Emulação de emoções em personagens não jogáveis através de técnicas de computação afetiva

Matheus E. C. de Souza
mecs@edu.univali.br
Universidade do Vale do Itajaí
Itajaí, Santa Catarina, BR

Rudimar L.S. Dazzi
rudimar@univali.br
Universidade do Vale do Itajaí
Itajaí, Santa Catarina, BR

RESUMO

Um dos desafios atuais dos jogos é a criação de oponentes artificiais que proporcionem uma experiência de uma partida entre dois jogadores. Entende-se que um fator determinante para isso são as emoções que o jogador responde durante uma partida. Este trabalho buscou utilizar as emoções como uma ferramenta de jogabilidade, de modo a simular nos agentes o comportamento emocional presente nos jogadores, por meio de técnicas de computação afetiva. O trabalho foi desenvolvido utilizando o *framework* GAMYGDALA juntamente com uma inteligência artificial baseada em lógica Fuzzy. Ao final, o trabalho atingiu seus objetivos, os agentes apresentaram comportamento emocional que corresponde às suas convicções preestabelecidas e situação presente no jogo.

KEYWORDS

Computação Afetiva, Personagens Não Jogáveis, Sistemas Multiagentes.

1 INTRODUÇÃO

Um personagem não jogável (NPC - *Non-Playable Character*), no contexto de jogos digitais, é um personagem cujo comportamento é controlado por um programa de computador, não por um jogador. O NPC desempenha um papel importante dentro de um jogo, constantemente o jogador pode interagir com ele, como por exemplo, falar, atirar e cooperar com àquela inteligência artificial (IA).

Um dos principais objetivos ao desenvolver um jogo é criar um ecossistema capaz de entreter e envolver seus usuários [7]. Os métodos de IA já permitem desenvolver agentes capazes de competir com jogadores de nível profissional [4]. A criação de NPCs consistentes o suficiente, para a maioria dos jogos, já é considerado um problema resolvido, porém o desafio é a criação de oponentes artificiais divertidos, que proporcionem uma experiência agradável ao jogador, se aproximando da experiência de uma partida entre dois jogadores [11].

[3] reconhecem que o papel da emoção em um jogo é uma preocupação chave. Durante uma partida, os jogadores respondem emocionalmente às interações que acontecem nelas, se sentindo irritados quando uma tarefa é muito desafiadora para ser completada ou tristes quando um personagem morre dentro do enredo do jogo [10]. Essas emoções influenciam diretamente o comportamento do jogador, modificando sua tomada de decisão.

A computação afetiva (CA) encapsula uma nova abordagem em inteligência artificial, com o foco em construir computadores com capacidade de demonstrar afetos humanos. As técnicas de CA podem ser aplicadas a domínios onde os sentimentos e emoções desempenham um papel importante, tais como entretenimento, comunicação expressiva ou educação [6].

Pensando nisso, o presente trabalho se propôs a desenvolver um sistema capaz de emular o comportamento emocional nos personagens não jogáveis em um jogo, utilizando-se de técnicas de computação afetiva. As emoções foram pautadas em uma teoria emocional consolidada e introduzidas no sistema através de um modelo computacional já preestabelecido.

2 GAMYGDALA

[9] desenvolveram o GAMYGDALA como um sistema de prova de conceito, de que a modelagem emocional dentro de um jogo poderia ser terceirizada por um sistema externo, de forma semelhante a uma *engine* de física, como já ocorre na maioria das plataformas de desenvolvimento de jogos atuais.

GAMYGDALA é baseado no modelo de [5]. O modelo, e todas outras teorias de avaliação, assumem que um agente tem alguma forma de meta, preocupação ou um conjunto de estados preferidos e um mecanismo para perceber o mundo e agir naquele mundo. Um pressuposto fundamental para se utilizar a ferramenta em um jogo é, portanto, que o jogo possa ser modelado através de eventos, que metas possam ser definidas para os NPCs que precisam da simulação de emoção e que a relevância dos eventos para os objetivos seja especificado.

Cada NPC em um jogo pode ser visualizado como um agente autônomo: dentro do quadro de seu propósito e possibilidade de ação, ele percebe o estado do jogo e escolhe comportamentos alternativos que se destinam a servir ao seu propósito. Sua meta é um estado desejado. Pode-se utilizar como exemplo um soldado que quer matar o jogador. Ao matar o jogador, o estado final foi alcançado. Um NPC pode ter várias metas, por exemplo, um soldado que quer impedir o jogador de entrar em um prédio pode ter como metas: manter o jogador distante e matar o jogador.

