Heloisa Matta da Cunha Pessoa

A Culture Model Enhancement Project: First-Person and Virtual Reality Gaming Simulation

RELATÓRIO DE PROJETO FINAL

DEPARTAMENTO DE INFORMÁTICA
Programa de graduação em Engenharia de Computação

Rio de Janeiro
Dezembro de 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Heloisa Matta da Cunha Pessoa

A Culture Model Enhancement Project: First-Person and Virtual Reality Gaming Simulation

Relatório de Projeto Final

Relatório de Projeto Final, apresentado ao programa Engenharia de Computaçãoda PUC-Rio como requisito parcial para a obtenção do titulo de Engenheiro de Computação.

Orientador: Prof. Bruno Feijó

Rio de Janeiro
Dezembro de 2022

Summary

List of Figures

List of Tables

1
Abstract

"Emotions, the most unpredictable aspect of a person, but also a common aspect" Unknow source.

The desire to experience the magic and the unbelievable furthers us in searching for inventions and innovations that shall turn what is part of the unreal world into reality as closely as possible. This work proposes a culture model that brings a reliable and efficient emotional simulation, affected by cultural and personality factors, based on previous models of emotions and culture developed at the ICAD/VisionLab laboratory. The proposed extension to the earlier models is adding Fuzzy Logic and considering a 3D and VR environment.

Key Words: NPCs behavior; Culture model; Game AI; Cultural behavior; Emotion and Personality Models; Proxemics; Fuzzy Logic;

2
Introduction

It holds no secrecy that humanity is always seeking that next step and advances in technology. It's a natural process and part of our nature to try, discover and create gadgets made to help make our lives easier. Video games have been in people's daily routine often as a resource to escape their reality, or coping with certain problems, such as stress, sociability, romance and frustrations [6]. In October's 21th, 2021, *Facebook's* company name was changed to *Meta* and the *Metaverse* was announced, a virtual world embed with purposes of fun, work, commerce and socialization. This innovation path *Meta* decided to invest is a clear response on how gaming industry has been evolving, primarily due to COVID-19 pandemic keeping everyone mandatorily indoors, giving people no choice but to use technology in order to socialize and play, which increased absurdly the gaming cluster [**1**].

Due to that, we find our society in general gradually becoming more involved with the virtual world, but this idea of a world that resemble ours (only with greater perspective and possibilities) wasn't born with *Metaverse*: It exists in our culture for far longer than we think, being said that it's first representation was in *The pygmalion's spectacles*, 1935 (Figure 2.1). [19], serving as context and theme for movies (*Tron*, 1982), books (*Ready Player One*, 2011, which gained it's movie adaptation in 2018), offering small pieces in open-world games such as *Rockstar's GTA V*, *CD Projekt RED's The Witcher: Wild Hunt*, that provides us with a taste of freedom and exploration, taking away the fear of getting hurt and it's consequences.

The present work aims to surf this new post-pandemic gaming era by taking another step into recreating our reality inside a virtual world, specifically, a video game. In 2017, professor Augusto Baffa wrote the paper *Dealing with the emotions of Non Player Characters*, in which he fought the idea of simplified NPC's(non-playable characters) inside video games, that normally give automatic predefined responses, scripted and deterministic [1]. For those who are unfamiliar with the term: A NPC is the character that populate video games and aren't represented by a real person, mainly being there to help the player trough the game, to be a part of the story, or just act as extras.


PYGMALION'S SPECTACLES

By STANLEY G. WEINBAUM

Author of "The Black Flame," "A Martian Odyssey," etc.

© 1935 by Continental Publications, Inc.

Unbelieving, still gripping the arms of that unseen chair, Don was staring at a forest

Figura 2.1: *"You are in the story, you speak to the shadows and they reply, and instead of being on a screen, the story is all about you, and you are in it" - The pygmalion's spectacles, 1935 [14]*

As stories in games evolve in quality, players experience more dense dialogues and character development from these NPC's, and it's inevitable that we feel empathy, friendship, as well as hatred and annoyance, overall not being able to feel indifferent towards them. We observe this in games like *God Of War*(2018), with Kratos's relation with his son, Atreus. *Naughty Dog's The Last Of Us*(2013, 2020) whole saga, that throws us in a rollercoaster of emotions gathering the two protagonists. And in V's relation with Johnny Silverhand from *Cyberpunk 2077* (2020), an already beloved NPC even before the game was released[1].

On the path of creating a more realistic virtual world by bringing intelligent responses to NPCs, we find Luis Fernando Bicalho's paper [**7**], entitled "A Culture Model for Non-Player Character's Behaviors in Role-Playing Games,"which uses Baffa's emotional model [1]. In Bicalho's paper, the game varies its responses according to the NPC's personality, culture, and humor (being represented as a current emotion). The present work aims to enhance Bicalho's model [7] by incorporating Fuzzy Logic into its emotion processing, and adding more cultural factors (which ended up staying for future improvements). The implementation uses the Unity game engine (coded in C) alongside the Flask API (coded in python). Also, it uses the libraries NumPy and Scikit-Fuzzy. Furthermore, we implement these extensions alongside a VR test in an FPS (first-person shooter) game.

This work is organized as follows. Section *Related Works* will present an analysis regarding games and projects that offer the same ideals, on what concerns a most diverse NPC experience and cultural simulation responses, as

[1] *Johnny Silverhand's* NPC character was played by actor Keanu Reeves, who also heavily promoted the game, bringing popularity and notoriety to it.

this current one, and what approaches they have done. In *Previous Models*, the details for both Baffa and Bicalho's models are explained, regarding their functionalities, what has changed and whatnot. The section *Fuzzy Logic* will walk through important concepts of Fuzzy Logic that were used in this work, as well as what has been made on the model with this concepts, and how. The *System Modeling, Implementation and Assets* will address what technologies were necessary, why and it's specifics, as well as details from the code.*Game Mechanics* will briefly explain the game functionalities, and also game design elements. *Conclusion and Future Work* are a closure of our work presenting final considerations and future research directions.

3
Related Works

This work contains three main characteristics that defines it's nature: *game AI*, *cultural simulation* and *emotional simulation*. Although we have some examples in the market for games that work with these characteristics, there are few to none that actually gather all three. The same can be said for academic projects, as we have only punctual approaches regarding each topic, but rarely one that embraces all together.

3.1
Game AI and Cultural Simulation

Normally, when it comes to culture in games, there's a different approach than what we see in *The Culture Model Project*. The AI implemented is not made to stimulate different reactions on a personal interaction level, but mostly in order to react in political and strategical ways. This type of AI behaviour is fairly common in turn-based strategy[1] games, like the *Civilization* series, and *Humankind*. This AI's are not made to be "perfect", but aims on offering a good challenge to the player and on top of all, a fun experience [13]. In *Humankind* and *Civilization* case, the cultural simulation is inherent to the game, as it affects how each player's people and territory will evolve, and how it's surroundings neighbours will treat them.

Specifically in *Humankind's* example, we have a very interesting factor, as the game actually allows the player to merge two or more different cultures to produce new perspectives. Concerning AI, that's one of the most interesting examples in video game. But cultural factors have been approached in various forms regardless of Game AI, and became very popular: We can cite the entire *Assassin's Creed* series, that bases it's games off several different cultures, making it the context, the NPC behaviour, and even the central part of the story. Unfortunately, this games still have a very deterministic approach upon NPC responses and possibilities, but it's cultural factors are extremely well represented and a mark in game industry.

[1] *"A turn-based strategy (TBS) game is a strategy game (usually some type of wargame, especially a strategic-level wargame) where players take turns when playing. This is distinguished from real-time strategy (RTS), in which all players play simultaneously"* [2].

3.2 Game AI, Fuzzy Logic and Emotional Simulation

Taking away the cultural influence, it gets easier to find more examples, even approaching Fuzzy Logic as well. In the papers "A.I. Techniques for Modelling Anger in Emotional Agents"[20], "Emotion Detection from Text using Fuzzy Logic"[17] and "An Emotion-Processing System Based on Fuzzy Inference and Its Subjective Observations"[23] we have examples of emotional processing through Fuzzy Logic, and even related to a gaming context, seen in [20]. A conclusion paper presented inside PUC-Rio university in 2018 by student Simon Rozenberg Travancas[18] also used Fuzzy Logic alongside a Newtonian Mechanical model that interprets Plutchik's Wheel in 4 dimensional R space to create a library focused on emotional processing inside a gaming context.

Torao Yanaru, Toyohiko Hirotja, and Naoki Kimura[23] studied a few paths to experiment with emotional processing and decided upon working with Plutchik's theory of emotions[15] and Fuzzy Logic (both of which are also used in this current work), and despite being far from a gaming context, utilized this processing to understand what emotions were triggered by certain images and its transitions. Their Fuzzy Logic focused not in the basic emotions transition, but mostly from mild emotions to intense emotions, also from neutral to positive and negative feelings. Similarly, Saqib Qamar and Parvez Ahmad[17] approached emotional processing through Fuzzy Logic, utilizing text stimuli, also aimed upon measuring solemnly intensity and nature (positivity, negativity) of emotional responses.

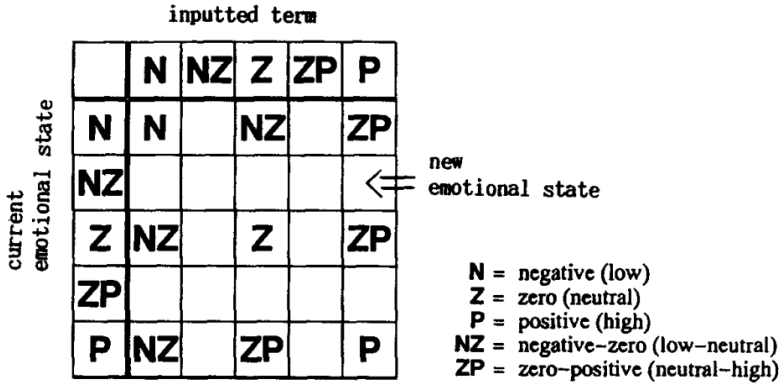

Figura 3.1: Fuzzy rule table relations taken from Torao Yanaru et al. [23]

Lastly, the first paper cited in this section, by Sarah Slater[20], is one that focuses more in a gaming context. Working with Finite State Machines[2],

[2] *"Finite State Machine (FSM) have been used for many years in video games to model the AI of Non Playing Characters (NPCs). It is one of the easiest ways to model and code*

she aimed upon giving NPC's, made mainly as enemies for the player, more humane reactions when triggered by anger or its emotional variants, therefore adding gestures and animations for different stimuli.

a character's behavior in a finite number of states. The model includes the states and the transition between these states, which changes according to input from the game."[4]

4
Previous Models

As stated before, this work is made upon Bicalho's Cultural Model, which was heavily based of Baffa's Emotion and Personality Model, hence the two main models that'll be explained in this chapter. In order to fully understand how Bicalho's model processes these factors altogether, this section will firstly go through the basic settings of personality and emotion set by Baffa, to then explore how these structures were gathered with cultural factors and processed by Bicalho.

4.1
Emotion and Personality Model

4.1.1
Emotion

Transforming emotions, a subjective matter, in a measurable dimension is not an easy task and to do so, Baffa created his way of gauging emotions through *Plutchik's Wheel of Emotions* (Figure 4.1), based on Plutchik's study from 1962 [15].

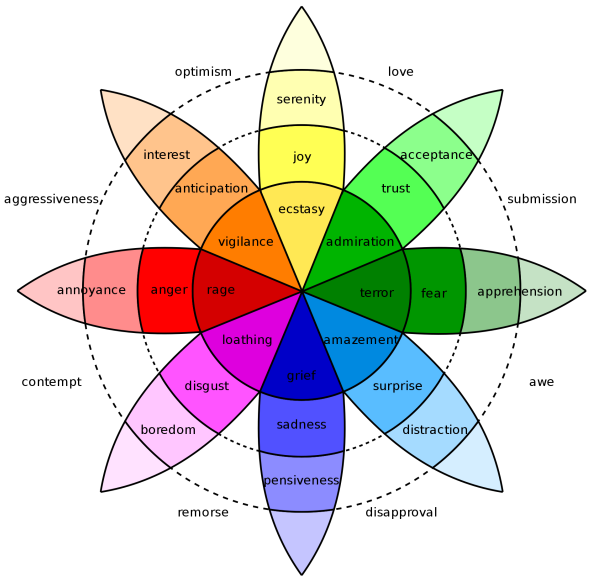

Figura 4.1: Plutchik's Wheel of Emotions

As you can see, each axe of emotion has determined intensities, taking by example the *Joy* axe: *Serenity* is a low intensity, the light form of this emotion, *Joy* is medium intensity and *Ecstasy* is high. This measure will be vital for when applying Fuzzy Logic in a few sections ahead. These axes will consist of three emotions designed as basic emotions, which can be combined with other axes's basic emotions. By simplifying Plutchik's wheel, Baffa, considering four axes ranging 0.0 to -1.0 and 1.0 to 0, absolute value 1.0 being the most intense basic emotion, built Figure 4.2. In Baffa's words, he explains that "*Plutchik defines that basic emotions can be combined in pairs to produce complex emotions. These combinations are classified in four groups: Primary Dyads (experienced often), secundary Dyads (sometimes perceived), Tertiary Dyads (rare) and opposite Dyads (cannot be combined). Primary Dyads are obtained by combining adjacent emotions, e.g., Joy + Trust = Love. The secundary Dyads are obtained by combining emotions that are two axes distant, for example, Joy + Fear = Excitement. The Tertiary Dyads are obtained by combining emotions that are three axes distant, for example, Joy + Surprise = Delight. The opposite Dyads are on the same axis but on opposite sides, for example, Joy and Sorrow cannot be combined, or cannot occur simultaneously (...)*".

