UNIVERSIDADE FEDERAL DO TOCANTINS
CÂMPUS DE ARAGUAÍNA
LICENCIATURA EM MATEMÁTICA

PEDRO DARC DA CRUZ ASSUNÇÃO

A MATEMÁTICA NA PRODUÇÃO DE JOGOS DIGITAIS COM INTELIGÊNCIA ARTIFICIAL

Araguaína/TO
2021

PEDRO DARC DA CRUZ ASSUNÇÃO

A MATEMÁTICA NA PRODUÇÃO DE JOGOS DIGITAIS COM INTELIGÊNCIA ARTIFICIAL

Trabalho de conclusão de curso apresentado ao curso de Licenciatura em matemática da Universidade Federal do Tocantins, como requisito parcial para a obtenção de título de Licenciado em Matemática.

Orientador: Prof. Dr. Deive Barbosa Alves

Araguaína/TO
2021

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Tocantins

A851m Assunção, Pedro Darc da Cruz .

A Matemática na produção de jogos digitais com Inteligência Artificial. / Pedro Darc da Cruz Assunção. – Araguaína, TO, 2021.

61 f.

Monografia Graduação - Universidade Federal do Tocantins – Câmpus Universitário de Araguaína - Curso de Matemática, 2021.

Orientador: Deive Barbosa Alves

1. Inteligência Artificial. 2. Jogos digitais. 3. Lógica Fuzzy. 4. Modelos Matemáticos. I. Título

CDD 510

Elaborado pelo sistema de geração automática de ficha catalográfica da UFT com os dados fornecidos pelo(a) autor(a).

PEDRO DARC DA CRUZ ASSUNÇÃO

A MATEMÁTICA NA PRODUÇÃO DE JOGOS DIGITAIS COM INTELIGÊNCIA ARTIFICIAL

Monografia foi avaliada e apresentada à UFT – Universidade Federal do Tocantins – Câmpus Universitário de Araguaína, Curso de Licenciatura em Matemática para obtenção do título de Licenciatura em Matemática e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Data de aprovação: 24/05/ 2021

Banca Examinadora

Prof. Dr. Deive Barbosa Alves, UFT

Profa. Dra. Samara Leandro Matos da Silva, UFT

Profa. Hevellyn Tays Lima da Silva

Araguaína, 2021

Dedico este trabalho aos meus familiares e a todos que fizeram parte desta trajetória.

AGRADECIMENTOS

Em primeiro lugar a Deus, devo a ele tudo o que sou. A minha mãe, Maria Aparecida, por sempre me incentivar e acreditar na minha capacidade. Ao meu filho, Pedro Henrique, que tanto amo. Aos meus tios: Waldecy, Vagner, Venerando e José por sempre estarem ao meu lado. Ao meu primo, José Maria, por ser o melhor amigo de uma vida inteira.

Ao colegiado de Matemática, meu muito obrigado, em especial aos professores: Álvaro Yucra, Fernanda Vital, Samara Leandro, Yukiko Massago. Ao meu orientador, Deive Barbosa, por toda paciência e dedicação na elaboração do presente trabalho, minha eterna gratidão.

Agradeço também aos colegas e novos amigos que acabei conhecendo durante o curso, principalmente das turmas 2017.1 e 2018.1. Atalia, Adrielly, Bárbara, Bruna, Dáffny, Daniel, Djane, Érica, Fernanda, Gabriel, Gabriella, Guilherme, Janaína, João Paulo, José Cláudio, Kemile, Kevellyn, Luccas, Ludemila, Maisa, Marcella, Marcos, Matheus, Morgana, Pablo, Pedro, Raieli, Ronaldo, Sara, Sinara, Sued, Thiago, Welder e a tantos outros que contribuíram de alguma forma para este momento.

RESUMO

Este trabalho teve como objetivo elencar os conceitos matemáticos que são explorados na construção de jogos digitais. Usou-se a pesquisa Bibliográfica em uma abordagem qualitativa. Os textos que nos serviram de base de dados (*corpus*) da investigação foram dois livros sobre desenvolvimento de jogos digitais dos autores: Tonéis (2015) e Buckland (2005). Como técnica para analisar os dados obtidos Desconstrução e unitarização realizada no *corpus* usou-se a Análise Textual Discursiva (ATD), a qual nos levou a duas unidades de sentido: o desenvolvimento de jogos digitais sem Inteligência Artificial e o desenvolvimento de jogos digitais com Inteligência Artificial. Um movimento dialético que nos fez obter como resultados modelos matemáticos que vão desde ao uso do Teorema de Pitágoras à teoria de conjuntos Fuzzy na implementação de jogos digitais. O que nos fez concluir que o novo emergente para as criações computacionais é também a urgência de, nos cursos de formação que trabalham com a Matemática, introduziram no processo formativo a Matemática Fuzzy.

Palavras-chaves: Inteligência artificial. Jogos digitais. Lógica Fuzzy. Modelos Matemáticos.

ABSTRACT

The objective of this work to list the mathematical concepts that are explored in the construction of digital games. Bibliographic research was used in a qualitative approach. The texts that served as the research database (corpus) were two books on the development of digital games by the authors: Tonéis (2015) and Buckland (2005). As a technique to analyze the data obtained deconstruction and unitarization carried out in the corpus, Discursive Textual Analysis (ATD) was used, which led us to two units of meaning: the development of digital games without Artificial Intelligence and the development of digital games with Intelligence Artificial. A dialectical movement that resulted in mathematical models that range from the use of the Pythagorean theorem to the Fuzzy set theory in the implementation of digital games. What made us conclude that the new emerging for computational creations is also the urgency of introducing Fuzzy Mathematics into the training courses that work with Mathematics.

Key-words: Artificial intelligence. Digital games. Fuzzy logic. Mathematical Models.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

IA - Inteligência Artificial

CMU - *Carnegie Mellon University*

CPU - *Central Processing Unit*

CRT - *Cathode Ray Tube*

FPS - *First Person Shooter*

FSM - *Finite State Machine*

FuSMS - *Fuzzy Finite State Machines*

HP - *Hit Points*

IBM - *International Busines Machines*

MIT - *Massachusetts Institute of Technology*

NPC - *Non-player Character*

RNA – Redes Neurais Artificiais

RTS - *Real-Time Strategy*

SUMÁRIO

1- INTRODUÇÃO

Este trabalho de conclusão de curso busca explorar os modelos matemáticos na construção de jogos digitais. Essa não invenção dos jogos digitais deu-se pelo pouco tempo que tivemos para desenvolver o trabalho, uma vez que compreender a matemática por trás dos jogos, por si só, nos tomou bastante tempo, pois, por exemplo, a Matemática Fuzzy não nos é ensinado na graduação de nossa universidade. E aliado a isso está o fato de que a construção de jogos digitais é, na maioria das vezes, realizada por uma equipe de diversas áreas do conhecimento humano como: Computação, Física, Matemática, Letras, Artes, entre outra (TONÉIS, 2015).