Para utilizar o GAMYGDALA, o desenvolvedor precisa definir explicitamente uma ou mais metas para cada NPC ou classe. Em seguida, o desenvolvedor agora pode definir quais eventos são relevantes para cada meta e especificar isso para a *engine*. A única modificação essencial para o jogo é que os eventos entregues para IA que controla o comportamento dos NPCs, agora também são entregues a GAMYGDALA. Por fim, é avaliado emocionalmente o evento e retornado a emoção mais plausível.

O trabalho foi desenvolvido a partir do jogo Survival Shooter, jogo que se ambienta em um quarto infantil onde o personagem controlado pelo jogador é um bebê em um pesadelo. O bebê é perseguido por pelúcias zumbis e deve se defender utilizando uma arma com munição ilimitada. O jogador deve sobreviver às hordas de inimigos que são geradas infinitamente, inimigos estes que são divididos em três classes: coelhos, elefantes e ursos. O fim do jogo consiste na morte do jogador. Pode-se observar o jogo na Figura 1.

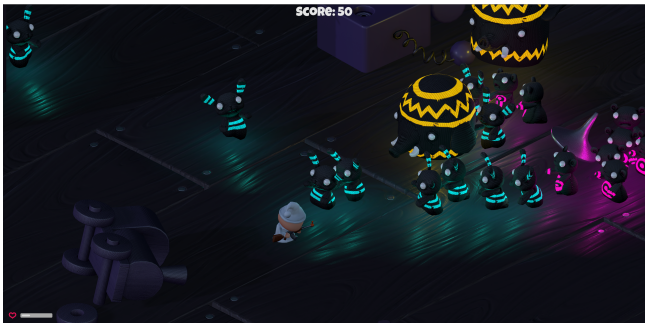

Figura 1: Jogo base utilizado no projeto

O sistema foi construído utilizando o jogo Survival Shooter como base, fazendo o uso das ferramentas de jogabilidade já presentes no jogo, as quais não sofreram alterações durante o desenvolvimento. A ideia do jogo e suas principais mecânicas foram mantidas, o jogador deve se defender de infindas hordas com uma arma de munição ilimitada, enquanto os personagens o perseguem e atacam ao entrar em contato.

O trabalho focou em adaptar o funcionamento interno do jogo, de modo a incluir o comportamento emocional nos personagens não jogáveis, e através da coloração exprimir a emoção dominante para cada agente individualmente.

O funcionamento do sistema é baseado em eventos, que compreendem as ações significativas dos personagens dentro do jogo. O GAMYGADALA é responsável por receber os eventos do jogo, avaliar emocionalmente e retornar as emoções de cada personagem não jogável através de um *array* que corresponde ao estado emocional atual do agente, cada emoção gerada é salva nesse *array* e decai seguindo a função de decadência.

O estado emocional do agente é avaliado por uma inteligência artificial baseada em lógica Fuzzy, a qual calcula o impacto da emoção na dinâmica do jogo, alterando os atributos de acordo com o conjunto de emoções retornados. Além da representação comportamental, foi utilizada uma ferramenta visual para explicitar a emoção dominante de cada NPC. Cada emoção foi classificada com uma cor respectiva e externalizada no corpo do personagem ao longo do jogo. A Figura 2 representa o funcionamento geral do trabalho, onde os eventos são passados para o GAMYGDALA, que retorna a emoção pertinente para a lógica Fuzzy, que por fim computa o comportamento adequado do NPC.

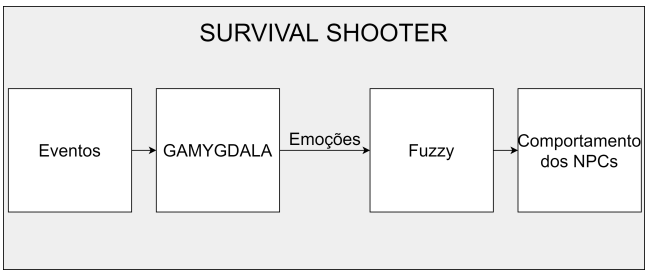

Figura 2: Visão geral do sistema

2.1 Modelagem do sistema de avaliação emocional

A modelagem dos agentes por meio do GAMYGADALA foi separada por classe de NPC, diferindo-se entre os coelhos, elefantes e ursos zumbis. Entretanto, por mais que as características inicias dos agentes sejam iguais dentro de uma mesma classe, cada agente tem sua gestão e dinâmica de emoções individualmente.