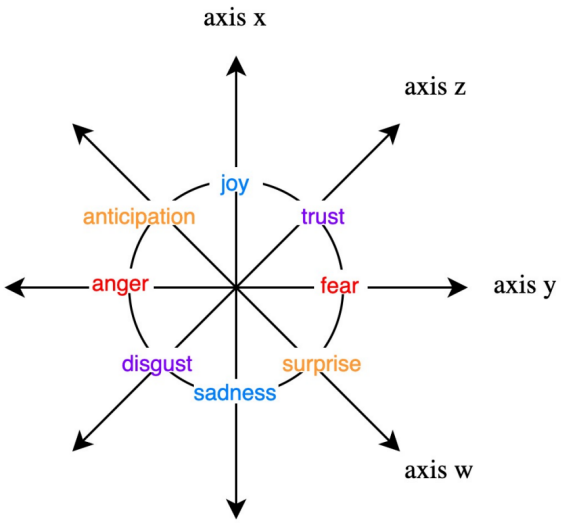

Figura 4.2: Bicalho's graphic reproduction[**7**] of Baffa's simplified 4-axis structure

Values were defined for each intensity, regarding the range cited above. Low intensity basic emotions were defined by absolute value of 0.2, medium by 0.5 and high by 1.0. These definitions can be seen in Table 4.1 and it's opposites in Table 4.2. These defined values are the main point on what's going to change when applying Fuzzy Logic, since it's primarily the deterministic part of choosing emotions that Fuzzy strives to modify.

1. Anger x Fear (A-F): Has value in the interval [-1, 1];
2. Joy x Sadness (J-S): Has value in the interval [-1, 1];
3. Anticipation x Surprise (A-S): Has value in the interval [-1, 1];
4. Trust x Disgust (T-D): Has value in the interval [-1, 1];

Mild emotion (0.2)	**Basic emotion (0.5)**	Intense emotion (1.0)
Serenity	**Joy**	Ecstasy
Acceptance	**Trust**	Admiration
Apprehension	**Fear**	Terror
Distraction	**Surprise**	Amazement

Tabela 4.1: Main emotions and its values, taken from Bicalho's paper [**7**]

Intense opposite (-1.0)	**Basic opposite** (-0.5)	Mild opposite (-0.2)
Grief	**Sadness**	Pensiveness
Boredom	**Disgust**	Loathing
Annoyance	**Anger**	Rage
Interest	**Anticipation**	Vigilance

Tabela 4.2: Main opposite emotions and its values, taken from Bicalho's paper [**7**]

Previously, the dyads combination were all set to behave according to the values combination seen in Table 4.3, where the medium category (absolute value 0.5) would serve as identification for joining two basic emotion axes. All the possible combinations are being taken account inside the model and can be seen in Tables 5.1, 5.2 and 5.3, inside section *Fuzzy Logic*, where they'll be further explored.

Primary Dyads	Combination	Values in Axis
Optimism	Anticipation + Joy	0, 0, 0.5, -0.5

Tabela 4.3: Example of primary dyad and its value, adapted from [**7**]

4.1.2 Personality

There are plenty of studies on how to map an individual's personality, but the one used on the model and the focus here will be mainly on *Big Five* [12] theory, also known as O.C.E.A.N theory. Each of the OCEAN factors below represents an aspect that builds a person's personality. It's relevance for the model is represented by how these factors affects each basic emotion axe (Figure 4.2), which can be seen in Table 4.4.

1. Openness to experience: Reflects how much an individual likes and seeks for new experiences.

2. Conscientiousness (Scrupulosity): Reflects how much careful and organized is an individual.

3. Extraversion: Reflects how an individual is oriented to the external world and get satisfaction from interacting with other people.

4. Agreeableness (Sociability): Reflects how much an individual like and try to please others.

5. Neuroticism (emotional instability): Neuroticism is the tendency to experience negative emotions. [1]

O	C	E	A	N	
-1	0	0	0	1	Fear
1	0	1	1	-1	Trust
1	0	1	1	-1	Joy
-1	0	1	1	1	Surprise
0	-1	0	0	1	Anger
-1	0	0	-1	1	Disgust
-1	1	1	0	1	Sadness
-1	1	-1	-1	1	Anticipation

Tabela 4.4: Personality traits influencing basic emotions, adapted from [**7**]

4.2 Cultural Model

4.2.1 Actions

In order to trigger a response from NPC's, an arrange of possible actions inside the game were defined, and associated with possible emotional consequences relating to that certain action. These relation can be seen in Table 4.5. Imagine a player decided to attack a NPC. As we can see, the possible reaction towards an attack are *Rage*, *Outrage*, *Despair* or *Terror*, which are emotions related to *Anger-Fear* axe. Keep in mind this is not the final emotion response, but a part of the path to calculating it's results. This first emotional response is called *event emotion*.

Player's Action	Possible event emotion
is_attacking	Rage, Outrage, Despair, Terror
is_shooting	Annoyance, Pessimism, Disapproval, Apprehension
is_harming	Anger, Contempt, Unbelief, Fear
is_injured	Anger, Disapproval, Pride, Joy
is_giving_item	Joy, Optimism, Hope, Trust
is_stealing_item	Sadness, Shame, Remorse, Disgust
is_giving_money	Admiration, Love, Sentimentality, Ecstasy
is_stealing_money	Grief, Dominance, Awe, Loathing
is_talking_politely	Boredom, Envy, Pride, Serenity
is_not_talking_politely	Pensiveness, Guilt, Morbidness, Acceptance

Tabela 4.5: Player's actions and the resultant NPC's emotions, based on the cultural dimension value range, taken from Bicalho's paper [**7**]

4.2.2 Proxemics

In Bicalho's words, *"Proxemics is the scientific area which studies human's use of space and the effects that population density has on behaviour communication, and social interaction"*. His studies derived from Edward T. Hall's *The Hidden Dimension* [3]. In this model, proxemics will work as another possibility of getting an emotional response from the NPC, which will vary according to how distant the player is standing from them. This relations are seen in Table 4.6.

Player's Action	Possible event emotion	Distance interval
is_social	Distraction, Anxiety, Delight, Interest	[0.7,0.45)
is_personal	Anticipation, Cynicism, Curiosity, Surprise	[0.45,0.25)
is_intimate	Vigilance, Aggressiveness, Submission, Amazement	[0.25,0)

Tabela 4.6: Proxemics triggers player's actions and the possible resultant event emotions, and action is defined by the distance between the player and NPC.

4.2.3 Cultural Factors

Uniting two studies upon social metrics and behaviour in society, Bicalho built this model utilizing four cultural metrics taken from Bölöni [5] et al., adding a parameter for "rationality", and adding a final metric taken from Hofstedes study [10].

Rationality was a metric added in order to weight on how intense our first emotional response, *event emotion*, would be.

1. Rationality: Its value affects how intense the resultant emotion will be (further explained throughout this report).

   From Bölöni studies, he added the following metrics.

2. Time: Influences how fast the NPC will move, multiplying his velocity value;

3. Wealth: Represents how much the player will care about player actions like: money give away, money theft, item give away or item theft. So, if its value is 0.7, this NPC is highly affected by this cited actions.

4. Dignity: Related to situations in which the player invades NPC's social, personal or intimate spaces. It also influences how the NPC will be emotionally affected, after he is shot or harmed.

5. Politeness: Represents how much a NPC is affected by a non-polite player approach or conversation;

   The final addition, adapted from Hoftede's theory, was *collectivism*.

6. Collectivism: Represents how much a NPC cares about others, when they are inside their public space;

It is cited in Bicalho's paper that he decided to add these last two metrics for the purpose of gaming context, that although purely using Bölöni's parameters are sufficient to define a good culture, it lacks on acting upon certain player's actions as for example: acting on a nearby NPC would trigger *collectivism*.

Each of this cultural factors are associated with possible player actions and are triggered for when calculating emotional responses. This relations can be seen in Table 4.7.

Player's Action	Cultural Dimension
is_attacking	dignity
is_shooting	collectivism
is_harming	collectivism
is_injured	trust_level
is_giving_item	wealth
is_stealing_item	wealth
is_giving_money	wealth
is_stealing_money	wealth
is_social	dignity
is_personal	dignity
is_intimate	dignity
is_talking_politely	politeness
is_not_talking_politely	politeness

Tabela 4.7: Player's actions and the related cultural dimension, retrived from Bicalho's paper [**7**]

4.2.4 Trust

The trust level will be the final step of the model, which is triggered by the emotional response of one mosts *influent emotion* inside of the NPC, towards the player. Each *influent emotion* has it's basic emotion axe related (Table 4.9), and each basic emotion has it's factor for calculating the trust (Table 4.8).

Mental State	Related Emotions
Anger	Rage, Anger, Annoyance, Outrage, Envy, Aggressiveness, Dominance
Fear	Apprehension, Fear, Terror, Guilt, Awe, Despair
Trust	Acceptance, Trust, Admiration, Hope, Submission
Disgust	Loathing, Disgust, Boredom, Unbelief, Contempt, Shame
Joy	Serenity, Joy, Ecstasy, Optimism, Love, Sentimentality, Morbidness, Pride
Sadness	Grief, Sadness, Pensiveness, Disapproval, Remorse, Pessimism
Surprise	Distraction, Surprise, Amazement, Delight, Curiosity
Anticipation	Vigilance, Anticipation, Interest, Anxiety, Cynicism

Tabela 4.8: Available mental states and its related emotions, retrived from Bicalho's paper [**7**]

Mental State	Trust Level Factor
Anger	-1
Fear	-1
Trust	+1
Disgust	-1
Joy	+1
Sadness	-1
Surprise	+1
Anticipation	-1

Tabela 4.9: Mental states and its related factors

4.2.5 Model

Finally, gathering everything that has been seen in this section, it is possible to connect all the dots, step by step: Firstly, the player makes an action. Then, this action will trigger a first emotional response, as stated before, called *event emotion*. *Event emotion* is calculated through Equation 4-1, taking the squareroot of one minus *rationality* multiplied by the value of the associated *cultural dimension* with the action commited by the player.

$$CulturalFactor = \sqrt{(CulturalDimension \times (1 - Rationality))} \qquad (4\text{-}1)$$

The *cultural factor* defines how intense the event emotion will be, which is selected through the options seen in *Actions* subsection. A *new emotion* is calculated by taking account of personality influence altogether with *event emotion*. This calculation can be seen in Equation 4-2, where p_j is each OCEAN factor value from NPC's personality, and $factor_{ji}$ is the OCEAN influence on each basic emotion axe.

$$NewEmotion_i = \frac{\left(\sum_{j=1}^{5} EventEmotion_i \times p_j \times factor_{ji}\right)}{5}, i \in [1, 4] \qquad (4\text{-}2)$$

Then, this *new emotion* is clamped and added to the NPC's *current emotion*, as seen in Equation 4-3.

$$(CurrEmo_i)_{t+1} = (CurrEmo_i)_t + Clamp(NewEmo_i, -1.0, 1.0) \qquad (4\text{-}3)$$

Then, a few steps are made using *current emotion* in order to select which intensity will be used to define the resultant emotion [**7**]:

1. Iterate through the current emotion values and find the biggest one (comparing absolute values);
2. Iterate again through the emotion values, comparing each one to the biggest value. If they are equal, the influent emotion receives this value, otherwise receives zero.
3. Iterate through the influent emotion values, rounding its values following some rules:
   (a) If there are two values bigger than zero, they both will be considered 0.5;
   (b) Otherwise, it will respect the following rules:
      i. If $Value <= 0.1$, $Value = 0$;
      ii. If $Value <= 0.3$, $Value = 0.2$;
      iii. If $Value <= 0.5$, $Value = 0.5$;
      iv. If $Value > 0.5$, $Value = 1.0$;

After selecting which is the most *influent emotion*, the NPC's mental state alters his trust level using the criteria seen in subsection *Trust*. This trust level is calculated through Equation 4-4.

$$tLvl_{t+1} = tLvl_t + pLvl \times mentalFactor \tag{4-4}$$

To summarize, here's a short version of all steps mentioned above:

1. Player makes an action towards NPC. (Proxemics calculation works as an action).
2. Action is related to an array of possible emotions that will become the first response towards that specific action.
3. Cultural dimension is added according to which action relates to it.
4. The *cultural factor* is calculated using that *cultural dimension* value from that NPC, adding his rationality to the equation, and the array for the *event emotion* gets calculated.
5. This *event emotion* is used to calculate a new emotion, utilizing the personality factors from NPC and their determined influence on each axe of basic emotion.
6. This new emotion is clamped, summed with NPC's current emotion, and from there, check for intervals and set most accordingly to each level of intensity. Fuzzy Logic will substitute this method.

5
Fuzzy Logic

Fuzzy Logic is an AI that allows us to work with diffuse concepts on programming language. An escape method from sharp, well-defined boundaries such as "true"or "false", also know as crisp valued parameters. Therefore, it‘s the perfect solution when it comes to emotions, since it‘s a rather subjective concept to work on mathematical understandings of it. This section will walk through the basic settings for a Fuzzy Logic, and how each concept were applied in the current Culture Model, all of which were heavily planned and written based off the book *Programming Game AI by Example*, by Mat Buckland [7].

5.1
Crisp sets

Fuzzy's whole point is to work with vague boundaries, but that's not how our computer understand variables. Therefore, we must begin our calculations based off sharp values, or *crisp values*. In order to define a fuzzy variable, it is presented a concept called *degree of membership*, which defines how present that variable is inside a definition or categorization.

Taking a crisp value as example, on Buckland's words: *"(...)the integer 3 is a member of the set of odd numbers, the set of prime numbers, and the set of all numbers less then 5. But in all these sets its degree of membership is 1."*[7]. In our case, the degree of membership will be related to how intense a resultant emotion can be, for example, an emotion can be 0.23 (23%) medium and 0.77 (77%) high intensity. However, this membership functions will be built according to each emotional axe in their own respective intensities.

