# Dynamic difficulty adjustment through parameter manipulation for Space Shooter game

Caetano Vieira Neto Segundo[1]* Kennet Emerson Avelino Calixto[1] Renê Pereira de Gusmão[2]

Faculdade Paraíso, Departamento de Sistemas de Informação, Brazil[1]
Universidade Federal de Sergipe, Departamento de Computação, Brazil[2]

**ABSTRACT**

An important criteria to be considered in the development of games is the game experience. It is directly linked to the difficulty of the game, which can be in the static or dynamic way. Players with different abilities in the same game may not fit in traditional difficulties models. The dynamic difficulty adjustment (DDA) provides a better gameplay, as the challenges in the game fit the player's abilities. Thus, this work proposes the development of the parameter manipulation technique for dynamically adjusting difficulty, aiming to improve the gaming experience. It is necessary to emphasize that the proposed approach uses probabilistic calculations that will be used in the challenge function. A questionnaire was applied to a sample of students in order to determine whether there were statistically significant differences in the perception of game play, difficulty of the game and desire to play several times with and without the use of the technique. The results showed that the dynamic version was better evaluated regarding game play and appropriate difficulty when compared to easy and hard versions.

**Keywords:** Artificial Intelligence, Dynamic Difficulty Adjustment, Digital games, Space Shooter, Parameter Manipulation.

## 1 INTRODUCTION

In the development of a game, a very important factor that should be assessed is the difficulty level and the types of challenges created for the player. In general, players choose the difficulty level they want to play at. Most of them see it as an obstacle the fact estimate their own abilities, and then end up playing a very easy or difficult game. Some of these problems are caused by the following: little variation in levels, gaps between one level and another and the lack of engaging challenges for the player as explained by Arulraj [9].

The different skills between players, of the same game, make the traditional model difficult to balance because it does not offer a level everyone feels comfortable with. This may be a problem because it can discourage many players. The users deception can cause them to have a bad experience and making it unlikely for them to play a particular game.

In this sense, balancing is a key factor in ensuring a pleasant gaming experience. This balancing is performed by the game designer, who performs an exhaustive amount of testing to find the features that represent the appropriate levels of the game as explained in Andrade, Ramalho and Santana [1].

In Hunicke [6], it's described that the dynamic balancing game consists of a mechanism that assesses the users performance in order to observe the challenges posed by the game. Then it automatically makes adjustments to the game in order to match the capacity of the player. On the other hand the static balancing is a traditional way of balancing, where the game has preset levels for all users and each of them chooses to start the match.

It is expected that the dynamic balancing provides a better game experience. The implementation of a DDA technique in the games provides consistent challenges to the players skills. Thus, it creates a fair scenario for the players. In this work the development of a dynamic balancing technique for the game Space Shooter will be discussed. The technique applied will be the manipulation of parameters. Where the challenge function will be defined through a probabilistic method that will be able to determine the situation of the player in the scenario.

The remainder of this work is structured as follows. Section 2 presents a review of some related works concerning dynamic difficulty adjustment applied to games. Section 3 presents an overview about the Space Shooter game and related concepts. Section 4 introduces the DDA technique applied to the Space Shooter. The results of the statistical analysis are shown in Section 5. The conclusion and future works are discussed in section 6.

## 2 RELATED WORK

In this section we discuss some related works. There are several works regarding dynamic difficult adjustment in digital games. Some of them are discussed below.

In Araujo and Feijo [2], the authors use the dynamic adaptively to evaluate casual and hardcore players, related to concepts of flow theory and the model core elements of the game experience. The authors also present an adaptive model for shooting games based on player modeling and online learning. The authors developed an implementation of a framework in Charles and Black [3] for player modelling and dynamic adaptivity. Two versions of the game were tested: an adaptive version using the implementation of the framework and a non-adaptive version. The results supported the idea that hardcore players had a better assimilation of the gaming experience with the adaptive version, as expected.

In Hunicke and Chapman [7], the authors explore and design requirements for a DDA system. They presented a probabilistic method for representing and reasoning about the uncertainty in games, also described the implementation of the proposed techniques and discussed how it can be applied to improve interactive experiences. In Hunicke [6], the author evaluated basic design requirements for effective DDA and presented an interactive DDA system. The study indicated that adjustment algorithms can improve performance, but retaining the players sense of agency and accomplishment.