O trabalho utilizou 7 das 16 emoções presentes e definidas no GAMYGADALA, que são: alegria, alívio, angústia, frustração, medos confirmados, raiva e satisfação. A utilização de apenas sete das emoções presentes no *framework*, justificou-se, em grande parte pelas limitações do próprio jogo utilizado. As interações entre os agentes são limitadas, o que dificulta a utilização de emoções mais complexas.

Os coelhos têm como sua maior meta matar o jogador e são utilizados como principal ferramenta de ataque no jogo, visto que seu comportamento está diretamente ligado com a barra de vida do jogador. Eles possuem uma inimizade com os ursos, utilidade negativa relacionada à pontuação, e senso de autopreservação.

Os elefantes são a classe com os maiores atributos do jogo, possuindo maiores números em vida e dano. Sua utilidade no jogo será de proteção dos outros personagens, possuindo a menor utilidade em autopreservação no jogo, fato que demonstra que ele estará mais preocupado em proteger os companheiros do que a si próprio.

Por fim, os ursos fazem parte da classe intermediária, sua principal funcionalidade é de servir como apoio para as metas do coelho e do elefante, aumentando a dinâmica do jogo. Como por exemplo, a inimizade percebida entre os coelhos com os ursos é retribuída, possuindo a mesma utilidade para a morte de um coelho, como também a mesma utilidade de autopreservação.

2.2 Impacto das emoções no jogo

O jogo base utilizado permitiu o desenvolvimento do trabalho devido sua grande quantidade de interações em tempo real, como também a simplicidade destas, visto que a interação está limitada aos ataques entre os agentes e jogador. A grande questão para o desenvolvimento, era como representar as emoções escolhidas, seguindo a narrativa apresentada na seção anterior.

Ao perceber o cenário em que o jogo se apresentava, foi decidido categorizar as emoções em torno da agressividade resultante nos agentes. Cada emoção tornaria os personagens não jogáveis mais ou menos agressivos dentro do jogo. Considerando que eles estão mais agressivos quando seus atributos estão com valores elevados, e menos agressivos quando seus atributos estão com valores baixos.

Neste contexto, as sete emoções foram separadas arbitrariamente, onde as emoções de maior valência seriam: raiva, quando o agente apresentaria maior agressividade, por se tratar de um momento de fúria; e alegria, quando o agente apresentaria menor agressividade, por se tratar de um momento de contentamento.

Ao utilizar um sistema Fuzzy, foi possível definir de forma coerente o impacto de um conjunto de emoções com variadas intensidades dentro do sistema. Um agente pode, ao mesmo tempo, apresentar emoções contraditórias com diferentes intensidades, como por exemplo: alegria, com baixa intensidade e raiva com alta intensidade.

2.3 Representação das emoções em cores

A representação das emoções através da coloração foi feita em cima do modelo base dos personagens, conforme pode-se observar na Figura 3. A coloração é possibilitada a partir da inclusão de uma textura, criada externamente no âmbito da plataforma Unity. Os valores de cor desta textura foram modificados de acordo com a emoção dominante no presente momento.

Figura 3: A esquerda, modelo base do personagem urso; e a direita o modelo do mesmo personagem com a coloração padrão do jogo aplicada

Foi utilizado uma inspiração nas cores da roda de emoções de [8], visando um embasamento mais coerente e uniforme da distinção entre as emoções e as cores que devem representá-las. Duas das sete emoções utilizadas já estão presentes no modelo de Plutchik, o que necessitou de uma adaptação para as restantes. A Tabela 1 demonstra a emoção utilizada no trabalho e a respectiva cor presente no modelo.