The crisp values entry for the model will be the first 4 axis emotion array calculated in the current model state, beginning by *event emotion*, generated as explained in Equation 4-1, gathering personality factors to generate the *new emotion* array, clamped and summed with *current emotion* array (also explained in the previous section), and then processed by Fuzzy Logic for it to be interpreted as a solid emotion response.

5.2
Fuzzy set

The degree of membership is defined by a Fuzzy Set's *membership function*. As said before, in our case, the emotional response will be evaluated according to it's *intensity*, which will compose the Fuzzy sets present in the model. The membership function is what's going to build fuzzy boundaries, and it is normally made upon some predefined function shapes, chosen according to what function represents best the current problem. In Figure 5.1, we can see the most common membership functions used in Fuzzy sets.

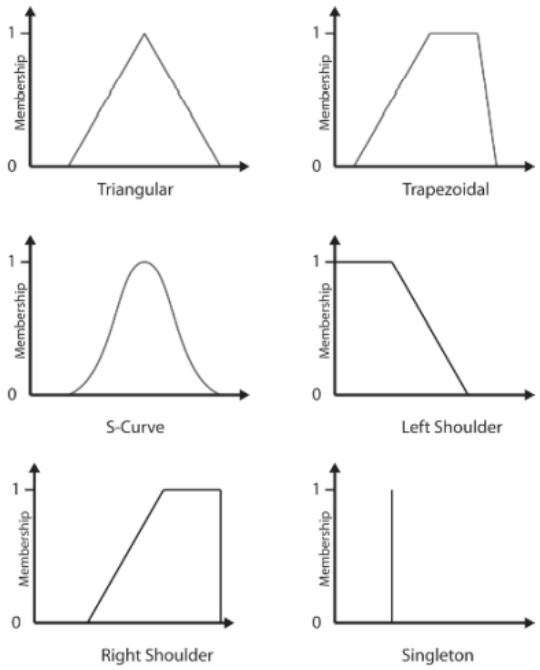

Figura 5.1: Figure from Buckland's book on chapter 10 "Fuzzy Logic", regarding membership functions[7].

5.2.1
Basic Plutchik's Axes

There are a few methods to calculate which function suits what problem the best. In our case, it isn't necessary to create and organize our variables with clustering, or work with undefined boundaries, since we have four defined emotional axes, with six possible intensities on each basic emotion axe: *negative low*, *negative medium*, *negative high*, *positive low*, *positive medium* and *positive high*. Here, it's assumed that the pattern of probability is the same for the less intense and most intense emotion, but a bit more likely on the medium intensity. Therefore, dividing its currency by six triangular membership functions. Each basic emotion axe is going to be divided by

intervals of [-100,100] following it's positive and negative parameters for calculating the resultant emotion (Following the behaviour seen in Table 4.5). We can see in Figure 5.2 how *Anger-Fear* Fuzzy Set would be.

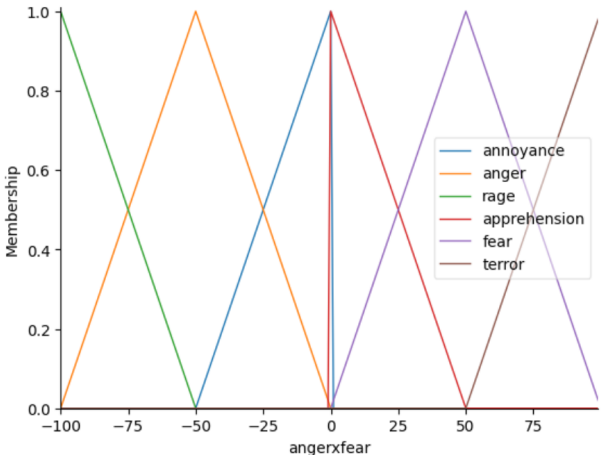

Figura 5.2: Fuzzy Axe Anger-Fear, with it's membership functions

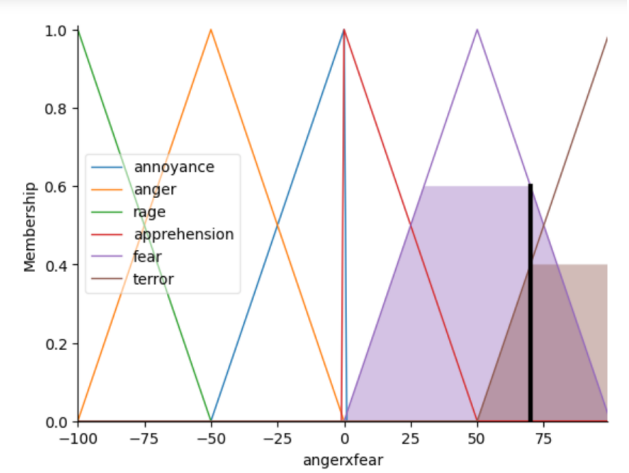

Figura 5.3: Fuzzy Axe Anger-Fear output example

A similar behaviour can be set when discussing the possibilities regarding the three combination dyads. When Baffa's emotional model was explained, there were primary, secundary and tertiary dyads from combined emotions cited, and it was said that the way they were interpreted by the model would drastically change by adding Fuzzy Logic. These dyads can be seen in Tables 5.1, 5.2 and 5.3, and they'll be incorporated as a part of our Fuzzy Sets, and consequently, fuzzy rules as well. All the combinations seen in these tables are solemnly related to medium intensity emotions, hence the main trigger for setting this second part of our fuzzy rules.

Combination	Primary Dyads
Anticipation + Joy	Optimism
Joy + Trust	Love
Trust + Fear	Submission
Fear + Surprise	Awe
Surprise + Sadness	Disapproval
Sadness + Disgust	Remorse
Disgust + Anger	Contempt
Anger + Anticipation	Aggressiveness

Tabela 5.1: Primary dyads and its basic emotion combination

Combination	Secundary Dyads
Anticipation + Trust	Hope
Joy + Fear	Guilt
Trust + Surprise	Curiosity
Fear + Sadness	Despair
Surprise + Disgust	Unbelief
Sadness + Anger	Envy
Disgust + Anticipation	Cynicism
Anger + Joy	Pride

Tabela 5.2: Secundary dyads and its basic emotion combination

As cited before, primary combination dyads are more possible of happening than secundary and tertiary dyads, which will be the main factor for deciding which emotional response should prevail.

5.2.2 Primary and Tertiary Dyads

In order to build a Fuzzy Set that would represent possible combination dyads as efficient and clear as the basic emotions fuzzy set were, instead of looking to Plutchik's axes, relations between each dyad and it's combinations were defined, such as which emotion could be geometrically represented close to another. In a certain way, building for each dyad it's own axes.

For the primary and tertiary dyad combination, it became rather simple. Both of these dyads were successfully divided into a Fuzzy Set only with two main differences regarding it's fuzzy sets, when compared to the basic emotions set: the membership values are referring more to *probability*, rather than intensity. Also, this probability is naturally diminished, since dyads are less likely to happen when compared to basic emotions, which can be seen in Figure 5.4 and Figure 5.5. Notice how the distance between emotions is significantly smaller (each emotion present in the dyads Fuzzy Sets has a

Combination	Tertiary Dyads
Anticipation + Fear	Anxiety
Joy + Surprise	Delight
Trust + Sadness	Sentimentality
Fear + Disgust	Shame
Suprise + Anger	Outrage
Sadness + Anticipation	Pessimism
Disgust + Joy	Morbidness
Anger + Trust	Dominance

Tabela 5.3: Tertiary dyads and its basic emotion combination

probability of 25%).

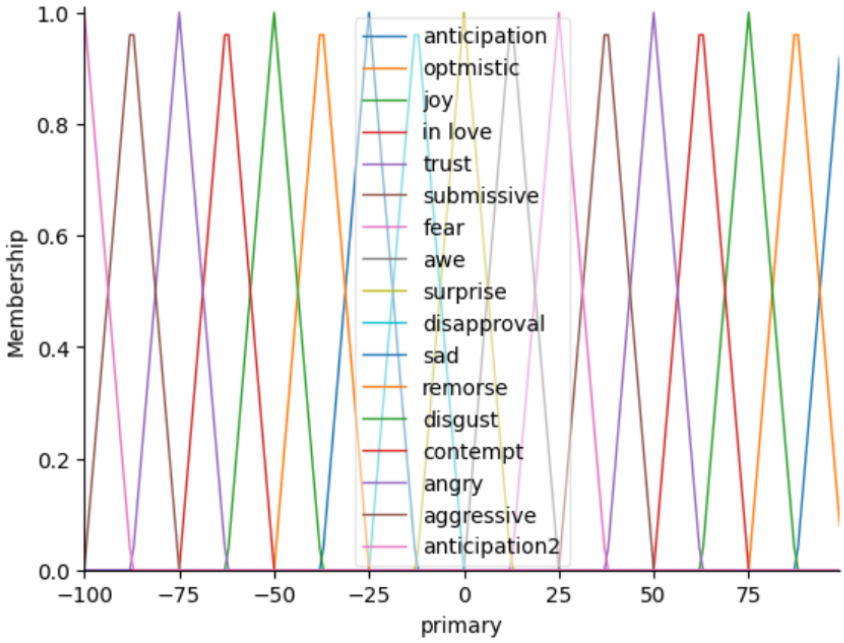

Figura 5.4: Primary Dyad Fuzzy Set

Anticipation and *Sadness* were divided in two to be represented in different areas of the fuzzy set, but represent the same emotional response. When joined, they have the same probability as the other emotions in these sets.

To put it simply, the tertiary and primary dyad became an axe that included not only the basic emotions that compose them but also their possible combinations.

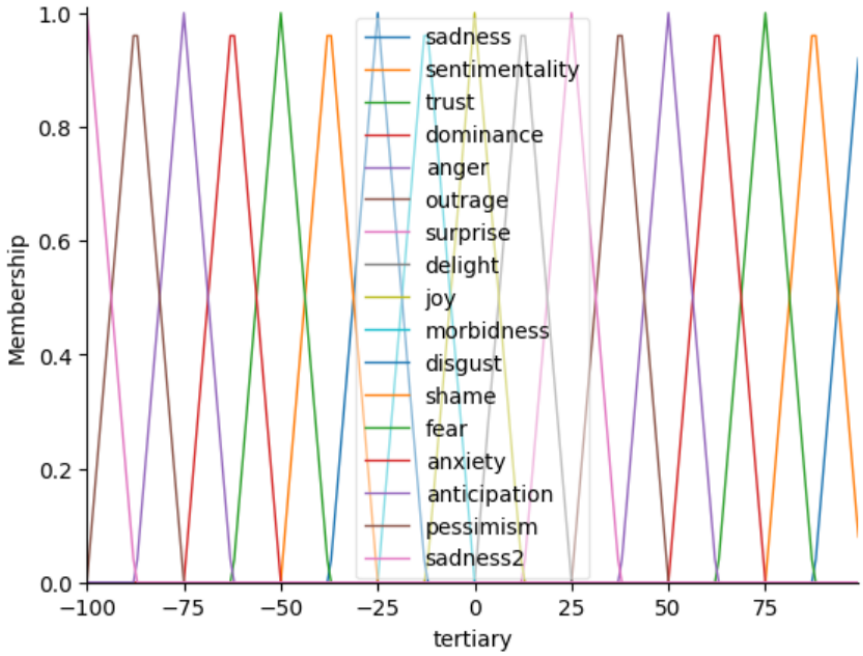

Figura 5.5: Tertiary Dyad Fuzzy Set

5.2.3 Secundary Dyads

The secundary dyad, however, couldn't be summed up to one complete Fuzzy Set, and had to be divided for a functional geometric approach. The first secundary fuzzy set can be seen in Figure 5.6, and represented the possible combination from *Anticipation-Surprise* and *Disgust-Trust* axes. The second part of secundary fuzzy set dyads can be seen in Figure 5.6, and references the combination between axes *Anger-Fear* and *Sadness-Joy*.

The values were also divided to preserve the proportion relating to the other dyads fuzzy sets. Here, *Anger* and *Disgust* were split the same way as previously seen with *Anticipation* and *Sadness*.

5.3 Fuzzy linguistic variables

It has already been established that the model's Fuzzy Sets will be divided by measures of *intensity*. In fuzzy presets, this is defined as our *Fuzzy linguistic variable*.

But for us, *intensity* is a generic *FLV* for defining the model behaviour (see Figure 5.8). In practice, our *FLV's* are going to be all the basic emotions axe in simplified Baffa's Plutchik's wheel, and it's members are the possible emotions in each axe, following the values [-100,-50,0,50,100]:

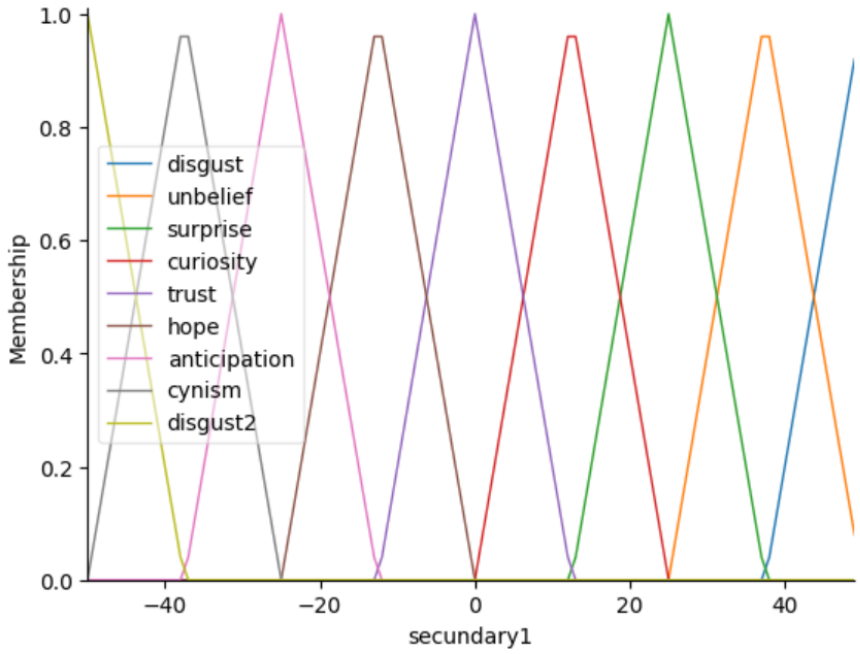

Figura 5.6: *Secundary1* Dyad Fuzzy Set

$$SadnessXJoy = \{Grief, Sadness, Pensiveness, Serenity, Joy, Ecstasy\} \tag{5-1}$$

$$DisgustXTrust = \{Loathing, Disgust, Boredom, Acceptance, Trust, Admiration\} \tag{5-2}$$

$$AngerXFear = \{Rage, Anger, Annoyance, Apprehension, Fear, Terror\} \tag{5-3}$$

$$AnticipationXSurprise = \{Vigilance, Anticipation, Interest, Distraction, Surprise, Amazement\} \tag{5-4}$$

This FLV's are representing the possible basic emotions, but it'll also be considered the possible combinations, regarding the three dyads, as FLV's.