Desse ponto de vista não se aplicou os jogos em sala de aula, por não os termos construídos, mas poderíamos ter discutindo-os no campo abstrato do ensino e aprendizagem da matemática, uma vez que poderíamos ter aplicado oficinas ou minicursos de discussão das matemáticas aplicadas aos jogos. Não fizemos isso por estarmos em período de isolamento social devido a pandemia do Covid-19, “uma infecção respiratória aguda causada pelo coronavírus SARS-CoV-2” (BRASIL, 2021).

Assim, este trabalho trata-se de um estudo teórico, inicial, uma vez que não temos a pretensão e nem conseguimos abarcar todas os modelos matemáticos de se trabalhar a matemática em jogos digitais.

Dito isso, a importância e força desta pesquisa, está, segundo Tonéis (2015, p. 12, grifos do autor) em apresentar que “a Matemática nos jogos digitais visa fornecer modelos e métodos numéricos que auxiliem na produção de mecânicas e de *Level Designs* inovadores para o universo dos *games*”. Os dizeres que constatam que o trabalho de desenvolvimento de jogos digitais (*games*) é um trabalho que engloba diversas áreas do conhecimento, entre elas os saberes matemáticos. Portanto, o professor, aluno ou matemático que se envolve na construção de jogos digitais seria o responsável por apresentar no projeto de construção do jogo (*Desingns*) modelos e métodos numéricos que atribuam tanto as “regras, restrições e objetivos que foram criados para afetar a experiência do jogador, o movimento do jogo” (Mecânica do jogo) como para estabelecer “o mapeamento, o desenho, de cada fase do jogo. É fundamental para a experiência do jogador e deve ser capaz de conduzi-lo para atingir as metas dos jogos, passando pelos obstáculos e desafios propostos” (OLIVEIRA, BORKS e NESTERIUK, 2017, p. 391). Assim este trabalho de conclusão de curso também é uma forma de nos introduzir aos estudos de uma nova forma de pensar a matemática, a Matemática Fuzzy e por ela, introduzimos, ainda, o conceito e práticas da inteligência artificial aplicada aos jogos digitais.

Desse contexto buscamos responder: **Quais os modelos matemáticos são explorados na construção de jogos digitais?** A partir de tal questionamento nosso objetivo geral é elencar

os conceitos matemáticos que são explorados na construção de jogos digitais. Já nossos objetivos específicos são:

1. Selecionar textos que abordem os conceitos matemáticos na construção de jogos digitais;
2. Elaborar e discutir categorias de jogos digitais;
3. Elaborar e discutir categorias dos conceitos matemáticos na construção de jogos digitais.

Nosso trabalho está dividido em cinco capítulos. O primeiro é este de introdução, que de forma suscinta apresentamos o que foi construído. No segundo, nossa fundamentação teórica, abordamos os principais conceitos do nosso trabalho. No terceiro expomos nossa metodológica, a qual estabelecemos desde a abordagem até a técnica de análise de dados. No quarto, a análise de dados, apresentamos os resultados e discussões do nosso trabalho teórico. Finalmente, no último, apresentamos as conclusões do trabalho ao respondermos a questão motivadora de nossa pesquisa.

2- FUNDAMENTAÇÃO TEÓRICA

A expressão “jogos digitais” possui diferentes definições na literatura, o que torna difícil ter um único conceito para referência. De acordo com Salen e Zimmerman (2004), os jogos digitais admitem variadas formas e aparecem em diversas plataformas computacionais. Isso inclui jogos para computadores pessoais, consoles de jogo e dispositivos móveis. Para Wolf (2008, p. 3), termos como “jogos de computador” e “jogos eletrônicos” as vezes são usados como sinônimos de videogames. De fato, durante muito tempo a expressão “jogos digitais” esteve ligada aos videogames e máquinas de fliperama, mas com o advento das tecnologias digitais este termo passou a englobar também computadores, *smartphones* e tablets.

A história dos jogos digitais, ao contrário do que se pensa, não é tão recente. Registros do *The Strong’s International Center for the History of Electronic Games® (ICHEG)* apresentam a patente de um computador projetado por Edward U. Condon da *Westinghouse* em 1940, para ser exibido na *New York World’s Fair*. A máquina criada por Condon, jogava um tradicional jogo chamado NIM, cujo objetivo dos jogadores era evitar pegar o último palito de fósforo. Alguns anos mais tarde, já em 1947, Thomas T. Goldsmith Jr. e Estle Ray Mann registram uma patente para um dispositivo de diversão CRT[1]. O jogo criado pela dupla, usava um tubo de raios catódicos conectados a um visor de osciloscópio e simulava um míssil sendo disparado em um alvo. Vários botões permitiam ajustar a curva e a velocidade do míssil. Como era inviável o desenho de gráficos eletronicamente na época, as imagens eram desenhadas e sobrepostas no CRT. Apesar de não ser considerado de fato um videogame, este é o sistema mais antigo projetado especificamente para jogos em uma tela CRT (SALEN e ZIMMERMAN, 2004).

A partir daí vários cientistas da computação começaram a utilizar jogos para testar os limites dos seus programas, até que em 1962 o estudante do Instituto de Tecnologia de Massachusetts (MIT), Steve Russel, inventa o *Spacewar!* o primeiro jogo de computador, que rapidamente se espalhou para os computadores de todo o país. Contudo, o primeiro videogame doméstico tal como conhecemos hoje, só foi definido alguns anos mais tarde, já em 1972, o *Magnavox Odyssey*, de Ralph Bear é lançado e com ele o jogo PONG, que se torna o primeiro grande sucesso do console. Desde então a popularização dos jogos digitais fez com que a indústria de jogos crescesse globalmente, tornando-se um mercado altamente lucrativo. De acordo com a firma de análise do mercado Superdata, em 2019 as companhias de games

[1] Esta tecnologia, no Brasil, ficou conhecido como monitores de tubo, um dispositivo de saída para exibição de dados, telas de tubos de raios catódicos (cathode ray tube - CRT).

movimentaram uma quantia recorde de US$ 120 bilhões, mostrando um crescimento de 4% em relação a 2018.