Tabela 1: Atribuição de cores para cada emoção

Emoção	Respectiva na roda	Cor
Alegria	Alegria	Amarelo
Alívio	Surpresa	Ciano
Angústia	Angústia	Azul
Frustração	Abatimento	Azul Ardósia
Medos confirmados	Terror	Verde
Raiva	Irritação	Vermelho
Satisfação	Extasia	Amarelo Claro

2.4 Modelagem do sistema Fuzzy

O sistema Fuzzy conta com sete entradas, uma para cada emoção presente no trabalho. O estado emocional de cada agente é composto por uma ou mais emoções. Essas emoções possuem uma intensidade atrelada, que decai ao longo do tempo, seguindo a função de decaimento presente para cada classe. Com isso em mente, as entradas possuem dois conjuntos: baixo e alto. Todas as sete emoções seguem a função de pertinência conforme a Figura 4, procurando categorizar a intensidade da emoção dentre as variáveis.

O jogo dispõe de quatro atributos para os agentes: vida, dano de ataque, velocidade de ataque e velocidade de movimento. Cada

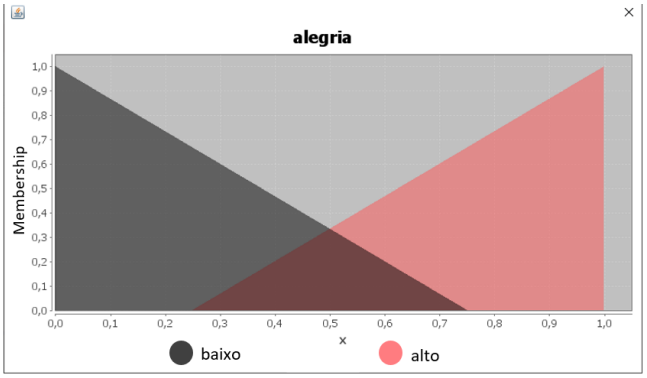

Figura 4: Fuzificação da emoção de alegria

saída do sistema contempla um modificador para cada um dos atributos, exceto o de vida, de forma a retornar um multiplicador para o valor base de cada agente. Todas as três saídas possuem cinco termos linguísticos, funções de pertinência triangulares e método de defuzificação centroide. Pode-se observar a função de pertinência do atributo de dano na Figura 5.

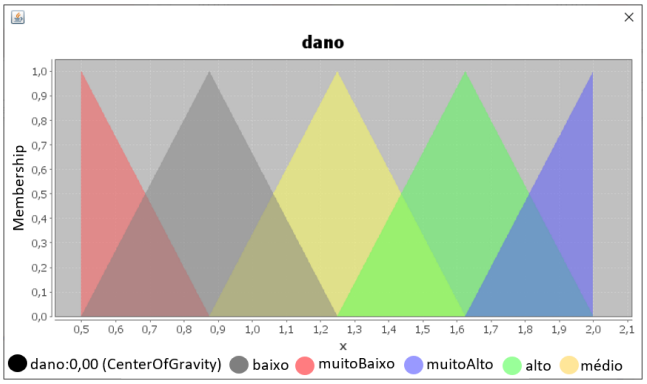

Figura 5: Defuzificação da variável de dano

A escolha da função de pertinência triangular para as entradas e saídas baseou-se na tentativa das emoções se comportarem de forma linear e constante entre suas transições. Percebe-se que com o uso de uma função senoidal, a suavidade do sistema seria maior, porém posteriormente foi percebido que a ferramenta utilizada para implementação do sistema apenas possuía funções de pertinência triangulares e trapezoidais, sendo este o principal motivo de escolha do método.

Cada uma das entradas afetará a saída de forma distinta. As emoções de alegria e raiva foram consideradas como as emoções de maior impacto no sistema, afetando todas as três saídas. As emoções de alívio, angústia e medos confirmados afetam apenas duas das três saídas, e por fim, as emoções de frustração e satisfação afetam apenas uma das saídas. Para melhor entendimento, pode-se observar a Tabela 2.

A construção das regras partiu do princípio de que caso todas as entradas estejam no conjunto baixo, todas as saídas do sistema

Tabela 2: Impacto das emoções nas saídas

Emoção	Vel. movimento	Dano	Vel. de ataque
Alegria	-	-	-
Alívio		-	-
Angústia		+	+
Frustração			+
Medos confirmados	+	+	
Raiva	+	+	+
Satisfação	-		

permanecem no termo médio. Conforme as emoções são recebidas e alteram seu estado para o conjunto alto, as saídas são deslocadas. Considerando uma cobertura de todas as possibilidades, o sistema necessitou de 384 regras, pois o sistema Fuzzy possui sete entradas com dois conjuntos, baixo e alto, para cada uma das três saídas.