5.4 Fuzzy rules

All this processes are gathered together in order to produce a conclusion, and *fuzzy rules* are the soul for making decisions. *Fuzzy rules* behave just like normal code `if a>= b then c=a`, but with a difference that more than one rule can proceed, just in different levels. The idea is to produce a conclusion that triggers actions or consequences. In this model, we'll work firstly with the following rule format: `IF` *`Intensity`* `AND` *`Basic_Emotion`* `THEN` *`Resultant_Emotion`*.

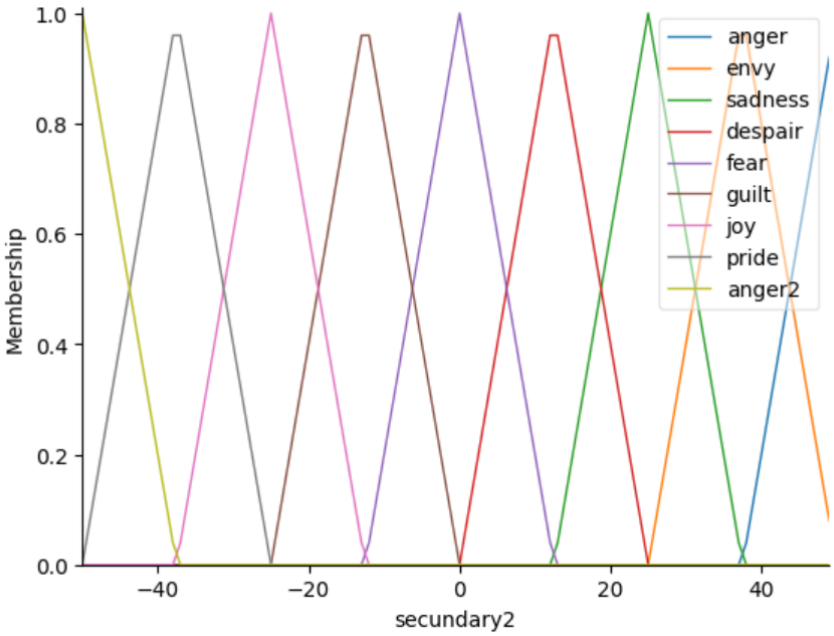

Figura 5.7: *Secundary2* Dyad Fuzzy Set

1. IF *Low* AND *Happy* THEN *Serene*
2. IF *Medium* AND *Happy* THEN *Joyful*
3. IF *High* AND *Happy* THEN *In ecstasy*
4. IF *Low* AND *Sad* THEN *Pensive*
5. IF *Medium* AND *Sad* THEN *Sad*
6. IF *High* AND *Sad* THEN *Grieving*
7. IF *Low* AND *Trusting* THEN *Receptive*
8. IF *Medium* AND *Trusting* THEN *Trusting*
9. IF *High* AND *Trusting* THEN *Admiring*
10. IF *Low* AND *Disgusted* THEN *Bored*
11. IF *Medium* AND *Disgusted* THEN *Disgusted*
12. IF *High* AND *Disgusted* THEN *Loathing*
13. IF *Low* AND *Fearful* THEN *Apprehensive*
14. IF *Medium* AND *Fearful* THEN *Fearful*
15. IF *High* AND *Fearful* THEN *Terrified*

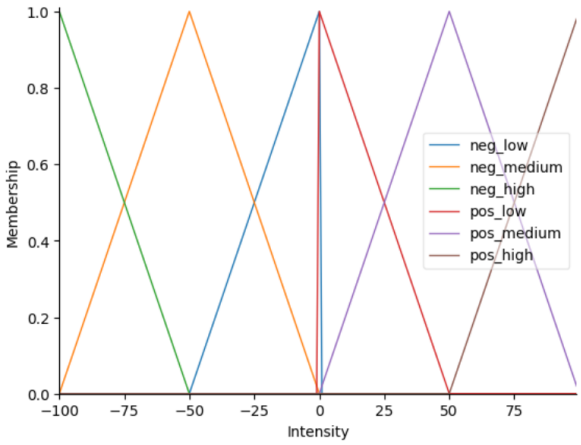

Figura 5.8: Generic *intensity* fuzzy set for basic emotions axe

16. IF *Low* AND *Angry* THEN *Annoyed*
17. IF *Medium* AND *Angry* THEN *Angry*
18. IF *High* AND *Angry* THEN *Rageful*
19. IF *Low* AND *Surprised* THEN *Distracted*
20. IF *Medium* AND *Surprised* THEN *Surprised*
21. IF *High* AND *Surprised* THEN *Amazed*
22. IF *Low* AND *Anticipatory* THEN *Interested*
23. IF *Medium* AND *Anticipatory* THEN *Anticipatory*
24. IF *High* AND *Anticipatory* THEN *Vigilante*

The first set of rules here will check if it's possible to combine this emotions (if they're medium intensity), being `IF` ***`Medium`*** `THEN` ***`Check_Combination`***, and then then check each possible dyad combination.

Primary dyad set of rules:

1. IF *Anticipatory* AND *Joyful* THEN *Optimistic*
2. IF *Joyful* AND *Trusting* THEN *In Love*
3. IF *Trusting* AND *Fearful* THEN *Submissive*
4. IF *Fearful* AND *Surprised* THEN *Awed*
5. IF *Surprised* AND *Sad* THEN *Disapproving*
6. IF *Sad* AND *Disgusted* THEN *Remorseful*

7. IF *Disgusted* AND *Angry* THEN *Contemptuous*
8. IF *Angry* AND *Anticipatory* THEN *Aggressive*

Then, secundary dyad rules:

1. IF *Anticipatory* AND *Trusting* THEN *Hopeful*
2. IF *Joyful* AND *Fearful* THEN *Guilty*
3. IF *Trusting* AND *Surprised* THEN *Curious*
4. IF *Fearful* AND *Sad* THEN *Despairing*
5. IF *Surprised* AND *Disgusted* THEN *Unbelieving*
6. IF *Sad* AND *Angry* THEN *Envious*
7. IF *Disgusted* AND *Anticipatory* THEN *Cynical*
8. IF *Angry* AND *Joyful* THEN *Prideful*

And finally, tertiary dyad rules:

1. IF *Anticipatory* AND *Fearful* THEN *Anxious*
2. IF *Joyful* AND *Surprised* THEN *Delightful*
3. IF *Trusting* AND *Sad* THEN *Sentimental*
4. IF *Fearful* AND *Disgusted* THEN *Shameful*
5. IF *Surprised* AND *Angry* THEN *Outraged*
6. IF *Sad* AND *Anticipatory* THEN *Pessimist*
7. IF *Disgusted* AND *Joyful* THEN *Morbid*
8. IF *Angry* AND *Trusting* THEN *Dominant*

The rules regarding intensity and basic Plutchik's emotions will not be directly programmed inside the Fuzzy Logic, they're presented here for explaining purposes. However, the fuzzy rules defined by the dyads are explicitly programmed inside the code.

6
Game Mechanics

In the following chapter, it'll be presented all the mechanic, gameplay, and visual details added in the final game product, such as how the NPC's receives determined actions, and how the UI works in the game, as well as chosen online assets. Originally, the idea was to take Bicalho's 2D first game version (Figure 6.1) and transforming it into a 3D project, alongside a VR experience.

In the *Main Menu*(Figure 6.2) screen, which also introduces the storytelling for the player (subsection *Storytelling*), the player chooses which experience he would like to try.

Figura 6.1: First version of *Future Falls*

6.1
The Virtual World

For the visuals of this work, the asset package *PolygonSciFiCity* was used[1], which included a demo scene with a pre-built city, that, after a few enhances and modifications, served as a skeleton for building both 3D and VR scenes (Figure 6.3).

[1]Low Poly packages can be easily found in Unity Asset sites, sometimes for free, other times for normally small fees.

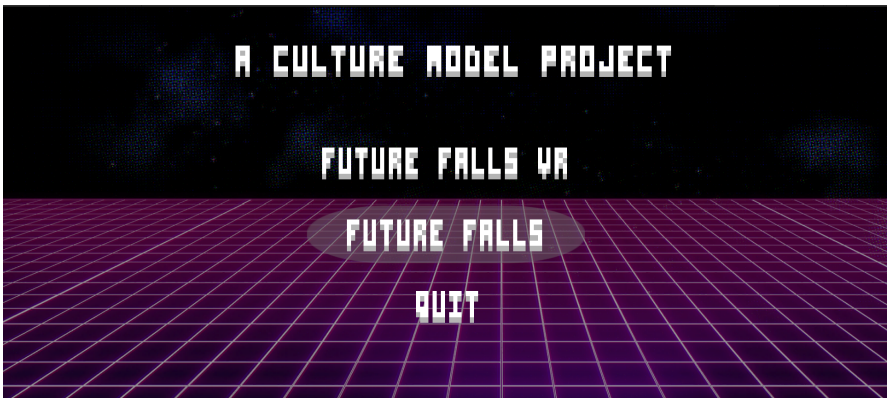

Figura 6.2: Main Menu screen

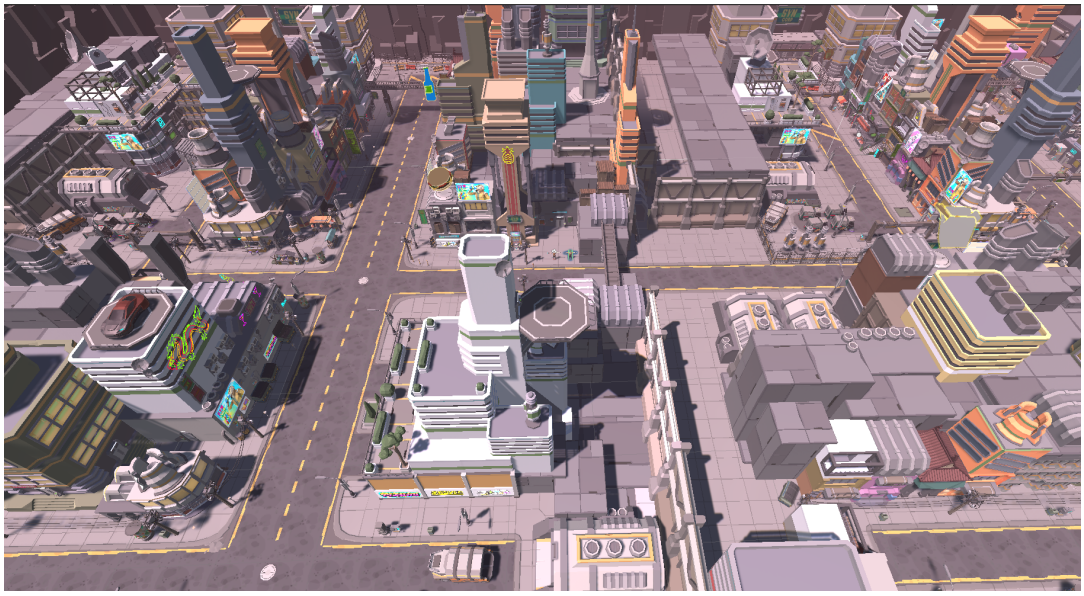

Figura 6.3: City game map, assets from package *PolygonSciFiCity*

6.2 3D Interface and Controls

Being directly based upon the first version interface((Figure 6.4)), the 3D interface (Figure 6.5) is divided in 4 main parts: The aim, centered in the screen, that will shoot the player when clicked. The NPC HUD, that shows all the important information regarding the NPC, such as his current mental state, his trust and health level, his culture and name. Secondly, the player's health level, together with an informing text that indicates when interactions with an NPC are possible, and to activate them by pressing `i` key. Both the NPC HUD an finally, when these interactions are activated by the player, the Action HUD appears in the interface(Figure 6.6), allowing a few clickable buttons: *Give*, *Steal*, *Talk* and *Exit*.