2.1- Inteligência Artificial Acadêmica versus Inteligência Artificial de Jogo

É interessante notar que a Inteligência Artificial (IA) Acadêmica e a IA de Jogos partem de definições diferentes. De acordo com Buckland (2005), a pesquisa acadêmica é dividida em dois campos: IA *forte* e IA *fraca.* O campo da IA *forte*, bastante controverso, considera que é possível criar uma máquina inteligente, copiando algum traço de comportamento humano, já o campo da IA *fraca* com objetivos mais realistas, tem como foco a solução de problemas do mundo real. No caso da IA aplicada a jogos o objetivo é lúdico, e portanto, tem-se o interesse numa IA que passe o teste de *Turing,* ou seja, um *player* jogando contra uma IA deve ter a sensação de estar jogando contra outro *player* e não contra uma máquina. Nesse sentido, Buckland (2005, p.21) afirma, "Com relação a IA de jogo, estou firmemente convicto de que se o jogador acredita que o agente com que ele está jogando é inteligente, então é inteligente. Simples assim. Nosso objetivo é projetar agentes que forneçam a ilusão de inteligência, nada mais".

Para entender melhor o termo "ilusão de inteligência", tomemos o jogo *PONG* como exemplo. Sua mecânica é bastante simples, e é muito fácil programá-lo usando a teoria da IA acadêmica de forma a criar um *bot*[2] invencível. Mas qual jogador gostaria de enfrentar um desafio que nunca irá vencer? Por outro lado, temos a IA aplicada a jogos, onde o *bot* precisa mostrar todo o seu potencial, mas de forma que não ganhe sempre. Isso exige a simulação do reflexo humano, tentar entender o pensamento do jogador como um bom enxadrista, antecipar as jogadas sem necessariamente recorrer a cálculos matemáticos. Porém, o cuidado aqui é em não criar algo tão simples que faça o jogador perder o interesse no jogo. Isso vai acontecer caso a IA for vista agindo de forma não convencional, como por exemplo, personagens correndo contra as paredes, não reagindo a comandos, *loops* infinitos, etc. (BUCKLAND, 2005).

Mas o que é Inteligência Artificial? De acordo com Rosa (2011, p. 3), "IA é o estudo de como fazer os computadores realizarem tarefas as quais, até o momento, os homens fazem melhor". Ou ainda, "A inteligência artificial (IA) pode ser definida como o ramo da ciência da computação que se ocupa da automação do comportamento inteligente" (LUGER, 2013, p. 1). Como podemos perceber, o conceito é bem abrangente e complexo, pois não existe consenso

[2] O diminutivo de robot (robô)

nem mesmo na definição do que seria “inteligência”. No entanto, podemos definir os sistemas de IA em quatro categorias conforme o quadro abaixo.

Quadro 1: As quatro categorias de IA e suas definições

Pensando como um Humano	“O novo e interessante esforço para fazer os computadores pensarem (...) máquinas com mentes, no sentido total e literal” (Haugeland, 1985)
Pensando racionalmente	“O estudo das computações que tornam possível perceber, raciocinar e agir.” (Winston, 1992)
Agindo como seres humanos	“O estudo de como os computadores podem fazer tarefas que hoje são melhor desempenhadas pelas pessoas.” (Rich and Knight, 1991)
Agindo racionalmente	“Inteligência Computacional é o estudo do projeto de agentes inteligentes.” (Poole *et al*., 1998)

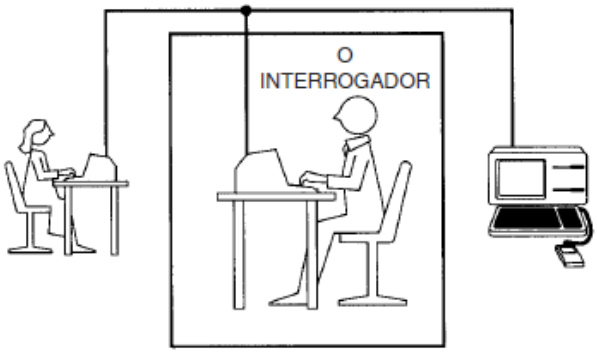
Fonte: Adaptado de Russel e Norvig (2013).

As duas categorias pensando como humano e agindo como seres humanos são práticas e envolvem a elaboração de hipóteses e a experimentação. As duas categorias pensando racionalmente e agindo racionalmente são teóricas e abrangem a Engenharia e a Matemática.

2.2- Agindo como seres humanos: O teste de Turing

De acordo com Luger (2013, p. 11) um dos primeiros trabalhos publicados sobre a inteligência de máquina especificamente em relação ao computador digital moderno, foi escrito em 1950 pelo matemático britânico Alan Turing. O artigo intitulado *Computing Machinery and Intelligence* associou o processo de pensamento humano ao computacional, onde Turing abordou se seria possível ou não fazer uma máquina pensar. Ao fazer essa reflexão, Turing notou então que haviam ambiguidades contidas em sua própria questão (o que é pensar?) e foi então que ele propôs o seu famoso teste, que ficou conhecido como o “Teste de Turing”, um exame empírico, claramente definido como meio de classificar a inteligência de uma máquina. O objetivo do teste de Turing era fazer com que uma pessoa, ao interrogar um computador não tivesse absoluta certeza de que estava conversando com um humano ou com uma máquina. (Figura 1).

Figura 1: Teste de Turing.



Fonte: Luger (2004).

O computador passaria no teste caso o interrogador não conseguisse identificar se estava conversando com um ser humano ou com uma máquina. Para um computador executar o teste de Turing, segundo Luger (2004), é preciso alguns requisitos:

- **Processamento de linguagem natural:** se comunicar no mesmo idioma do interrogador.
- **Representação do conhecimento:** armazenar o conhecimento prévio e as novas informações recebidas no interrogatório.
- **Raciocínio automatizado:** usar as informações guardadas para responder as questões e obter novas ideias.

  Aprendizado de Máquina *(Machine Learning)*: Se adaptar a novas situações e detectar normas e modelos.

> O teste de *Turing* evitou deliberadamente a interação física direta entre o interrogador e o computador porque a simulação física de uma pessoa é desnecessária para a inteligência. Entretanto, o chamado teste de *Turing* total inclui um sinal de vídeo, de forma que o interrogador possa testar as habilidades de percepção do indivíduo, além de oferecer ao interrogador a oportunidade de repassar objetos físicos [...] (RUSSEL; NORVING, 2013, p. 4, grifos do autor).