O sistema Fuzzy foi incorporado na plataforma de desenvolvimento do jogo por meio da ferramenta construída por [2], um editor visual baseado no *framework* AForge.NET, que pode ser incluído na Unity como um *plugin*, facilitando a construção de sistemas baseados em lógica difusa. O Fuzzy Logic Editor (FLE) possui apenas duas telas, que permitem ao desenvolvedor incluir as variáveis linguísticas e as regras do sistema. Com o uso do FLE, foram necessários poucos ajustes para se obter o sistema funcional.

2.5 Comunicação entre o Gamygdala e Unity

O ponto de maior risco para o presente trabalho era a integração entre o *framework* GAMYGDALA e a plataforma de desenvolvimento, visto que o *framework* é escrito inteiramente em JavaScript, em apenas um arquivo, enquanto a Unity é codificada em C#, impossibilitando a comunicação direta entre as duas ferramentas.

Para isso foi utilizado a biblioteca Jurassic [1], uma implementação da linguagem ECMAScript sob licença MIT, que tem como objetivo fornecer o melhor desempenho e a implementação mais compatível com os padrões de JavaScript para .NET. Possibilitando aos desenvolvedores de um programa em .NET compilar e executar códigos em JavaScript.

Com a inclusão da biblioteca, foi possível compilar o arquivo que continha todo o *framework* GAMYGDALA, como também possibilitou a comunicação em tempo real entre o *framework* e a ferramenta de desenvolvimento. Dentro do ambiente da Unity, foi criado uma classe para gerenciar a compilação do GAMYGDALA, onde por meio da função $ExecuteFile$ do Jurassic, todo o contexto do arquivo era compilado e carregado no ambiente em C#. Por fim, a comunicação entre os dois ambientes foi possibilitada através de variáveis do tipo JSON.

3 RESULTADOS

De modo a concluir o correto funcionamento do trabalho proposto, o sistema foi avaliado em três diferentes aspectos: a coerência dos conjuntos de emoções retornados pela ferramenta de avaliação emocional, dado um determinado contexto; o valor dos modificadores retornados pelo sistema Fuzzy após entrada das novas emoções; e por fim, avaliado se o sistema está externalizando corretamente a cor da emoção dominante para cada agente inserido.

O jogo foi submetido a oito cenários de teste diferentes, primeiramente instanciando apenas um agente de cada classe e posteriormente em duplas, onde pode-se observar as diferenças entre as classes, como também que seja perceptível que os agentes tiveram uma avaliação emocional independente, mesmo que pertençam a mesma classe, o que comprovou-se verdadeiro. Os cenários envolveram o jogador atacar diferentes agentes durante a partida, como também ser atacado por eles. A cada evento confirmado foi anotado o estado emocional retornado pela ferramenta. Para fins de documentação, será apresentado três dos cenários aplicados.

3.1 Avaliação emocional

Os dois cenários que serão abrangidos consistem em o jogador atacar diretamente um urso repetidamente até que ele seja eliminado. Ao atacar o urso pela primeira vez, percebe-se que o coelho retorna a emoção de alegria em maior intensidade, além de angústia e raiva com menor intensidade. Por sua vez, o urso atacado retorna a emoção de angústia e raiva em grande intensidade. O elefante, ao ver seu colega ser atacado também retorna as emoções de angústia e raiva, no entanto, apresenta menor intensidade que o urso.

Ao retomar o ataque contra o urso, percebe-se um aumento na emoção de raiva, enquanto os demais agentes não sofrem incremento emocional, apenas têm suas emoções decaídas pelo tempo. Quando o urso é eliminado, o coelho retorna as emoções de alegria e satisfação, pois seu rival foi morto, além de angústia, raiva e medos confirmados pois o jogador pontuou. Por fim, o elefante retorna as emoções de angústia e raiva com maior intensidade e medos confirmados, pois o urso foi eliminado.

3.2 Avaliação do sistema Fuzzy

A avaliação do sistema Fuzzy consistiu em duas etapas. Primeiramente, valores ideais foram inseridos de forma manual no sistema, que pudessem ser facilmente aferidos, e observado suas respectivas saídas, no qual já pôde-se perceber o correto funcionamento do sistema. Em seguida, foram utilizados os três cenários em que o agente urso é atacado para demonstrar um funcionamento real do sistema e observou-se as saídas deslocarem conforme planejado. Pode-se observar os resultados nas Tabelas 3, 4 e 5.