The authors in Hawkins, Nesbitt and Brown [5] described a modeling technique known as particle filtering which is used to model several levels of the players ability while it can also consider the players risk profile since the performance in some games also de-

*e-mail: caetanov120@gmail.com

pends on the players desire to take risk. The authors demonstrated the technique by creating a game challenge where the player can risk more, take less time and fail more or the player can risk less, take more time to evaluate and fail less.

In Sha *et al.* [11], the authors contributed to proposing DDA as an approach to create an appropriate challenge level game opponent, in which the game used was Dead-End. Two kinds of DDA were proposed. The first is DDA by time-constrained-CI which is based on Monte Carlo for tree search. The second is DDA by knowledge-based-time-constrained-CI which is based on artificial neural networks. The former more appropriate for standalone PC game and the latter more appropriate for multi-player online games.

## 3 Discussion

In this section we present a discussion about Spacer Shooter game and some related concepts about game design and DDA.

### 3.1 Space Shooter

The chosen game for the implementation of the technique was the Space Shooter, an action game made with Unity [12]. The choice was based on the few number of commands that the game requires to be played, which makes it faster learning of the user. This game can be found in the unity repository and all of its assets are available free of charge.

In Space Shooter game, the player has a ship cover of ammo and life (the player starts with 100 hit points and 50 bullets in your inventory), and he faced his enemies and obstacles, in order to collect points. Throughout the match packages will be offered, they can increase ammo or life when in contact with the player's ship. The punctuation of the player is added only when it killing your enemy with your ammo, if the ship collides with his enemies, the ship loses life and the points remain.

For the development of the game, we used the game engine Unity. Unity provides ease and speed for the creation of 2D and 3D games. And its multi-platform resource includes platforms such as iOS, Android, Windows, Linux, SmartTVs. Moreover, it offered the developer learning resources, community and documentation in Unity [12].

### 3.2 Dynamic balancing

In literature, it proposes different techniques for DDA games. There are proposals that handle global mode game and others that work directly with the characters of the non-player characters game (NPC's), but that each of them can act correctly it is necessary to know the user level. To determine this level of difficulty, approaches found using a challenging call function. It aims to determine the difficulty experienced by the user at any given time of the game. According to the result of the function, the policy will be applied, which are aimed at setting the challenge to the player. For the application of any of a technical study of the problem is critical.

#### 3.2.1 Dynamic Scripts

Traditionally the NPC's has its behavior applied by the intelligent agents. The agent has several sensors like perceiving the environment and through its actuators, it can perform appropriate actions. The decision of the choice to be made is defined by the agent function. To understand the function, we created a table that describes the possible agent entries and each entry is recorded in Russell and Norvig [10].

#### 3.2.2 Genetic Algorithms

Some studies propose the use of this technique for balancing. Melanie [8] proposes the use of learning based on genetic algorithms through crossover and mutation are created new individuals. In this context it will not be the best selected, but those who improve to adapt to the user level. This ability is given by the heuristic function.

#### 3.2.3 Parameter manipulation

This is the approach that was used in the present paper. It has a global control of the variables, i.e, the whole game can be controlled by the algorithm. In this model if we apply the algorithm correctly, you can adjust any parameter of the play structure at the time that the player is active as shown by Hunicke [6].

This technique uses an economic vision market. This economic model of supply and demand parameters goal is to keep everything balanced as best as possible.

In the DDA, the supply parameter is seen as the items that the game offers to the user (health, weapons) and also the parameters of the characters (life, strength and endurance). Demand is set by manipulating the parameters of enemies (force, weapons, behavior). The implementations of game policies should cover enemies with different levels of difficulty. Thus the algorithm will have more level options to be dynamically chosen in Hunicke and Chapman [7].

In Space Shooter the offer of parameters analyzed are: hit points and ammunition; the demand parameters are: the enemy's behavior and strength. The balance between the curves ensure that the level is appropriate difficulty felt by the player. Figure 1 shows the flow channel that the player must meet throughout the game, avoiding the undesirable states that are too difficult or too easy in Hunicke and Chapman [7].