The first three options will open up another hud. *Give* and *Steal* opens up the same secondary menu, since it's possible to make these two actions

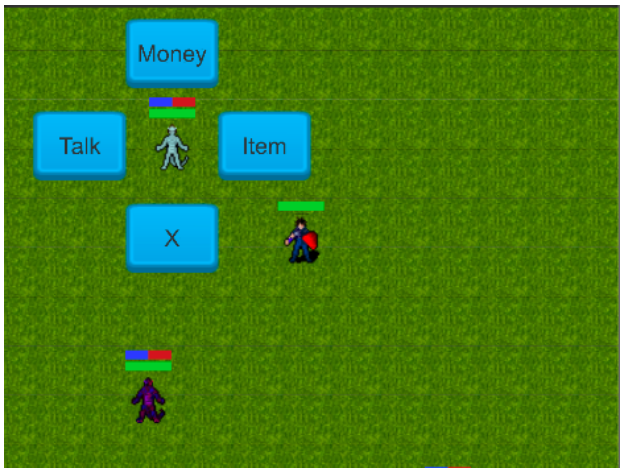

Figura 6.4: Future Falls previous version, 2D Action menu

Figura 6.5: Future Falls 3D Interface

regarding an *Item* or *Money*. So, the flow is decided by first selecting what you wish to do, and then selecting to which belonging it'll be done in these cases. The *Talk* button opens a similar HUD, but with *Be Polite* and *Be Rude* options, to select how you wish to talk to that NPC (Figure 6.7). When the Action HUD is selected by the player, gameplay time will freeze so that the mouse pointer (that acts like the aiming system inside the game) can roam through the screen freely and click the buttons. The *Exit* button will unfreeze the screen and deactivate this interactions options.

Movement is controlled by the *WASD*[2] logic movement. It also allows movement to be controlled by a conected controller. The game camera follows the mouse pointer, that's stuck on the aim sprite located in the center of the

[2] *Four keyboard keys that are used to interact with video games in lieu of Arrow keys or a controller. W and S control forward and backward movement, while A and D are left and right. Q, E, F, X, Shift and Spacebar are also used in various games.*[11]

Figura 6.6: Future Falls 3D Interface, with ActionHUD activated

Figura 6.7: Action HUD all paths and possibilities

screen.

6.3 VR Interface and Controls

VR interactions are a bit more sophisticated, since it isn't possible to enable a HUD that's innate to the computer screen. But programming voice commands for dialogue interactions was a little apart from this work goals, and far from being priority. Hence that, the solution found was adding a bracelet (Figure 6.8) that would present the player all important information from NPC HUD, his heatlh level, *Talk* options and a *Pause* button. The aiming system, originally attached to the mouse pointer, was replaced by a clickable raytracing canvas inside the NPC's game object(Figure 6.9), that activates when pointing at him, works the same as pressing `i` in the 3D version.

Give and *Steal* options were upgraded to physical commands with objects that are in two pockets, hosted in the NPC prefab. One pocket, that holds a weapon, triggers the *Stealing Item* when taken and *Giving Item* when an object is placed in that pocket, located on the side of the NPC. And the back pocket, that contains a jar (Figure 6.10), respecting the same behaviours cited for the side pocket, triggers them related to *Money*.

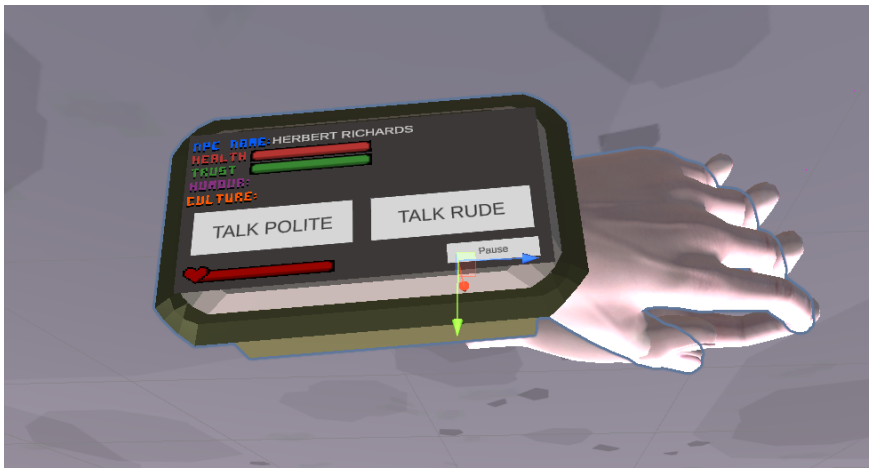

Figura 6.8: Interface nested inside the VR bracelet

Figura 6.9: Raycast triggered by pointing at the NPC

6.4 Animations

The most anticipated feature for this 3D experience was the possibility of animated responses that would represent the NPC emotional reaction. There were a few possible animations inherited from Bicalho's project that could be easily connected to the emotional responses. The final relation is presented in Table 6.1, but other animations were added for better game immersion, like *walking* animation and *neutral* animation.

6.5 Storytelling

In the beginning of *Main Menu*, there are 4 pop-ups that introduces the context and objective of the game for the player. These pop-ups are displayed in the order shown here, and can be seen in Figures 6.11, 6.12, 6.13 and 6.14.

Figura 6.10: Jar grabbed from NPC's back pocket

Mental State	NPC Animation
Fear	Terrified Animation
Anger	Yelling Animation
Trust	Thankful Animation
Disgust	Loser Animation
Joy	Happy Animation
Sadness	Sad Idle Animation
Surprise	Surprised Animation
Anticipation	Excited Animation

Tabela 6.1: Mental state and NPC's animation response

Figura 6.11: Introduction part 1

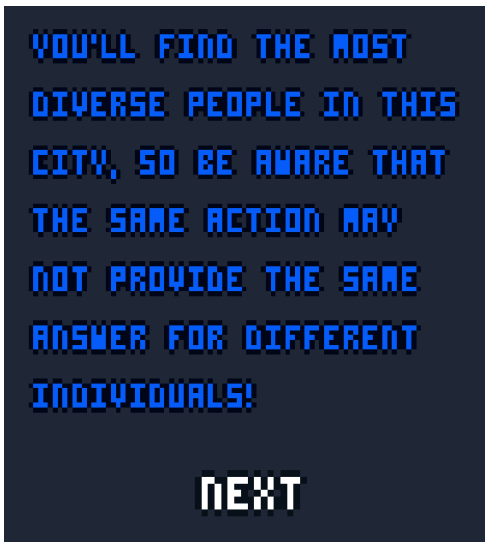

Figura 6.12: Introduction part 2

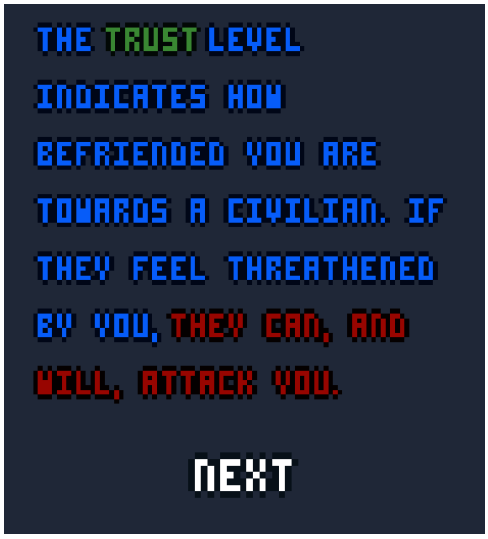

Figura 6.13: Introduction part 4

Figura 6.14: Introduction part 4

7
System Modeling, Implementation and Tools

In this chapter, finally details of the code, programming and organization will be explored, as well as a brief walktrough concerning the tools that were used, why they were chosen, advantages and disadvantages and also how they were incorporated in this work. But Firstly, it'll be presented the models in this game, sectioned by each game scene.

7.1
System Modeling

The first scene presented in the game is the Main Menu scene, which allows the player to decide which experience he wishes to play. In Figure 7.1 we can see this relation in a simplified model. Next, it is presented the model for the Flask API, where the Fuzzy Logic is being processed (Figure 7.2). Then, in Figure 7.3, a UML model for the 3D game version is seen, with the explicit relations between classes: Classes named by their own characteristics like Personality and Culture hold the dictionaries responsible for each characteristics definition, like the cultural factors and OCEAN.

Emotion, however, is responsible not for holding the dictionary for all emotions, but for making it's respective calculations like the Current Emotion, New Emotion and Event Emotion, whic is why it's the class that calls the Fuzzy API. The dictionary for all emotions and it's relative actions are in AllEmotions.cs and ActionEmotion.cs. All the processing upon the receiving actions are made in the NPC class. The ActionHUD class takes care of the visual and possible input actions. The model for the VR game version is seen in Figure 7.4 and works the same way as the 3D model, differing only in the action input.

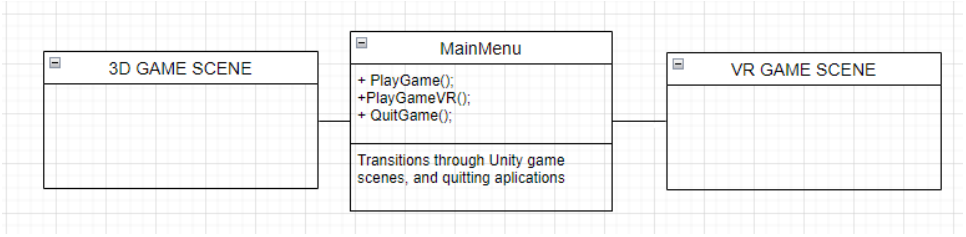

Figura 7.1: Main Menu model

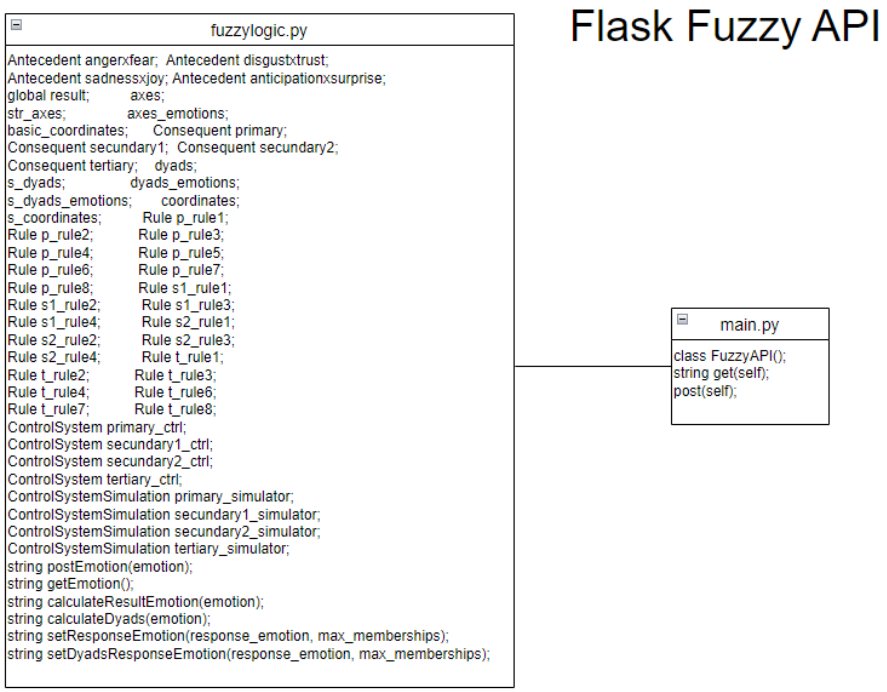

Figura 7.2: Flask python API modelling

7.2 Tools

For this work to be born, a few difficult choices were taken account regarding the code structure and details. One of the major challenges was finding out how the Fuzzy Logic would be incorporated inside the game. There was never a doubt upon using the *Unity* engine to build the game, so the main focus was to find Fuzzy Logic libraries in C#, or a way to insert python codes inside Unity (since libraries for Fuzzy Logic in python wasn't really hard to find). Some unnoficial add-ons were found, but they were unstable and not very practical for an efficient exchange of data between the game and the fuzzy logic. Finally, it was decided upon making the Fuzzy Logic using python's library *Scikit-fuzzy*, and incorporating it in the game through a local host using *Flask* for API calls, which damaged the game performance but worked perfectly for processing emotions. In the following subsections, this tools will be further explained.

7.2.1 Unity

Unity[22] game engine has some advantages that made the decision for it to be used in this work, not a hard one. Firstly, it's content online is massive,