3.3 Avaliação da representação emocional

Além do recurso comportamental, o jogo apresenta um recurso visual para o jogador entender qual é a emoção dominante em um determinado momento. O funcionamento é bastante simples, o sistema deve entender a emoção com maior intensidade no estado emocional de cada agente e expelir a cor determinada.

Ao iniciar o jogo todos os agentes expelem a cor branca, pois não possuem estado emocional. Conforme as crenças são concretizadas, o estado emocional dos agentes é alterado. No primeiro cenário, o jogador ataca diretamente o urso, o retorno emocional pode ser observado na Figura 6.

Tabela 3: Entradas e saídas dos agentes após atacar o urso

Agente	Alegria	Angústia	Raiva	Vel. movimento	Vel. de ataque	Dano
Coelho	0,71	0,55	0,55	1,13	1,32	1,32
Elefante		0,79	0,79	1,62	1,86	1,86
Urso		0,83	0,83	1,62	1,86	1,86

Tabela 4: Entradas e saídas dos agentes após atacar o urso novamente

Agente	Alegria	Angústia	Raiva	Vel. movimento	Vel. de ataque	Dano
Coelho	0,55	0,37	0,37	1,20	1,32	1,32
Elefante		0,64	0,64	1,50	1,61	1,61
Urso		0,71	0,88	1,62	1,81	1,81

Tabela 5: Entradas e saídas dos agentes após a morte do urso

Agente	Alegria	Angústia	Medos Confirmados	Raiva	Satisfação	Vel. movimento	Vel. de ataque	Dano
Coelho	0,75	0,60	0,23	0,37	0,38	1,04	1,36	1,36
Elefante		0,82	0,48	0,64	0,82	1,69	1,84	1,84

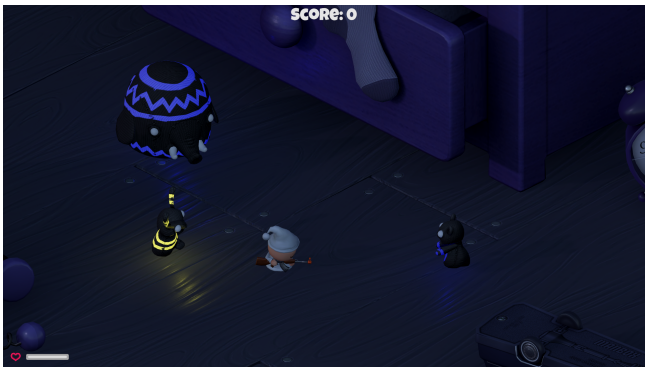

Figura 6: Coloração dos agentes ao atacar o urso; elefante e urso com a coloração azul, representando a emoção de angústia; e coelho com a coloração amarela, representando a emoção de alegria

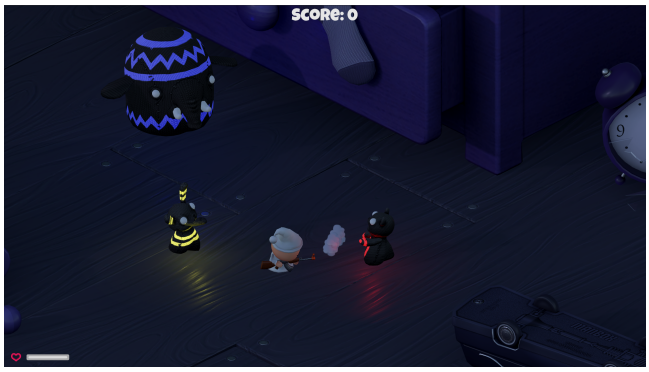

Figura 7: Coloração dos agentes ao atacar o urso novamente; elefante com a coloração azul, representando a emoção de angústia; urso com a coloração vermelha, representando a emoção de raiva; e coelho com a coloração amarela, representando a emoção de alegria

A angústia está retornando como emoção dominante no urso e no elefante, enquanto o coelho está retornando alegria. Ao retomar o ataque, o urso começa a retornar raiva como emoção dominante, enquanto os demais agentes mantiveram suas emoções dominantes anteriores, como pode-se observar na Figura 7

A partir dos resultados dos dois cenários apresentados, pôde-se concluir o correto funcionamento da representação emocional, permitindo ao usuário distinguir a emoção de maior intensidade em cada agente separadamente. Isto também conclui a seção, que em geral apresentou resultados muito satisfatórios dentre os planejados.