Além dos 4 requisitos já citados, para a conclusão total do teste de *Turing*, segundo Russel e Norving (2013), a máquina também precisará de:

- **Visão computacional:** Para percepção de objetos.
- **Robótica:** Para manipular os objetos e movimentar-se.

O teste de Turing permanece relevante mesmo após 70 anos. Ainda assim, alguns pesquisadores da IA acreditam que o mais importante seria compreender os princípios básicos da inteligência e não imitar um modelo. “O teste de Turing, apesar do seu apelo intuitivo, é vulnerável a várias críticas justificáveis. Uma das mais importantes se refere a uma tendência de ser direcionado para tarefas de solução de problemas fictícios” (LUGER, 2013, p. 12). De fato, para esses pesquisadores a tarefa mais importante da IA moderna seria o desenvolvimento de ferramentas para solucionar problemas práticos e específicos. Vale frisar, que até o presente momento, nenhuma máquina foi capaz de concluir um teste de Turing em sua totalidade.

2.3- Pensando como humano

A principal dificuldade ao lidar com sistemas que pensam como nós, seres humanos, é exatamente entender e determinar as características do nosso pensamento, uma vez que além de racional, o homem também é dotado de emoções, valores éticos e morais. Para desenvolver esses sistemas, os pesquisadores da área utilizam a técnica de introspecção na tentativa de

"capturar" os próprios pensamentos à medida que se desenvolvem por meio de experimentos psicológicos e só então tentar expressa-los como um programa informático. Se o *input/output*, ou seja, a inserção e o retorno de dados do programa corresponderem ao pensamento humano, podemos tomar como um indicativo de que alguns dos mecanismos deste programa também podem estar atuando nos seres humanos (RUSSEL e NORVING, 2013). A esse respeito

> [...] Allen Newell e Herbert Simon, que desenvolveram o GPS, o "Resolvedor Geral de Problemas" (do inglês "*General Problem Solver*") (Newell e Simon, 1961), não se contentaram em fazer seu programa resolver problemas de modo correto. Eles estavam mais preocupados em comparar os passos de suas etapas de raciocínio aos passos de indivíduos humanos resolvendo os mesmos problemas. (*RUSSEL; NORVING*, 2013, p.5).

Nos primórdios da IA, havia uma certa confusão por parte dos autores com relação aos campos da IA e a ciência cognitiva. Hoje há uma clara distinção, o que permitiu que as duas ciências pudessem se desenvolver com maior rapidez. No campo da ciência cognitiva há várias linhas de investigação que partem da ciência da computação e inteligência artificial, em especial no ramo das redes neurais (RUSSEL e NORVING, 2013).

2.4- Pensando Racionalmente

A lógica aristotélica, baseada nos silogismos do filósofo grego Aristóteles (384-322 a.C.), forneceram ferramentas para a análise de argumentos empregados em premissas que sempre levavam a conclusões coerentes. Baseados nesses fundamentos, lógicos do século XX criaram uma notação aprimorada para as mais variadas declarações e isso permitiu já na década de 60, que programas pudessem resolver qualquer tipo de problema lógico que fosse solucionável. A única ressalva era que caso não houvesse solução o programa entraria em um *loop* infinito em busca da resposta (RUSSEL e NORVING, 2013).

Esse tipo de abordagem enfrenta dois grandes problemas. O primeiro se concentra na dificuldade em transpor o conhecimento informal na sua amplitude em lógica formal, principalmente quando esse aprendizado não é 100% correto. Em segundo lugar, existe uma grande diferença entre ser capaz de resolver um problema em teoria e resolvê-lo na prática. Um problema aparentemente "simples" pode consumir todos os recursos computacionais de um computador caso o seu conjunto de instruções não possua a capacidade de resolver esse determinado problema. Apesar de serem obstáculos inerentes a qualquer tentativa de construir um sistema inteligente, eles surgiram primeiramente na tradição logicista (RUSSEL e NORVING, 2013).

2.5- Agindo Racionalmente

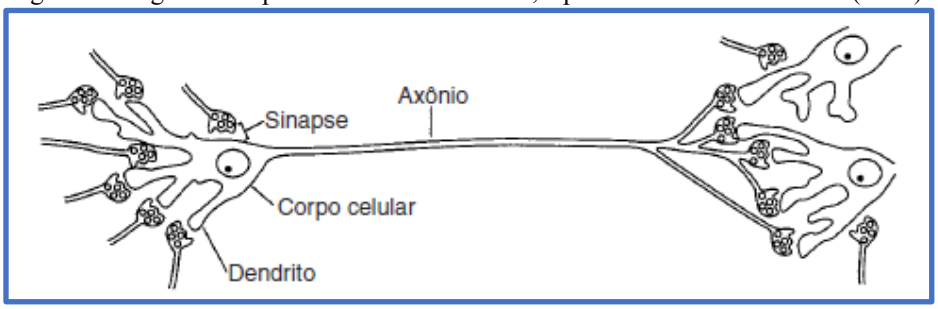
Segundo Russel e Norving (2013), um agente racional sempre irá buscar o melhor resultado possível para determinado problema, em caso de incerteza, a busca será pelo melhor resultado esperado. Na abordagem das leis do pensamento para IA, foi dada ênfase para deduções corretas, por outro lado, existem situações em que não existe uma tomada de decisão exata, mas uma decisão obrigatoriamente deve ser tomada. Assim, também há modos de agir racionalmente que não se pode dizer que envolvem inferências.

Essa abordagem tem duas vantagens em relação as outras. Primeiro ela é mais geral que as demais, podendo ter outros mecanismos além da inferência para se alcançar a racionalidade. Ela é também mais acessível ao desenvolvimento cientifico, pois o modelo de lógica é matematicamente bem definido, podendo gerar modelos de agentes que comprovadamente irão atingi-lo (RUSSEL e NORVING, 2013).

2.6- Primeiros passos da IA (1943-1950)

No início, a Inteligência Artificial foi categorizada em duas abordagens: a linha simbólica e a linha conexionista. A IA simbólica está relacionada a forma como o ser humano raciocina, um exemplo, são os sistemas especialistas e principalmente a linguagem PROLOG. Já a IA conexionista é baseada na modelagem da inteligência humana, e foi essa linha que os primeiros estudos seguiram, em especial ao modelo de neurônios artificiais proposto por Warren Mcculloch e Walter Pitts em 1943, que teve como inspiração as redes neurais biológicas do cérebro humano (RUSSEL e NORVING, 2013).