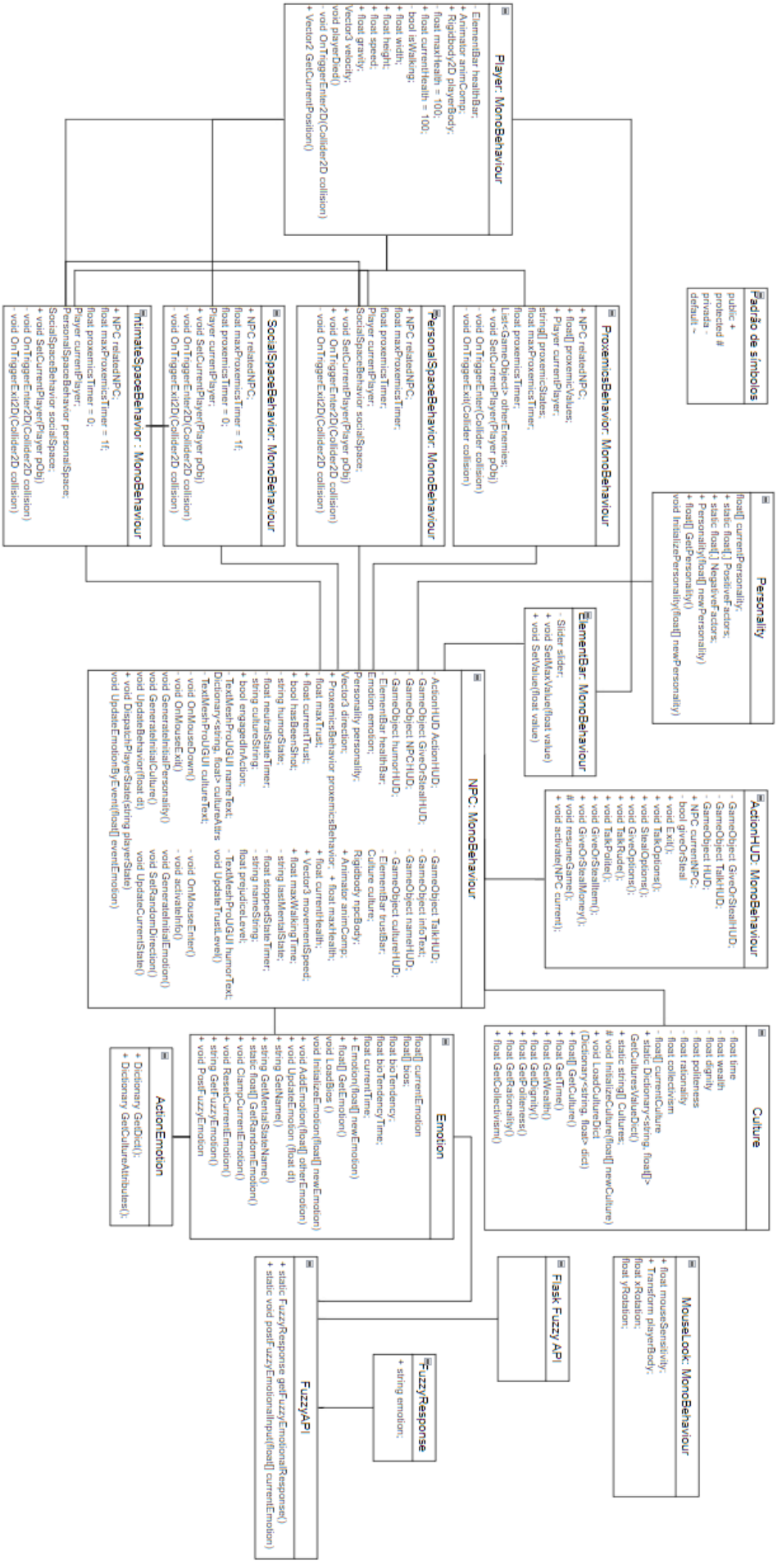

Figura 7.3: 3D Version Project Modelling

Figura 7.4: VR Version Project Modelling

there's a large community and free documentation and tutorials, videos, forums and assets which transforms most of Unity Projects in a task force initiative. Besides that, it's also a rather easy-to-learn framework, with an intuitive UI. However, it's language is C#, which normally is a good factor, but in this specific work, since there was the need to incorporate a complex Fuzzy Logic to the game, it became more of an obstacle due to the fact that Unity is a bit stiff when it comes to working with other programming languages. Before deciding on the Flask approach, the Python plugin for Unity was extremely limited and would hardly allow a more complex processing like the one we need here.

7.2.2 Flask

Flask[16] is a tool made for building API calls in a simplified way. In this work, a localhost build was made to allow exchanges between Unity and the Scikit-Fuzzy logic. The *current emotion* would be sent through POST methods for the API, that returns the string name of the resultant emotion through the GET method (Figure 7.5). The API coding itself wasn't hard, just needed some attention in the beginning, but it's a simple code (Figure 7.6)[1].

```
public static class FuzzyAPI
{

    1 referência
    public static FuzzyResponse getFuzzyEmotionalResponse()
    {
        HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://127.0.0.1:5000/fuzzyemotionapi");
        HttpWebResponse response = (HttpWebResponse)request.GetResponse();
        StreamReader reader = new StreamReader(response.GetResponseStream());
        string json = reader.ReadToEnd();
        return JsonUtility.FromJson<FuzzyResponse>(json);

    }

    1 referência
    public static void postFuzzyEmotionalInput(float[] currentEmotion)
    {
        string[] emotionKeys = { "axeAF", "axeDT", "axeSJ", "axeAS" };
        HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://127.0.0.1:5000/fuzzyemotionapi");
        var postData = "&" + emotionKeys[0] + "=" + Uri.EscapeDataString(currentEmotion[0].ToString());
        for (var i=1 ; i<4 ; i++)
        {
            postData += "&"+ emotionKeys[i] + "=" + Uri.EscapeDataString(currentEmotion[i].ToString());
```

Figura 7.5: API calls inside Unity

7.2.3 Scikit-fuzzy

"*Scikit-fuzzy* is a collection of fuzzy logic algorithms intended for use in the SciPy Stack, written in the Python computing language."[21]. It allowed the Fuzzy Logic in this projject to be simplified in a maximum, and at the same time exploring it's possibilities without any loss. Utilizing it's subpackage

[1]RESTful is a term used for a type of architecture used in Web API's [9]

```python
app = Flask(__name__)
api = Api(app)
...
class FuzzyAPI(Resource):
    def get(self):
        emo = getEmotion();
        return {"emotion" : emo}

    def post(self):
        emo = []
        emo.append(request.form['axeAF'])
        emo.append(request.form['axeDT'])
        emo.append(request.form['axeSJ'])
        emo.append(request.form['axeAS'])
        for i in range(4):
            emo[i] = emo[i].replace(",",".");
            emo[i] = float(emo[i]);
        postEmotion(emo);
        return;
api.add_resource(FuzzyAPI, "/fuzzyemotionapi")
if __name__ == "__main__":
    app.run(debug=True)
```

Figura 7.6: Main.py, enabling RESTful requests with Flask.

control, every detail of the Fuzzy Logic (explain in this previous section) was able to be easily defined, as shown below:

1. `angerxfear = ctrl.Antecedent(np.arange(-100,100,1),'angerxfear')`
2. `axes[i][axes_emotions[i][j]] = fuzz.trimf(axes[i].universe, basic_coordinates[j])`
3. `primary = ctrl.Consequent(np.arange(-100,100,1),'primary')`
4. `p_rule1 = ctrl.Rule(anticipationxsurprise['anticipation'] sadnessxjoy['joy'], primary['optmistic'])`
5. `primary_ctrl = ctrl.ControlSystem([p_rule1,p_rule2, ...,p_rule8])`
6. `primary_simulator = ctrl.ControlSystemSimulation(primary_ctrl)`
7. `fuzz.interp_membership(axes[i].universe, axes[i][axes_emotions[i][j]].mf, emotion[i]*100)`
8. `primary_simulator.compute()`

These are examples of the main functions used in scikit-fuzzy for this work, which sums up most of the Fuzzy Logic code.

7.2.4 Hurricane VR

"HurricaneVR is a complete VR Interaction Framework with a heavy emphasis on quality physics interactions. The Physics Hands are controlled by tuned PD Controllers enabling smooth and responsive hand physics, high quality collision, two handed holding, throwing, and interactions with your physics objects."[8]. *Hurricane* is straight up what made the VR version possible, it facilitates and presents so much examples and presets, assets and scenes that it makes programming a VR experience an easy task. However, on a total opposite from Unity, it's very hard to find discussions and material online from users, very much likely due to the fact that it's a slight more expensive paid asset.

7.3 Implementation

These next presented subsections are going to explain the most important details of the code.

7.3.1 Personality

Personality factors are being randomly generated inside the NPC mono-behaviour, and are represented by an array of floats, each one representing one of each O.C.E.A.N personality factors[12]. Inside the personality file, it's also two structures that save the influence each OCEAN factor has in each basic emotion, as presented in Table 4.4.

```
9 referências
public class Personality {

    // These four positions represent the
    // 5 attributes of OCEAN's Personality Model
    // [0] Openness [Range(-1f, 1f)]
    // [1] Conscientiousness [Range(-1f, 1f)]
    // [2] Extraversion [Range(-1f, 1f)]
    // [3] Agreeableness [Range(-1f, 1f)]
    // [4] Neuroticism [Range(-1f, 1f)]
    float[] currentPersonality = new float[5];
```

Figura 7.7: Personality represented as an array of float[5]

7.3.2 Actions

Each action is defined according to it's position in the code, but all of them trigger the same function inside of the NPC. This function is seen in

Figure 7.9, that calculates the event emotion. Below, in Figure 7.8, the example code for talking politely trigger.

```
public void TalkPolite()
{

    currentNPC.DispatchPlayerState("is_talking_politely");
    TalkHUD.SetActive(false);
    resumeGame();
}
```

Figura 7.8: NPC is talked to, nicely. Code from 3D version

7.3.3 Emotion

In Figure 7.9, it's shown how the *Event Emotion* is calculated, and in Figure 7.10, the *New Emotion* before it is clamped and added to the *Current Emotion*.

```
Dictionary<string, string[]> stateEmo = ActionEmotions.GetDict();
Dictionary<string, string> stateAttrs = ActionEmotions.GetCultureAttributes();
Dictionary<string, float[]> allEmo = AllEmotions.GetDict();

if (playerState != "")
{
    string[] emotionsArray = stateEmo[playerState];
    string attrName = stateAttrs[playerState];
    float rat = 1 - culture.GetRationality();
    float attrValue = cultureAttrs[attrName];
    float result = Mathf.Sqrt(attrValue * rat);
    string resEmotion = emotionsArray[0];

    for (int i = 1; i < emotionBands.Length; i++)
    {
        if (result > emotionBands[i])
        {
            resEmotion = emotionsArray[i];
        }
    }

    UpdateEmotionByEvent(allEmo[resEmotion]);
    UpdateTrustLevel();
    UpdateCurrentState();
```

Figura 7.9: Event Emotion calculation

7.3.4 Culture

The use of cultural attributes for affecting emotions is described in the subsection *Emotion*. Its factor values are randomly chosen for each NPC every time the game starts, using the predefined cultures from Table 7.1.

```
void UpdateEmotionByEvent(float[] eventEmotion)
{
    float[] newEmotion = new float[4];
    float[] p = personality.GetPersonality();
    float[,] pFactors = Personality.PositiveFactors;
    float[,] nFactors = Personality.NegativeFactors;

    // Generate new emotion based on Personality Traits and Factors
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (eventEmotion[i] > 0)
                newEmotion[i] += eventEmotion[i] * p[j] * pFactors[j, i];
            else
                newEmotion[i] += eventEmotion[i] * p[j] * nFactors[j, i];
        }
        newEmotion[i] = newEmotion[i] / 5;
    }

    //// Add new generated emotion
    emotion.AddEmotion(newEmotion);
    //emotion.calculateEmotion();
    //// Clamp after emotion add
    emotion.ClampCurrentEmotion();
```

Figura 7.10: New Emotion calculation

Culture	time	wealth	dignity	politeness	rationality	collectivism
Ogre	0.15	0.63	0.31	0.02	0.15	0.93
Traveller	0.28	0.25	0.82	0.75	0.39	0.43
Ranger	0.72	0.56	0.88	0.73	0.76	1
Adventurer	0.83	0.24	0.65	0.58	0.25	0.77
Downside	0.05	1	0.02	0.15	0.93	0.22

Tabela 7.1: Predefined cultures built in the game

7.3.5 Fuzzy Logic AI

A big part of the code in this Fuzzy Logic document is reserved for definitions of the logic parameters, as in the basic emotion axes were defined as the antecedents for the simulation, and the dyads defined as the consequents, and then, defining their Fuzzy Sets (Figures 7.11 and 7.12).

Then, the Fuzzy rules are defined, and as mentioned before, only the rules for the dyads were explicitely programmed (Figure 7.13). The previous rules for our model are being interpreted according to the membership functions, processed in the function seen in Figure 7.14. For this processing to begin, the clamped *current emotion* received by the POST method is multiplied by 100 to match the membership values.

These values are inserted on the membership functions of basic axes of Plutchik's wheel, then they're compared, and tested out to see if there are more than one emotion in medium intensity that's same valued in the maximum of the memberships values (Figure 7.15). If there are, then the crisp values of

```
angerxfear = ctrl.Antecedent(np.arange(-100,100,1),'angerxfear')
disgustxtrust = ctrl.Antecedent(np.arange(-100,100,1),'disgustxtrust')
sadnessxjoy = ctrl.Antecedent(np.arange(-100,100,1),'sadnessxjoy')
anticipationxsurprise = ctrl.Antecedent(np.arange(-100,100,1),'anticipationxsurprise')

# Creating membership functions)

axes = [angerxfear,disgustxtrust,sadnessxjoy,anticipationxsurprise]
str_axes = ['angerxfear','disgustxtrust','sadnessxjoy','anticipationxsurprise']
axes_emotions = [['annoyance','anger','rage','apprehension','fear','terror'],
                 ['boredom','disgust','loathing','acceptance','trust','admiration'],
                 ['pensiveness','sadness','grief','serenity','joy','ecstasy'],
                 ['interest','anticipation','vigilance','distraction','surprise','amazement']]
basic_coordinates = [[-50,0,0],[-100,-50,0],[-100,-100,-50], [0,0,50], [0,50,100], [50,100,100]]

#Plutchik's axes membership functions
for i in range(len(axes)):
    for j in range(len(axes_emotions[0])):
        axes[i][axes_emotions[i][j]] = fuzz.trimf(axes[i].universe,basic_coordinates[j])
```

Figura 7.11: All the simplified Plutchik's axes definition inside the model

```
#Creating fuzzy response sets for dyads
primary = ctrl.Consequent(np.arange(-100,100,1),'primary')
secundary1 = ctrl.Consequent(np.arange(-50,50,1),'secundary1')
secundary2 = ctrl.Consequent(np.arange(-50,50,1),'secundary2')
tertiary = ctrl.Consequent(np.arange(-100,100,1),'tertiary')

dyads = [primary,tertiary]
s_dyads = [secundary1,secundary2]      You, 3 days ago • fuzzy logic incorporated


dyads_emotions = [['anticipation','optmistic','joy','in love','trust','submissive','fear','awe','surprise',
                   'disapproval','sad','remorse','disgust','contempt','angry','aggressive','anticipation2'],
                  ['sadness','sentimentality','trust','dominance','anger','outrage','surprise','delight','joy',
                   'morbidness','disgust','shame','fear','anxiety','anticipation','pessimism','sadness2']]

s_dyads_emotions = [['disgust','unbelief','surprise','curiosity','trust','hope','anticipation','cynism','disgust2'],
                    ['anger','envy','sadness','despair','fear','guilt','joy','pride','anger2']]



coordinates = [[87.5,100,100],[75,87.5,100],[62.5,75,87.5],[50,62.5,75],[37.5,50,62.5],[25,37.5,50],[12.5,25,37.5],
               [0,12.5,25],[-12.5,0,12.5],[-25,-12.5,0],[-37.5,-25,-12.5],[-50,-37.5,-25],[-62.5,-50,-37.5],
               [-75,-62.5,-50],[-87.5,-75,-62.5],[-100,-87.5,-75],[-100,-100,-87.5]]

s_coordinates = [[37.5,50,50],[25,37.5,50],[12.5,25,37.5],[0,12.5,25],[-12.5,0,12.5],
                 [-25,-12.5,0],[-37.5,-25,-12.5],[-50,-37.5,-25],[-50,-50,-37.5]]


#Dyads membership functions
for i in range(len(dyads)):
    for j in range(len(dyads_emotions[0])):
        dyads[i][dyads_emotions[i][j]] = fuzz.trimf(dyads[i].universe,coordinates[j])

for i in range(len(s_dyads)):
    for j in range(len(s_dyads_emotions[0])):
        s_dyads[i][s_dyads_emotions[i][j]] = fuzz.trimf(s_dyads[i].universe,s_coordinates[j])
```

Figura 7.12: All the dyads definition inside the model

Current Emotion are sent to be computed by the dyads system.