Figure 1: Difficult flow of the player, adapted from Hunicke and Chapman [7]

## 4 Proposed Methodology

In this section we introduce the methodology used for the dynamic difficult adjustment for the Space Shooter.

### 4.1 Heuristic function

It is necessary to determine the user level in the game, we assume that the level varies according to the damage suffered in a given time (t) in the game. We performed a probabilistic calculation with the damage of the player, and we have to answer an estimated death at a future time. In addition to the estimated death is associated with the amount of life the player to determine if the algorithm should act in some way.

The sum of the damages (d) at a time is given by the function (1) where $i$ represents the damage's value.

$$\sum_{i=1}^{n} d(i) \tag{1}$$

The set of times allows us to calculate the probability through the normal distribution function or Gaussian distribution, given by (2) function.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-1}{2}\left(\frac{x-m}{\sigma}\right)^2} \tag{2}$$

Although the Gaussian distribution manages the entire curve, it is necessary to calculate the mean and standard deviation, in which the mean is obtained by the equation (3). The calculation is done using a C# library that implements several methods. Meta Numerics supports on mathematical and statistical calculations to the .NET platform, such as the C# proposed by CodePlex [4].

$$\mu = \frac{t_1 + t_2 + \cdots + t_n}{n} = \frac{1}{n}\sum_{i=1}^{n} t(i) \quad (3)$$

### 4.2 Control zones

The area is the mechanism used to classify the level of the player at a given time instant. In this approach we used two areas: comfort and discomfort. Each has distinct characteristics and from them you can apply control policies.

In Figure 1, the comfort zone would be the part below the flow (very easy), so it would not be requiring any effort for the player, which may cause a lack of interest in the game. The top of the flow would be the discomfort zone (very difficult), it would be presenting challenges that exceed the ability of the player. The flow channel must be the site that the player will remain and the challenges are consistent with the ability of the same, but there is also unpredictability, since the player can change position within that stream without the algorithm reaction. Unpredictability becomes important for the game does not present repetitive challenges and end up discouraging the player.

Define the zone that the player belongs is essential for policies to be applied properly, however the values of each parameter should be studied and defined by a game design. In this work we do not discuss about the game design. In Space Shooter there are two areas according to the likelihood of damage and the percentage of the player's life. For the junction of these two variables are extremely important to know the level of difficulty that the player is feeling.

#### 4.2.1 Comfort zone

When the player is in the comfort zone, it was decided that the likelihood of damage is less than or equal to 70% and its life percentage is greater than or equal to 60%. In the comfort zone, the player will be in a hostile area with more enemies, more damage and less support (life and bullet) of the game.

#### 4.2.2 Discomfort Zone

This area defined the players who do not have many skills in the game, so that the user is characterized in that area, it was determined that the percentage living below 50% and the damage estimate over 55%. The player enters this area when he feels difficulty from the challenges posed by the game. In this area the player will have more support (life and bullet) of game, enemies in smaller quantities and with less damage.

#### 4.2.3 Flow Zone or Unpredictability

In this case we have an ideal scenario, i.e, the player is between the two previously mentioned areas. When you are in that area, the algorithm will not take any measure related to user assistance.

In this scenario, the game will provide challenges according to the player's abilities. Thus, the algorithm will not provide any help according to the player's health and ammo demand, but will continue running the assessments to interfere in the future if necessary.

### 4.3 Policies

Policies are interventions that manipulate the game parameters in order to move the user through the channel, as shown in Figure 1. After the classification of the player in the region, a policy will be applied, and may increase or decrease the challenges of the game, or simply keep as being.

Adjustments can be in or out of the stage. The actions taken in, are handled at the time of the confrontation, and if necessary the parameters are changed (enemy's strength, life, ammo). Changes offstage, are performed when the enemies are being generated and may modify the order, the amount and the damage.

The actions taken by the game will only be noticed by the user in the next wave, ie, adjustments are made out of the game scenario. It is believed that this will ensure a good unpredictability, because although in certain the player to be helped, it is not desirable that the algorithm is making changes in the scenario at all times.

#### 4.3.1 Settings

Adjustments are made through two functions. The first one is the assessment, which is performed by a subroutine of the game. In this function, the algorithm starts probabilistic calculation player death after 7 times of the game each time and set to 5 seconds. The function calculates the mean and standard deviation. Finally it is returned probability.