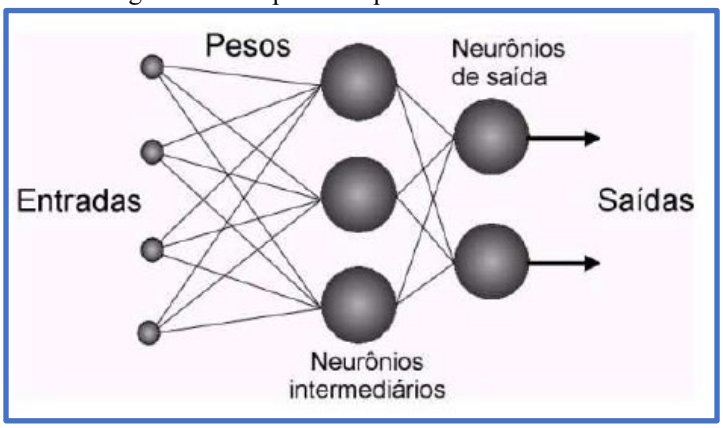
Figura 2: Diagrama simplificado de um neurônio, a partir de Crick e Asanuma (1986).



Fonte: Luger (2013)

Segundo Nakamiti (2009), as Redes Neurais Artificiais (RNA) podem ser definidas como sistemas paralelos distribuídos, formados por unidades de processamento simples intituladas neurônios artificiais, que realizam o cálculo de determinadas funções matemáticas, organizadas em uma ou múltiplas camadas e interligadas por conexões relacionadas a pesos com a função de armazenamento.

Figura 3: Exemplo de arquitetura de uma RNA.



Fonte: Nakamiti (2009).

A principal vantagem das RNA's como ferramenta computacional é a capacidade de aprender com dados conhecidos e apresentar respostas com dados desconhecidos.

2.6.1- O nascimento da IA (1951-1969)

A empolgação crescia à medida que novas descobertas eram realizadas, foi nessa época que surgiram os primeiros programas capazes de jogar xadrez criados respectivamente por Claude Shannon, 1950 e Alan Turing, 1953, provar teoremas de lógica e tentativas de imitar o raciocínio humano. Em 1951 foi construído no MIT o primeiro computador neural chamado SNARC, pela dupla Marvin Minsky e Dead Edmonds. Por suas contribuições a IA, Minsky é conhecido como o Pai da Inteligência Artificial Simbólica (RUSSEL e NORVING, 2013).

Mas foi em 1956, que Jhon McCarthy, um cientista americano doutor pela Universidade de Princeton, convenceu os pesquisadores interessados em redes neurais, Marvin Minsky, Nathaniel Rochester, Claude Shanon e outros, a se reunirem em um encontro de dois meses no Darthmouth College, NH, USA, durante o verão. Foi nessa conferência que o termo Inteligência Artificial foi utilizado de forma oficial para denominar este novo campo de estudo. "O seminário de Dartmouth não trouxe nenhuma novidade, mas apresentou uns aos outros todos os personagens importantes da história. Nos 20 anos seguintes, o campo seria dominado por essas pessoas e por seus alunos e colegas do MIT, da CMU, de Stanford e da IBM" (RUSSEL e NORVING, 2013, p. 42).

2.6.2- Os primeiros problemas (1970-1979)

Nessa fase, uma das dificuldades enfrentadas pelos pesquisadores era em relação a forma como os programas processavam a informação. Os primeiros programas não tinham conhecimento do assunto, então os resultados eram obtidos por meio de manipulações

sintáticas, assim, em quase todos os casos os sistemas falhavam quando eram apresentados programas mais complexos. Nessa época acreditava-se que para resolver "problemas maiores" seriam necessários apenas maior capacidade de armazenamento de dados e tempo de processamento, mas com o surgimento da Teoria da Complexidade Computacional de Stephen Cook e Leonid Levin em 1971, ficou comprovado que a solução desses problemas não dependia apenas de *hardwares* mais rápidos e maior armazenagem. Isso acabou de certa forma diminuindo o entusiasmo inicial (RUSSEL e NORVING, 2013, p. 42).

Durante a década de 70, a IA permaneceu quase restrita ao ambiente acadêmico, os focos de pesquisas estavam voltados a construção de teorias e desenvolvimento de programas que as verificassem.

2.6.3- A IA como indústria (1980-dias atuais)

De acordo com Nakamiti (2009), no início dos anos 80, as redes neurais passaram a receber múltiplas camadas, e com isso vão se tornando mais sofisticadas, reacendendo o interesse da comunidade cientifica. No ano de 1981, os japoneses anunciam um projeto de computador de quinta geração, que teria o PROLOG como linguagem de máquina, permitindo então a realização de milhões de cálculos por segundo. Nos anos seguintes, tanto os Estados Unidos quanto a Europa passaram a investir pesado temendo que um domínio japonês pudesse ocorrer.

> De modo geral, a indústria da IA se expandiu de alguns milhões de dólares em 1980, para bilhões de dólares em 1988, incluindo centenas de empresas construindo sistemas especialistas, sistemas de visão, robôs e softwares e hardware especializados para esses propósitos. (RUSSEL e NORVIG, 2013, p. 28).

Dessa forma a IA volta a ser uma área bastante ativa, mas agora com pesquisas no campo da visão, manufatura, robótica, computação quântica, nanotecnologia, etc..

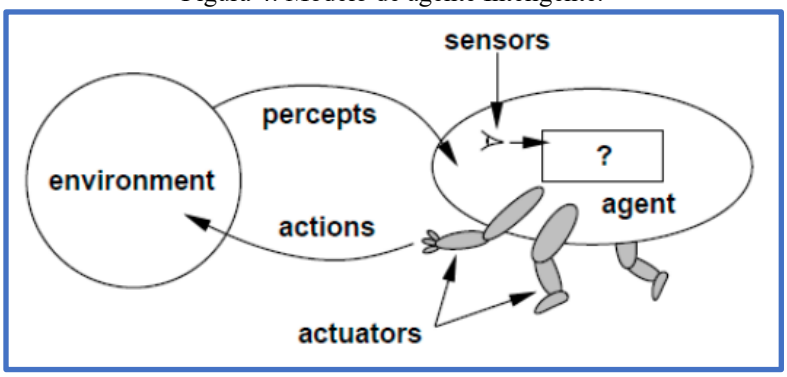
2.6.4- Agentes inteligentes

O conceito de "agentes inteligentes" surgiu da evolução de pesquisas na área de Inteligência Artificial. De acordo com Minsk (1994, apud ARAÚJO, 2004) a inserção de agentes para realização de tarefas computacionais foi idealizada pelos pesquisadores Nicholas Negroponte e Alan Kay. O objetivo primário era aumentar o nível de comunicação entre o homem e o computador, de forma a aproximar ao máximo a expressão da máquina com a forma humana. O início de estudo sobre agentes inteligentes iniciou-se em meados da década de 80. Com a evolução dos computadores, que passaram a ter processadores cada vez mais rápidos e com a explosão da internet nos anos 90, o estudo dos agentes inteligentes se estabeleceu de vez.

