate the results using a larger collected data set, such as that performed in [9].

Acknowledgment

We would like to thank all the students and teachers involved who could make this happen.

References

[1] J. Yi, Y. Lee, and S. H. Kim, “Determinants of growth and decline in mobile game diffusion,” *Journal of Business Research*, 2019.

[2] J. R. da Silva Junior, M. Joselli, E. Clua, M. Pelegrino, and E. Mendonça, “An architecture for new ways of game user interaction using mobile devices,” *X Brazilian Symposium on Computer Games and Digital Entertainment, SBGames 2011*, pp. 1–4, 2011.

[3] S. M. Malfatti, F. F. Dos Santos, and S. R. Dos Santos, “Using mobile phones to control desktop multiplayer games,” *Proceedings - 2010 Brazilian Symposium on Games and Digital Entertainment, SBGames 2010*, pp. 230–238, 2011.

[4] L. Torok, M. Pelegrino, J. Lessa, D. Trevisan, and E. Clua, “AdaptControl: An Adaptive Mobile Touch Control for Games,” *Proceedings of SBGames 2013*, pp. 1–9, 2013. [Online]. Available: http://doi.acm.org/10.1145/2669062.2669081

[5] L. Torok, M. Pelegrino, D. Trevisan, A. Montenegro, and E. Clua, “Smart controller: Introducing a dynamic interface adapted to the gameplay,” *Entertainment Computing*, 2018.

[6] T. F. Costa, I. S. Silva, and A. R. Barbosa, “An Architecture for Using Smartphones as Interfaces for Computer Games,” *Proceedings of SBGames 2018*, pp. 611–614, 2018.

[7] J. K. Haas, *A History of the Unity Game Engine*. Worcester Polytechnic Institute, 2014, no. March. [Online]. Available: https://digitalcommons.wpi.edu/iqp-all

[8] K. A. A. Nazeer and M. P. Sebastian, “Improving the Accuracy and Efficiency of the k-means Clustering Algorithm,” *Proceedings of the World Congress on Engineering*, vol. I, no. May, p. 5, 2009.

[9] D. Lora, A. S. Antonio, P. A. Gonz, and A. G. Marco, “Dynamic Difficulty Adjustment in Tetris,” *The Twenty-Ninth International Flairs Conference*, pp. 1–5, 2016.