The second function is the challenge function that every 5 seconds performs a check of the values provided by the assessment and performs the necessary actions to adjust the difficulty regarding the characteristics of the areas in which the player is placed. The assessment and the adjustment is based on the probability of damage and life of the player. The adjustment can put the player in three different areas.

### 4.4 Example

Figure 2: Playing Space Shooter

In a practical example for three zones, as shown in the Figure 2, one can imagine a situation where the player has life in 40% and the probability calculated damage is approximately 70%, which makes higher the chances player die. At this point, the algorithm evaluation function feeds the challenge function, which will analyze the likelihood of damage and the players life consistent with the characteristics of the discomfort zone, and to help the player, the algorithm will provide a life box every 5 seconds and reduce the enemies' damage and amount. On the other hand, in a situation of the comfort zone, the algorithm would provide a life box every 20 seconds, increase the enemies damage and gradually the amount.

## 5 Results

In this section we present the results found by the statistical analyses made from the results of questionnaires applied to the players.

### 5.1 Descriptive statistics

To evaluate the perception of individuals with regard the appropriate difficulty, the playability of the game and the desire to play the game several times, a questionnaire with questions assessing these factors was applied to 30 people. For this, people played the easy version of the game and answered the questions facing cited variables, doing the same for dynamic and difficult version of the game. The answer scale used to the questions ranged from 1 to 5, where 1 meant totally disagree with the statement and 5 meant totally agree with the statement.

Regarding the evaluation of the best version after playing the three versions of the game, the participants mostly (18 people, representing 60% of the sample) classified the dynamic version as the

best version of the game. On the other hand, 9 people evaluated the hard version as the best and 3 people evaluated the easy version as the best.

Regarding the score of the participants, half of them performed better in the easy version of the game, representing 50% of the sample. In the dynamic version, 15 of them (50% of the sample) also had better scores, which supports the assessment of participants who rated above the dynamic version as being the best. Thus, when people played the dynamic version they were as good as when they played the easy version of the game. For the hard version no one got better score, as expected.

### 5.2 Inferential statistics

In order to compare the easy and the dynamic versions of the game with regard to game play, appropriate difficulty and will play the game several times, some T test for paired samples were made in order that the same people participated the three experimental conditions (easy game, dynamic game, difficult game).

Table 1 presents the results of the comparisons between the easy and dynamic version of the game, in which GP means Game play, AD means appropriate difficult and PST means play several times.

Table 1: Easy version vs. dynamic version

| | Easy | | Dynamic | | | | |
|---|---|---|---|---|---|---|---|
| | M | SD | M | SD | t(df) | p | 95% CI |
| GP | 4,03 | 0,76 | 4.27 | 0.69 | -2,25(29) | 0,032 | -0,45;-0,02 |
| AD | 2,40 | 1,25 | 4,30 | 0,60 | -8,78(29) | 0,001 | -2,34;-1,46 |
| PST | 2,63 | 1,35 | 4,03 | 0,76 | -5,66(29) | 0,001 | -1,91;-0,89 |

Regarding the game play, the test results showed a statistically significant differences regarding game play easy and dynamic version of the game ($t(29) = -2.25, p < 0.05$) . In this case, the participants evaluated the dynamic version as better ($M = 4.27, SD = 0.69$) compared to the easy version of the game ($M = 4.03, SD = 0.76$). Thus, it can be said that the dynamic version was evaluated as the one best to play, comparing the easy version of the game.

The results of t-test showed that there are statistically significant differences in the assessment of appropriate game difficulty ($t(29) = -8.78, p < 0.001$). The dynamic version has more difficulty ($M = 4.30, SD = 0.60$) compared adequate easy version of the game ($M = 2.40, SD = 1.25$). Therefore, the participants believe that the dynamic version has a better difficulty, or better play.

Regarding the willingness to play the game repeatedly, the t test showed that there is also significant differences in the two versions($t(29) = -5.66, p < 0.001$). The participants felt more desire to play the dynamic version ($M = 4.03, SD = 0.76$) instead of the easy version ($M = 2.63, SD = 1.25$).