As teorias e modelos de agentes inteligentes se diferenciam dos demais por apresentarem um maior nível de subjetividade e simbolismo, remetendo a IA simbólica, ou seja, parte-se do princípio de que a inteligência de alguma forma já deve estar presente nos objetos autônomos. O termo "agente" encontra-se presente nas mais diversas áreas do conhecimento (sociologia, economia, robótica, biologia, computação, etc.), mas existe uma característica que unifica todos os significados, embora distintos, que é a existência de um espaço no qual os objetos autônomos interagem por meio da troca de informações e conhecimentos, demonstrando uma capacidade elevada em executar tarefas, das mais simples as mais complexa (RUSSEL e NORVIG, 2013).

Para os referidos autores, um agente é qualquer coisa com a capacidade de identificar o ambiente por meio de sensores e operar nesse ambiente por meio de atuadores (figura 4). Ou ainda, segundo Barreto (2001, apud DAMIÃO et al, 2014, p. 4), "um agente é um sistema dinâmico com capacidade de receber informações e agir sobre um ambiente objetivando realizar uma determinada tarefa".

Figura 4: Modelo de agente Inteligente.



Fonte: Russel e Norvig (2013).

A utilização dos agentes se dá nos mais variados tipos de ambientes, sempre com o objetivo de facilitar a realização de atividades que muitas vezes se tornam repetitivas. "O trabalho da IA é projetar o programa de agente que implementa a função do agente – que mapeia percepções em ações" (RUSSEL e NORVIG, 2013, p. 75). O programa do agente é uma implantação de uma função do agente (histórico de tudo o que um agente percebeu até o momento) para uma determinada ação.

$$Agente = Arquitetura + Programa\ de\ Agente$$

Segundo Russel e Norvig (2013), como exemplos de agentes, podemos citar:

- agentes de *software*, que são *softwares* programados por usuários;

- agentes robóticos, utilizam câmeras e sensores infravermelhos atuando como sensores e motores como atuadores;

agente humano, que possui olhos, ouvidos e outros órgãos que atuam como sensores e mãos, pernas e demais partes do corpo que desempenham a função de atuadores.

Levando para o campo da computação, podemos interpretar o "agente inteligente" como qualquer coisa que se utiliza de conhecimento, o qual toma decisões e realiza ações após considerar o seu melhor resultado (RUSSEL e NORVIG, 2013).

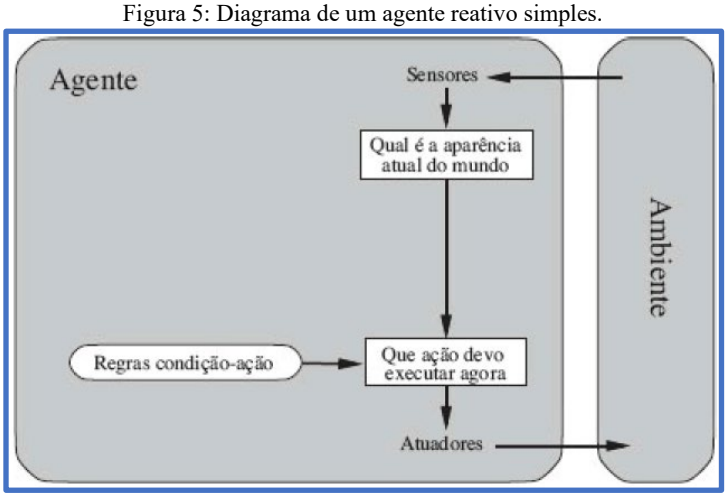
2.6.5- Tipos de Agentes Inteligentes

Podemos agrupar os agentes em quatro classes com base no seu grau de inteligência, percepção e capacidade.

2.6.5.1- Agentes Reativos Simples

São agentes que operam apenas com base na percepção atual, ignorando completamente acontecimentos anteriores. A função do agente é baseada numa espécie de estrutura de condição, onde a condição é verificada e caso a condição for satisfeita (V), então a ação é executada, caso contrário (F), então a ação não será executada. Essa função do agente só terá êxito caso o ambiente seja totalmente observável, caso o ambiente seja parcialmente observável, problemas como *loops* infinitos se tornarão inevitáveis (RUSSEL E NORVIG, 2013).

Figura 5: Diagrama de um agente reativo simples.



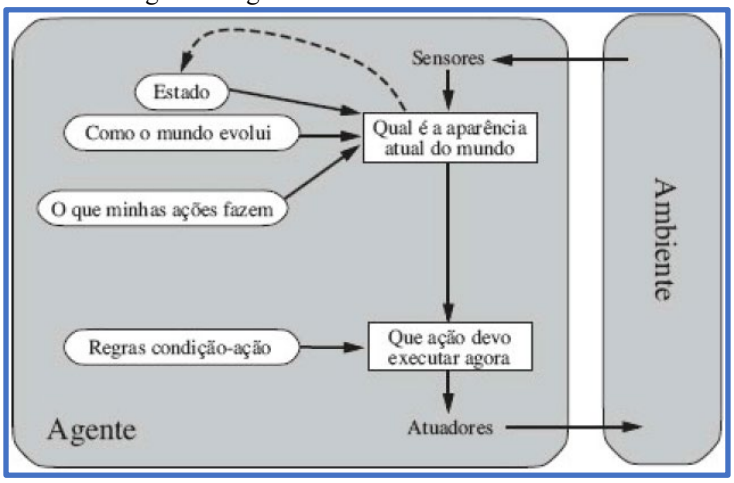
Fonte: Russel e Norvig (2013).

Os principais problemas com agentes reativos simples são: Inteligência limitada, a falta de conhecimento das partes não perceptivas e caso o ambiente tenha alguma mudança, o conjunto de regras obrigatoriamente deve ser atualizado.

2.6.5.2- Agentes reativos baseados em modelos

Para que esse modo funcione corretamente é necessário encontrar uma regra cuja condição corresponda ao estado atual. Este tipo de agente pode ligar com ambientes parcialmente observáveis desde que usem um modelo sobre o mundo.