In the dyads simulation, the *Current Emotion* values are inserted on the basic axes, for it to be processed by the control system(Figure 7.17). Then, the same process for comparing membership functions is made, but this time the criteria for tiebreaking is the probability of each dyad (Primary, Secondary, and Tertiary)(Figure 7.18).

```
# primary dyads
p_rule1 = ctrl.Rule(anticipationxsurprise['anticipation'] & sadnessxjoy['joy'], primary['optmistic'])
p_rule2 = ctrl.Rule(sadnessxjoy['joy'] & disgustxtrust['trust'], primary['in love'])
p_rule3 = ctrl.Rule(disgustxtrust['trust'] & angerxfear['fear'], primary['submissive'])
p_rule4 = ctrl.Rule(angerxfear['fear'] & anticipationxsurprise['surprise'], primary['awe'])
p_rule5 = ctrl.Rule(anticipationxsurprise['surprise'] & sadnessxjoy['sadness'], primary['disapproval'])
p_rule6 = ctrl.Rule(sadnessxjoy['sadness'] & disgustxtrust['disgust'], primary['remorse'])
p_rule7 = ctrl.Rule(disgustxtrust['disgust'] & angerxfear['anger'], primary['contempt'])
p_rule8 = ctrl.Rule(angerxfear['anger'] & anticipationxsurprise['anticipation'], primary['aggressive'])

# secundary 1 dyads
s1_rule1 = ctrl.Rule(anticipationxsurprise['anticipation'] & disgustxtrust['trust'], secundary1['hope'])
s1_rule2 = ctrl.Rule(disgustxtrust['disgust'] & anticipationxsurprise['anticipation'], secundary1['cynism'])
s1_rule3 = ctrl.Rule(disgustxtrust['trust'] & anticipationxsurprise['surprise'], secundary1['curiosity'])
s1_rule4 = ctrl.Rule(anticipationxsurprise['surprise'] & disgustxtrust['disgust'], secundary1['unbelief'])

# secundary 2 dyads
s2_rule1 = ctrl.Rule(sadnessxjoy['sadness'] & angerxfear['anger'], secundary2['envy'])
s2_rule2 = ctrl.Rule(angerxfear['fear'] & sadnessxjoy['sadness'], secundary2['despair'])
s2_rule3 = ctrl.Rule(sadnessxjoy['joy'] & angerxfear['fear'], secundary2['guilt'])
s2_rule4 = ctrl.Rule(angerxfear['anger'] & sadnessxjoy['joy'], secundary2['pride'])

# tertiary dyads
t_rule1 = ctrl.Rule(anticipationxsurprise['anticipation'] & angerxfear['fear'], tertiary['anxiety'])
t_rule2 = ctrl.Rule(sadnessxjoy['joy'] & anticipationxsurprise['surprise'], tertiary['delight'])
t_rule3 = ctrl.Rule(disgustxtrust['trust'] & sadnessxjoy['sadness'], tertiary['sentimentality'])
t_rule4 = ctrl.Rule(angerxfear['fear'] & disgustxtrust['disgust'], tertiary['shame'])
t_rule5 = ctrl.Rule(anticipationxsurprise['surprise'] & angerxfear['anger'], tertiary['outrage'])
t_rule6 = ctrl.Rule(sadnessxjoy['sadness'] & anticipationxsurprise['anticipation'], tertiary['pessimism'])
t_rule7 = ctrl.Rule(disgustxtrust['disgust'] & sadnessxjoy['joy'], tertiary['morbidness'])
t_rule8 = ctrl.Rule(angerxfear['anger'] & disgustxtrust['trust'], tertiary['dominance'])
```

Figura 7.13: Fuzzy rules definitions to be added on the control systems

7.3.6 Animations

The animations are programmed to respond upon the newly calculated NPC's mental state, defined in the Animator Unity tool (utilized to organize transitions and animation triggers), that can be seen in Figure 7.19. These transitions are defined inside the NPC class, varying according to the humourState received, and triggering the animComp variable.

```
def calculateResultEmotion(emotion):
    for i in range(len(emotion)):
        emotion[i] = emotion[i]*100 #adjust emotion axes values as percentages for membership functions

    response_emotion = []
    membership_values = []
    max_values = []
    response_emotion.clear()
    membership_values.clear()
    max_values.clear()
    for i in range(len(axes)):
        membership_values.append([])
        for j in range(len(axes_emotions[0])):
            membership_values[i].append(fuzz.interp_membership(axes[i].universe, axes[i][axes_emotions[i][j]].mf, emotion[i]*100))
```

Figura 7.14: Calculating the basic axes membership functions

```
cont_medium = 0
for i in range(len(membership_values)):
    response_emotion.append(membership_values[i].index(max(membership_values[i])))
    max_values.append(max(membership_values[i]))
    if(response_emotion[i] == 1 or response_emotion[i] == 4):
        cont_medium += 1
if(cont_medium <= 1):
   result = setResponseEmotion(response_emotion, max_values)
   return result;
else:
    result = calculateDyads(emotion)
    return result;
```

Figura 7.15: Comparing the maximum on basic axes membership functions

```
# Fuzzy Sistems and Simulations
primary_ctrl = ctrl.ControlSystem([p_rule1,p_rule2,p_rule3,p_rule4,p_rule5,p_rule6,p_rule7,p_rule8])
secundary1_ctrl = ctrl.ControlSystem([s1_rule1,s1_rule2,s1_rule3,s1_rule4])
secundary2_ctrl = ctrl.ControlSystem([s2_rule1,s2_rule2,s2_rule3,s2_rule4])
tertiary_ctrl = ctrl.ControlSystem([t_rule1,t_rule2,t_rule3,t_rule4,t_rule5,t_rule6,t_rule7,t_rule8])

primary_simulator = ctrl.ControlSystemSimulation(primary_ctrl)
secundary1_simulator = ctrl.ControlSystemSimulation(secundary1_ctrl)
secundary2_simulator = ctrl.ControlSystemSimulation(secundary2_ctrl)
tertiary_simulator = ctrl.ControlSystemSimulation(tertiary_ctrl)
```

Figura 7.16: Fuzzy simulations definitions

```
def calculateDyads(emotion):
    for i in range(len(emotion)):
        primary_simulator.input[str_axes[i]] = emotion[i]
        tertiary_simulator.input[str_axes[i]] = emotion[i]
        if(i==0 or i==2):
            secundary2_simulator.input[str_axes[i]] = emotion[i]
        elif(i==1 or i==3):
            secundary1_simulator.input[str_axes[i]] = emotion[i]



    primary_simulator.compute()
    secundary1_simulator.compute()
    secundary2_simulator.compute()
    tertiary_simulator.compute()


    outputs = [primary_simulator.output['primary'],tertiary_simulator.output['tertiary']]
    s_outputs = [secundary1_simulator.output['secundary1'],secundary2_simulator.output['secundary2']]

    response_emotion = []
    membership_values = []
    response_emotion.clear()
    membership_values.clear()

    mvalues_size = 0
    for i in range(len(dyads)):
        membership_values.append([])
        mvalues_size += 1
        for j in range(len(dyads_emotions[0])):
            membership_values[i].append(fuzz.interp_membership(dyads[i].universe, dyads[i][dyads_emotions[i][j]].mf, outputs[i]))


    for i in range(len(s_dyads)):
        membership_values.append([])
        for j in range(len(s_dyads_emotions[0])):
            membership_values[i + mvalues_size].append(fuzz.interp_membership(s_dyads[i].universe,
                                                       s_dyads[i][s_dyads_emotions[i][j]].mf, s_outputs[i]))
```

Figura 7.17: Dyads simulations

```python
def setDyadsResponseEmotion(response_emotion, max_memberships):
    final_index = 0;
    max_index = []
    max_index.clear()
    max_ = max(max_memberships)

    for i in range(len(max_memberships)):
            if max_memberships[i] == max_:
                max_index.append(i);

    if(len(max_index) == 1):
        if max_index[0] == 0 or max_index[0] == 1:
            result = str(dyads_emotions[max_index[0]][response_emotion[max_index[0]]])
            return result.capitalize()
        else:
            result = str(s_dyads_emotions[max_index[0]][response_emotion[max_index[0]]])
            return result.capitalize()
    else: #ordem de frequencia das dyads
        for i in range(len(max_index)):
            if max_index[i] == 0: #primary
                result = str(dyads_emotions[max_index[i]][response_emotion[max_index[i]]])
                return result.capitalize()
            elif max_index[i] == 2 or max_index[i] == 3: #secundary
                result = str(s_dyads_emotions[max_index[i]][response_emotion[max_index[i]]])
                return result.capitalize()
            else: #tertiary
                result = str(dyads_emotions[max_index[i]][response_emotion[max_index[i]]])
                return result.capitalize()
```

Figura 7.18: Dyads simulations

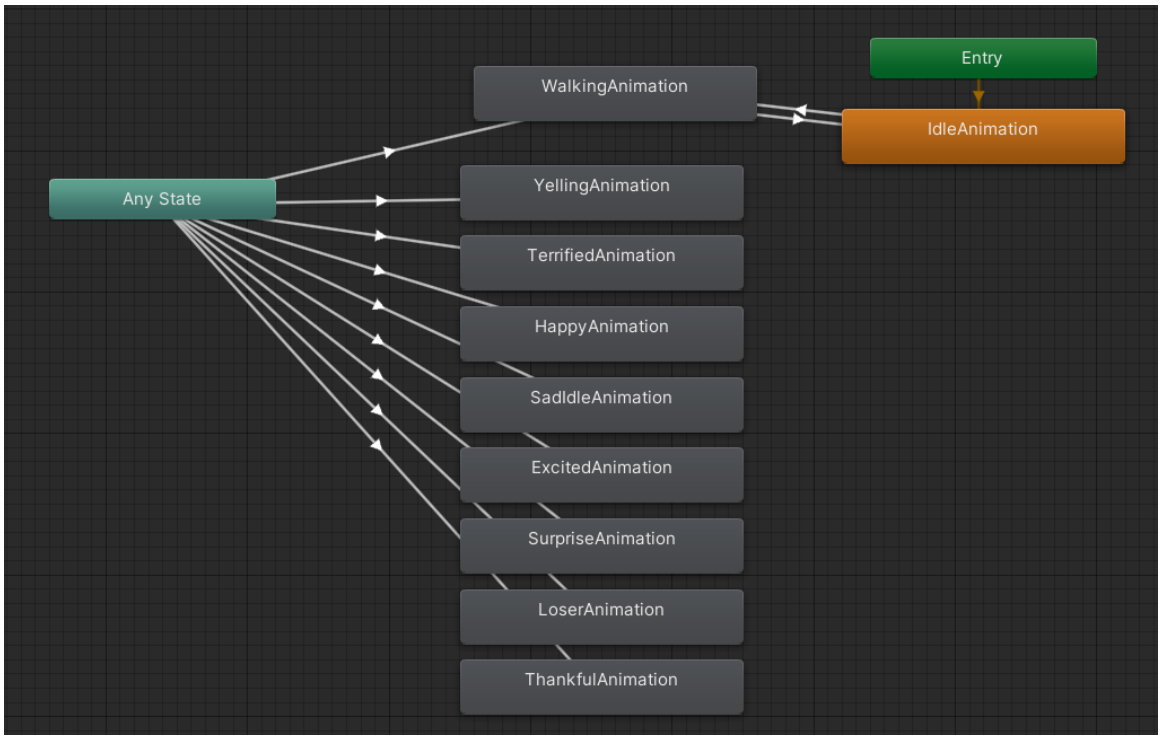

Figura 7.19: NPC Animator component

8
Conclusions and Future Work

When you propose to build a software that has such big ambitions, it's inevitable that the obstacles will show a proportional challenge. It's up for us to dominate, study, and face the problems for big things to be made, and grow even more. This initiative was born in a mix of a lot of effort from different people, and it's desired that it'll continue on this path of growth.

This work initially proposed adding more cultural factors and possible actions towards the NPC's, but it stayed more focused on making a reliable Fuzzy Logic, and gaming developments. The AI has achieved a good decision-making, a functional and concise AI that fulfilled it's purpose of enhancing the original Cultural Model Project, but it needs more treatment, more experimenting, more theory based enhancement

s, and specially more possible incomes that ended up lacking attention and development. The deffuzification process was analysed but ended up not making into the logic, instead, it focused much more on fuzzification processes in order to return fuzzy answers, that matches up to the nature of emotional behaviour, but it can be improved. The idea that *rationality* could affect the size of Fuzzy Sets has opened up a path on amplifying the reliability of this simulation, aiming on a much bigger variant scenario that can't be predicted at the moment.