Table 2: Hard version vs. dynamic version

| | Hard | | Dynamic | | | | |
|---|---|---|---|---|---|---|---|
| | M | SD | M | SD | t(df) | p | 95% CI |
| GP | 3,97 | 0,96 | 4,27 | 0,69 | -2,07(29) | 0,048 | -0,60;-0,003 |
| AD | 2,60 | 1,45 | 4,30 | 0,60 | -5,53(29) | 0,001 | -2,33;-1,07 |
| PST | 3,70 | 1,34 | 4,03 | 0,76 | -1,33(29) | 0,194 | -0,85;-0,18 |

Table 2 presents the results comparing the hard version with the dynamic version. For the gameplay, the results of t test revealed a statistically significant difference when comparing the version difficult with the dynamic version ($t(29) = -2.07, p < 0.05$). The participants assessed the dynamic version ($M = 4.27, SD = 0.69$) as the best gameplay compared to hard version (M = 3.97, SD = 0.96) of the game.

Regarding the appropriate difficult for these versions, there was also statistically significant difference ($t(29) = -5.53, p < 0.001$). Participants reported that the dynamic version ($M = 4.30, SD = 0.60$) has more appropriate difficulty compared to hard version of the game ($M = 2.60, SD = 1.45$).

For the desire to play the game several times, the difficult version ($M = 3.70, SD = 1.34$) did not differ from the dynamic version of the game ($M = 4.03, SD = 0.76$), ($t(29) = -5.66, p < 0.001$). The participants would like to repeatedly play the two, with no statistically significant differences in this desire, even if the dynamic version has been evaluated with a higher average.

## 6 Conclusion

This paper proposed the use of parameter manipulation technique for dynamic difficulty adjustment applied to the game space shooter. A sample of people tested the game in three difficulties, two of them static (easy and hard) and the third dynamic. After playing in three versions, one questionnaire was used to assess the perception of people as the game play, difficulty and will play several times the game for each version.

A statistical analysis was performed to determine whether there were statistically significant differences in the responses of the questionnaires. The results of this analysis showed that the version with dynamic balancing achieved the best results when compared to the other versions regarding the game play and will play several times. As the score, half the people in the sample had better scores in the easy version and the other half had the best score in the dynamic version.

The game is in an initial version and can be improved in future. The improvements can serve for academic purposes for maximize the understanding, demonstrate and apply the method of DDA using the parameter manipulation. As future works, we suggest resources such as new phases for the game, new enemies and improve the interface, offering this ways a better game experience. We also suggest identify the viability of other techniques of dynamic balancing.

## References


[1] G. Andrade, G. Ramalho, H. Santana, and V. Corruble. Extending reinforcement learning to provide dynamic game balancing. In *Proceedings of the 2005 IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games*, pages 7–12, 2005.

[2] B. B. P. L. Araujo and B. Feijo. Evaluating dynamic difficulty adaptivity in shootem up games. In *SBC Proceedings of SBGames 2013*, pages 229–238, 2013.

[3] D. Charles and M. Black. Dynamic player modeling: A framework for player-centered digital games. In *International Conference on Computer Games: Artificial Intelligence, Design and Education*, page 2935, 2004.

[4] CodePlex. The meta.numerics math and statistics library.

[5] G. Hawkins, K. Nesbitt, and S. Brown. Dynamic difficulty balancing for cautious players and risk takers. *International Journal of Computer Games Technology*, 2012, 2012.

[6] R. Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 429–433, 2005.

[7] R. Hunicke and V. Chapman. Evaluating dynamic difficulty adaptivity in shootem up games. In *Challenges in Game Artificial Intelligence AAAI Workshop*, pages 91–96, 2004.

[8] Melanie Mitchell. An introduction to genetic algorithms. pages 4–23, 1999.

[9] A. Joy James Prabhu. Adaptive agent generation using machine learning for dynamic difficulty adjustment. *IEEE*, pages 746 – 751, 2010.

[10] J. Stuart Russell and P. Norvig. Artificial intelligence: A modern approach. pages 31–33, 2013.

[11] L. Sha, S. He, J. Wang, J. Yang, Y. Gao, Y. Zhang, and X. Yu. Creating appropriate challenge level game opponent by the use of dynamic difficulty adjustment. In *Sixth International Conference on Natural Computation*, pages 3897–3901, 2010.

[12] Unity Technologies. Space shooter tutorial.