Figura 6: Agente reativo baseado em modelo.



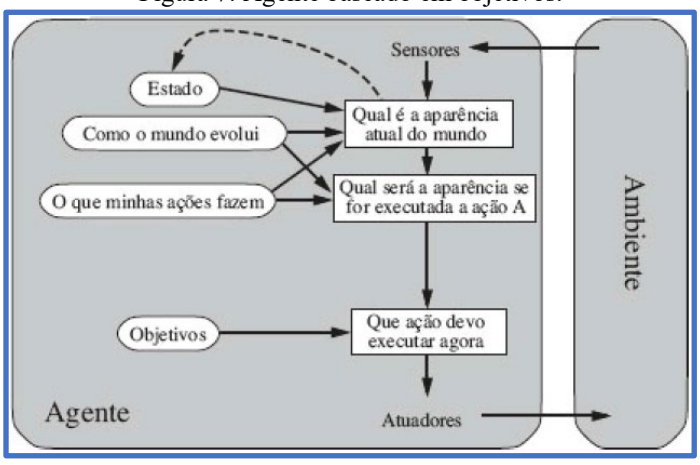
Fonte: Russel e Norvig (2013).

De acordo com Russel e Norvig (2013), um agente reativo baseado em modelo deve manter algum tipo de estado interno, com intuito de obter informações de acontecimentos anteriores e analisar alguns aspectos que ainda não tenham ocorrido no estado atual, como por exemplo, a forma como o mundo evolui autonomamente sem a interferência do agente e como suas ações interferem no mundo.

2.6.5.3- Agentes baseados em objetivos

São agentes que tomam decisões com base em quão longe estão do seu objetivo, são capazes de considerar ações em ralação ao futuro com a meta de alcança-los.

Figura 7: Agente baseado em objetivos.



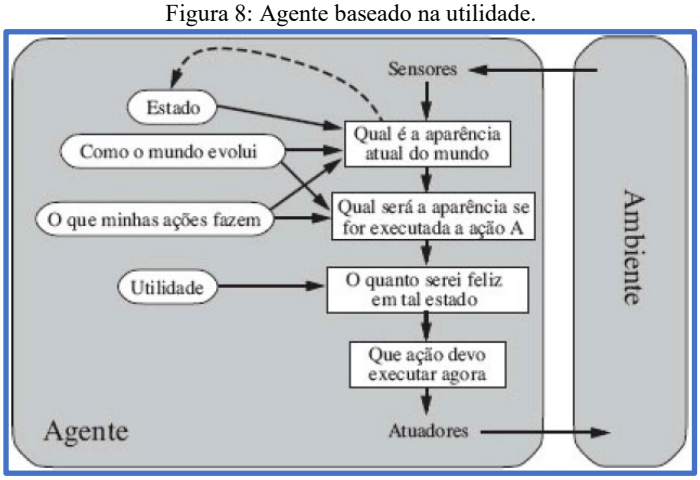
Fonte: Russel e Norvig (2013).

Para isso utilizam-se de algoritmos de buscas com o intuito de encontrar sequências de ações que auxiliem nas tomadas de decisões. Como todas as suas ações visam diminuir a distância para alcançar o objetivo, isso permite ao agente escolher entre múltiplas possibilidades, selecionando a mais indicada. Em geral são agentes flexíveis, que exigem pesquisa e planejamento. Porém, isso não significa necessariamente que o objetivo irá garantir o melhor comportamento para o agente, apenas que os estados objetivos serão diferentes dos não objetivos.

2.6.5.4- Agentes baseados na utilidade

Quando há várias alternativas possíveis, para decidir qual é a melhor, são usados os agentes baseados na utilidade. Eles escolhem as ações baseados na utilidade para cada estado.

Figura 8: Agente baseado na utilidade.



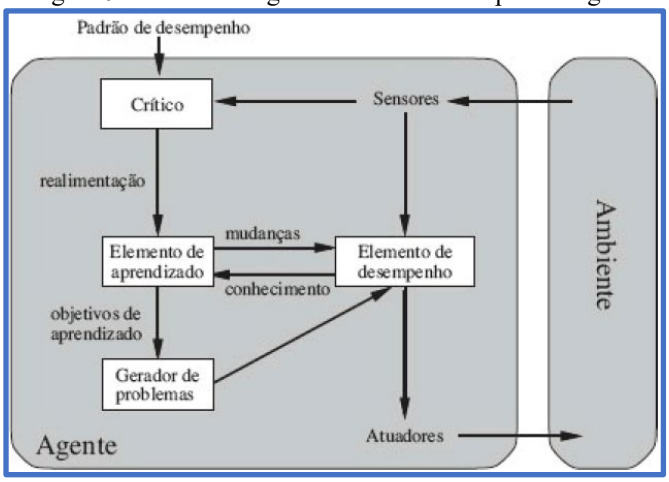
Fonte: Russel e Norvig (2013).

Russel e Norvig (2013) explicam que esse tipo de agente utilizam um modelo do mundo em companhia com uma função utilidade que tem por objetivo verificar suas preferências entre os estados do mundo.

2.6.5.5- Agente baseado em Aprendizagem

Esse tipo de agente pode aprender com as suas experiências anteriores, operar em um ambiente inexplorado e através do aprendizado conseguir interpretar e atuar neste ambiente de forma eficiente.

Figura 9: Modelo de agentes baseados em aprendizagem.



Fonte: Russel e Norvig (2013).

São baseados na máquina de Turing e conceitualmente tem quatro componentes principais, que são:

- Elemento de aprendizagem: realiza melhorias aprendendo com o ambiente.
- Crítico: responsável por enviar o feedback ao elemento de aprendizagem.
- Elemento de desempenho: realiza a tarefa de selecionar ações externas.
- Gerador de problemas: apresenta ações que o levarão a novas experiências.

2.7- Aplicações em Jogos

Como citado anteriormente a IA para jogos é diferente da IA acadêmica. O objetivo da IA na maioria dos jogos é fornecer aos jogadores, oponentes que lhes proporcionem um certo desafio e que ao mesmo tempo amplifique a habilidade e inteligência do jogador. Esse ambiente competidor, onde os agentes estão em confrontos, dando origem a um cenário competitivo é conhecido como jogo (RUSSEL e NORVING, 2013).