4 CONCLUSÃO

Este trabalho apresentou o desenvolvimento de um sistema que se utiliza de técnicas de computação afetiva, de modo a auxiliar na credibilidade do comportamento dos NPCs e expandir a vida útil de um jogo. O sistema foi desenvolvido com o uso do *framework* GAMYGDALA, responsável pela avaliação emocional dos agentes. Esta inclusão foi viabilizada pela biblioteca Jurassic, que possibilita compilar códigos na linguagem JavaScript em um contexto C#. O sistema difuso foi construído na Unity através do editor visual Fuzzy Logic Editor. Por fim, foi representado de forma visual a emoção dominante de cada agente em tempo real através da coloração de seus corpos.

O trabalho apresentou resultados nos quais pôde-se comprovar o funcionamento correto do sistema responsável pela avaliação emocional, apresentando estados emocionais coerentes para o contexto inserido. O sistema difuso construído também apresentou

coerência, comprovando sua eficiência através de testes ideais e posteriormente testes reais do jogo. O sistema também foi capaz de identificar a emoção dominante de cada agente e expelir em seus corpos a cor que a representa.

Durante os testes, pôde-se perceber que as emoções afetaram diretamente o comportamento dos agentes. Deste modo, quando a emoção dominante era felicidade, o jogador não precisou se preocupar visto que a agressividade dos agentes diminuiu drasticamente, reduzindo o dano causado por eles. Por outro lado, quando a emoção dominante foi raiva, havia a necessidade de preocupação do jogador, uma vez que qualquer ataque poderia ser fatal devido ao aumento da agressividade dos agentes.

Para trabalhos futuros, pretende-se portar o GAMYGDALA para C#, de modo a incorporá-lo diretamente na Unity, melhorando o desempenho, além de aumentar a flexibilidade do desenvolvimento. Pretende-se desenvolver um jogo que se encaixe melhor como um produto final, onde o estado emocional retornado possa refletir em um comportamento mais claro para o jogador, não só um multiplicador de seus atributos, como também possibilitar o uso de todas as 16 emoções presentes na ferramenta.

REFERÊNCIAS

[1] Paulo Bartrum. 2019. Jurassic. https://github.com/paulbartrum/jurassic
[2] Francisco Diego Moreira Feitosa. 2019. Desenvolvimento de um editor de lógica Fuzzy para o motor de jogo Unity. (2019).
[3] Antje Herbon, Christian Peter, Lydia Markert, Elke Van Der Meer, and Jörg Voskamp. 2005. Emotion studies in HCI-a new approach. In *Proceedings of the 2005 HCI International Conference*.
[4] Inseok Oh, Seungeun Rho, Sangbin Moon, Seongho Son, Hyoil Lee, and Jinyun Chung. 2021. Creating pro-level AI for a real-time fighting game using deep reinforcement learning. *IEEE Transactions on Games* (2021).
[5] Andrew Ortony, Gerald L Clore, and Allan Collins. 1988. The cognitive structure of emotions Cambridge. *UK: Cambridge University Press9* (1988).
[6] Rosalind W Picard. 2000. *Affective computing*. MIT press.
[7] David Pinelle, Nelson Wong, and Tadeusz Stach. 2008. Heuristic evaluation for games: usability principles for video game design. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1453–1462.
[8] Robert Plutchik. 2001. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American scientist* 89, 4 (2001), 344–350.
[9] Alexandru Popescu, Joost Broekens, and Maarten Van Someren. 2013. Gamygdala: An emotion engine for games. *IEEE Transactions on Affective Computing* 5, 1 (2013), 32–44.
[10] Doyo Setiono, David Saputra, Kaleb Putra, Jurike V Moniaga, and Andry Chowanda. 2021. Enhancing Player Experience in Game With Affective Computing. *Procedia Computer Science* 179 (2021), 781–788.
[11] Kaori Yuda, Maxim Mozgovoy, and Anna Danielewicz-Betz. 2019. Creating an Affective Fighting Game AI System with Gamygdala. In *2019 IEEE Conference on Games (CoG)*. IEEE, 1–4.