When it comes to game development, the possibilities are gigantic. That is the beauty of ambitious initiatives like the ones born in PUC-Rio. The Cultural Model Project has grown and evolved, as has it's features and, proportionally, it's original potential. Working on gamedev allows programmers to have a huge perspective of future add-ons. The possibilities are numerous, being that a game can follow several directions at the same time, so much that it's hard to stay focused even while making initial prototypes. It's a fact seen a lot in game industry, in games that maintain themselves on top of sales and keep the community active, like *Destiny 2* and *GTA V* that from time to time, enriches and upgrades their products and maintains players captivated.

Futhermore, this section will point out ideas that wasn't approached in this work, but yet considered at some point. Regarding the Fuzzy Logic, this AI is heavily influenced by it's programmer beliefs and definitions. Taking

that into consideration, there are several other paths that could be taken when modelling this AI, as for example, bearing in mind that *intensity* is so connected to the *rationality* factor from NPC's cultural dimensions, this factor could, and in real life would, affect the probability of which emotion this NPC would feel. In mathematical terms, it means that for each NPC the membership function would alter depending on his rationality. And in practice, the behaviour seen in Fuzzy Logic is that one NPC's emotional Fuzzy Sets would be constructed by different membership functions, that would better represent his specific response towards emotions.

Following that thought, in this study it wasn't very simple to find geometric and matchematical interpretations concerning Plutchik's wheel of emotion, rather than Baffa's[1] research and papers like Yanaru's[23] (that already had an AI product). It's undiscussable that different, more diverse interpretations regarding the mathematical nature of emotional dyads and Plutchik's[15] basic emotion axes would open space for even more AI interpretations concerning emotional processing, specially for game context purposes.

In this work's original proposal, there was an idea for enlarging the model approach by adding more cultural factors and possible input actions. One that was heavily considered but didn't advance was a religious influence factor, that would open up for so much more interpretations and cultural possibilities to be incorporated inside the game. Following the same path, it was also considered having cultures divided geographically inside the city game map, which would enrich much more the player experience. Another add-on that would be interesting, this time concerning more about the gaming experience rather than the AI behind it, is overall adding more VR possible interactions. One cited in this work was a voice-recognition asset, that could income as the *talk rude* and *talk polite* actions. Still following this line of thought, some games approach their user knowledge throughout experiencing the game, mostly in ways for learning the gameplay, but this could be used as an interesting method for learning about cultures: through experiencing with the environment, and not only it's people.

And finally, a cultural simulation project can obviously open up space for aiming in another direction, other than having a simulated fictional scifi experience: by pre-programming existing cultures, this game could be prepared futurally for educational purposes, in order to bring knowledge and notorierity for cultures that resides in remote areas of the planet or even cultures that are already extinct, maintaining alive their legacy.

A
Schedules

In Final Project 1, most of the time dedicated for this research were spent in order to fully understand both Bicalho's and Baffa's model, as well as heavy studying concepts of Fuzzy Logic, and basic theories to explore on cultural factors and emotional thoeries. A lot of erroneous decisions and testings were made that helped have a better understanding of the whole process, model and AI.

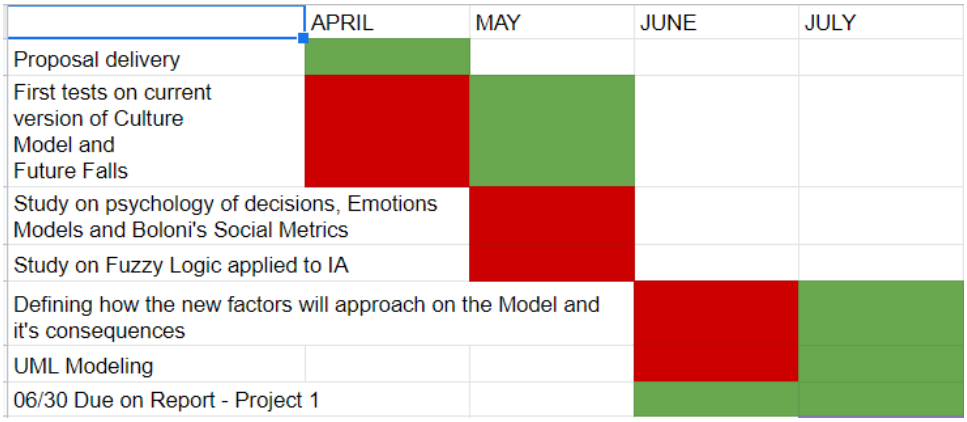

Figura A.1: Schedule for Final Project I

In Figure A.2, a schedule fo Final Project 2 was made, and in Figure A.3 we can see how the activities truly went. A lot of the original planning has changed, as in Final Project 1 there wasn't even a though upon gathering a VR version. This details of VR version were considered upon planning inside the *Game build-up* activity. As you can see, the *Creating narrative* was originally planned as a bigger aspect for the game, but turned out being a small part in it, mainly to guide the player in understanding the game objective in the beginning.

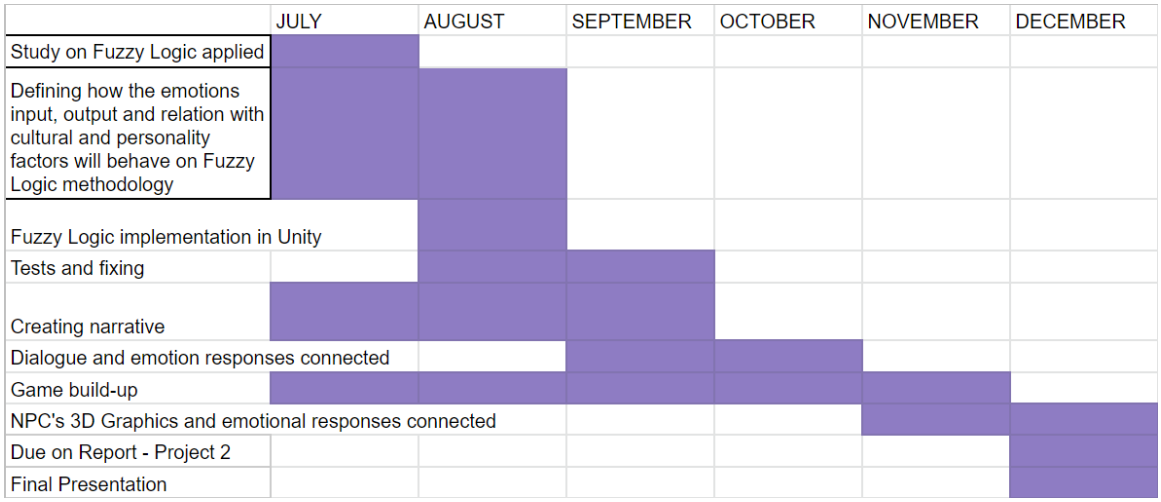

Figura A.2: Schedule planned in Final Project I for Final Project 2

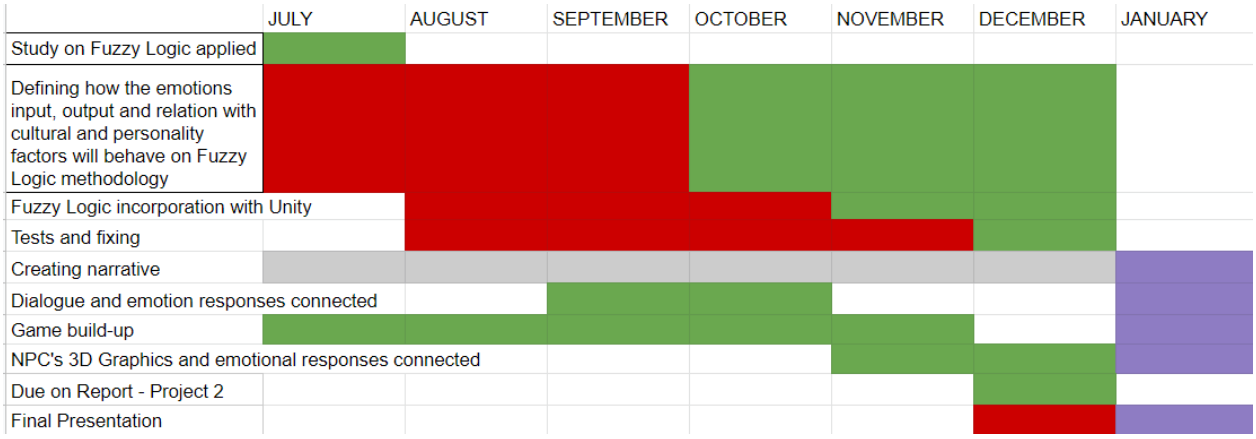

Figura A.3: Schedule for Final Project II

References

[1] Augusto Baffa et al. “Dealing with the Emotions of Non Player Characters”. Em: *2017 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. 2017, pp. 76–87. DOI: 10.1109/SBGames.2017.00017.

[2] M.H.J. Bergsma e P.H.M. Spronck. “Adaptive spatial reasoning for turn-based strategy games”. English. Em: *The Fourth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*. Ed. por M. Mateas e C. Darken. AAAI Press, 2008, pp. 161–166.

[3] Luís Fernando Bicalho, Bruno Feijó e Augusto Baffa. “A Culture Model for Non-Player Characters’ Behaviors in Role-Playing Games”. Em: *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. 2020, pp. 9–18. DOI: 10.1109/SBGames51465.2020.00013.

[4] Manuvra Games Development Blog. *Modeling a Simple AI behavior using a Finite State Machine*. URL: https://archive.ph/20121202054532/http://blog.manuvra.com/modeling-a-simple-ai-behavior-using-a-finite-state-machine/.

[5] Ladislau Bölöni et al. “Towards a computational model of social norms”. Em: *PLOS ONE* 13.4 (abr. de 2018), pp. 1–26. DOI: 10.1371/journal.pone.0195331. URL: https://doi.org/10.1371/journal.pone.0195331%5C%7D.

[6] J. Broughton. *Video Games Help Players Escape Reality*. URL: https://www.tc.columbia.edu/articles/2004/december/video-games-help-players-escape-reality/.

[7] Mat. Buckland. *Programming game AI by example*. eng. Wordware game developer’s library. Plano, Tex: Wordware Pub., 2005. ISBN: 1556220782.

[8] 2019 Cloudwalkin Games. *Hurricane VR - Physics Interaction Toolkit*. URL: https://www.gameassetdeals.com/asset/177300/hurricane-vr-physics-interaction-toolkit#:~:text=HurricaneVR.

[9] L. Gupta. *What is REST - REST API Tutorial*. URL: https://restfulapi.net.

[10] G. Hofstede, G.J. Hofstede e M. Minkov. *Cultures and Organizations: Software of the Mind, Third Edition*. McGraw-Hill Education, 2010. ISBN: 9780071770156. URL: http://books.google.de/books?id=o4OqTgV3V00C.

[11] The Computer Language Co Inc. *WASD keys*. URL: https://www.pcmag.com/encyclopedia/term/wasd-keys.

[12] Oliver P. John e Sanjay Srivastava. "The Big Five Trait taxonomy: History, measurement, and theoretical perspectives." Em: 1999.

[13] S. Johnson. *Playing To Lose: AI and Civilization: GDC 2008*. URL: https://www.gdcvault.com/play/364/Playing-to-Lose-AI-and.

[14] Musings Of A Mario Minion. *Pygmalion's Spectacles: Using Berkeley's Immaterialism to Understand the Potential for Telepresence in Virtual Reality*. URL: https://medium.com/@musingsofamariominion/pygmalions-spectacles-using-berkeley-s-immaterialism-to-understand-the-potential-for-telepresence-46b9e46eba42.

[15] Robert Plutchik. "The emotions: Facts, theories and a new model." Em: *American Journal of Psychology* 77 (1964), p. 518.

[16] The Pallets Projects. *Welcome to Flask - Flask Documentation (2.2x)*. URL: https://flask.palletsprojects.com/en/2.2.x/.

[17] Saqib Qamar e Parvez Ahmad. "Emotion Detection from Text using Fuzzy Logic". Em: *International Journal of Computer Applications* 121 (jul. de 2015), pp. 29–32. DOI: 10.5120/21522-4501.

[18] Simon Rozenberg Travancas. "Biblioteca para simulação de emoções em jogos". Em: *PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO, CENTRO TÉCNICO CIENTÍFICO - CTC DEPARTAMENTO DE INFORMÁTICA, Curso de Graduação em Engenharia da Computação* (2018).

[19] R. Romão Silva. *ORIGENS E INFLUêNCIAS DA REALIDADE VIRTUAL: a influência da História da Arte na tecnologia imersiva*. URL: https://recode.org.br/wp-content/uploads/2018/03/cineastas360-texto-origens-e-influencias-da-realidade-virtual.pdf.

[20] Sarah Slater, Robert Moreton e Kevan Buckley. "A.I. Techniques for Modelling Anger in Emotional Agents". Em: (jan. de 2008).

[21] The scikit-fuzzy development team. *User guide - skfuzzy v0.2 docs*. URL: https://pythonhosted.org/scikit-fuzzy/user_guide.html.

[22] Unity Technologies. *Unity Documentation*. URL: https://docs.unity.com.

[23] Yanaru Torao et al. "An emotion processing system based on fuzzy inference and subjective observations". Em: *Information Sciences* 101.3 (1997). Advanced Neuro-Fuzzy Techniques and Their Applications, pp. 217–247. ISSN: 0020-0255. DOI: https://doi.org/10.1016/S0020-0255(97)00011-X\%7D. URL: https://www.sciencedirect.com/science/article/pii/S002002559700011X%5C%7D.