Nos primeiros jogos eletrônicos, a maioria dos programadores utilizavam movimentos padronizados, em sua maioria repetitivos ou aleatórios, essa limitação era causada principalmente por que os computadores da época não tinham memória e processamento suficientes. Essa programação de IA era mais conhecida como "*gameplay programming*" porque na realidade não havia nenhuma inteligência nos personagens controlados pela CPU (Central Process Unit, ou Unidade Central de Processamento). Nessa primeira tentativa de "inteligência" os personagens NPCs (non-player character ou personagem não jogável) eram controlados basicamente por meio de regras implementadas por meio de estruturas condicionais do tipo *if / then / else* (se/então/senão) em seu código. Um dos principais problemas dessa

abordagem era a falta de flexibilidade para implementar novas funções caso fosse necessário (SCHWAB, 2009).

Com a evolução computacional e um maior nível de exigência da indústria em relação a jogabilidade, novas técnica de modelagem passaram a ser utilizadas, sobretudo, aquelas que pudessem permitir aos personagens um nível maior de inteligência nas tomadas de decisões, aumentando assim a competividade dos jogos. A partir da década de 90, os desenvolvedores passaram a utilizar cada vez mais a IA nos jogos, impulsionados principalmente pelo sucesso de jogos RTS como *Dune II* , *Civilization, Age of Empires, Starcraf* e *Warcraft III*. Esses jogos ajudaram a expandir e a estabelecer a IA de vez nos jogos (SCHWAB, 2009).

Quadro 2: Linha do tempo dos jogos com IA.

Sem IA	*SpaceWar!* O primeiro jogo de computador é lançado	1962
	O jogo *PONG* é lançado simultaneamente ao *Magnavox Odyssey,* o primeiro videogame da história.	1972
Algoritmos de I,A. determinísticos e padrões de movimento	Os jogos *Pursuit* e *Qwak!* trazem ao jogador a experiência de atirar em um alvo em movimento.	1974
	Gun Fight o primeiro videogame a usar um microprocessador e a simular um combate entre humanos.	1975
	Space Invaders foi um dos primeiros jogos de tiro com gráfico bidimensional, os inimigos tinham movimentos padronizados, mas também eram capazes de revidar ataques. Foi um dos jogos responsáveis por expandir a indústria dos games a nível mundial.	1978
	Galaxian se utilizou da mesma fórmula de *Space Invaders*, mas com movimentos um pouco mais complexos.	1979
	Pac-Man apesar de apresentar movimentos padronizados, introduziu uma espécie de "personalidade" para cada um dos fantasmas no modo caçador.	1980
	Um computador vence um GM de xadrez pela primeira vez.	1983
	Karate Champ é lançado, um dos primeiros jogos de luta um contra um, primeiro contra a CPU, algum tempo depois foi lançado para dois jogadores.	1984
FSM – Máquinas de Estado Finita	Primeiro jogo de *RTS* é lançado – *Herzog Wei.* O conceito de turnos, muito presente nos jogos de tabuleiro dá espaço para ao conceito de estratégia em tempo real com ações simultâneas dos jogadores.	1990
	Lançamento de *Doom* – jogo de tiro em primeira pessoa que foi um enorme sucesso popularizando o gênero FPS, atualmente um dos gêneros mais jogados.	1993
Técnicas Diversas	*BattleCruiser: 3000AD* – O primeiro jogo comercial a usar redes neurais	1996
	O *Deep Blue,* um supercomputador criado pela IBM, derrota o então campeão mundial de Xadrez, *Gary Kasparov* considerado o maior enxadrista de todos os tempos.	1997
	Half-Life é lançado utilizando uma IA baseada em script e é considerado como o melhor jogo com IA da época.	1998
	Black & White é lançado, o jogo utiliza redes neurais, aprendizagem por observação e aprendizagem por reforço.	2001

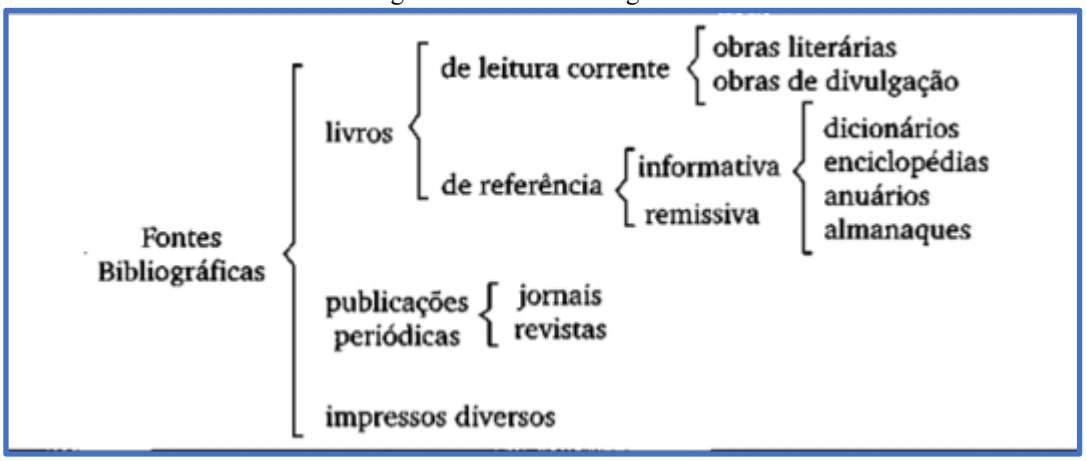
Fonte: Schwab (2009).

Nesse percurso histórico, dentre as diversas técnicas utilizadas pelos desenvolvedores, podemos citar o uso das árvores de decisão, regras de produção, máquinas de estado finitas (FSMs) e máquinas de estado *fuzzy* (FuSMs), sendo as duas últimas as mais utilizadas atualmente (SCHWAB, 2009).

3- METODOLOGIA

Nosso trabalho não tem uma análise estatística, por isso, classificamos sua abordagem como qualitativa. Para Kripka, Scheller e Bonotto (2015, p. 57) os "[...] estudos qualitativos se caracterizam como aqueles que buscam compreender um fenômeno em seu ambiente natural, onde esses ocorrem e do qual faz parte". Devido ao isolamento social por causa da Covid-19, bem como a complexidade fomos forçados a trabalhar na perspectiva da pesquisa bibliográfica, a qual segundo "[...] é desenvolvida com base em material já elaborado, constituído principalmente de livros e artigos científicos. [...] As fontes bibliográficas são em grande número e podem ser [...] classificadas" (Gil, 2002, p. 44) como apresentamos na figura:

Figura 10: Fontes bibliográficas.



Fonte: Gil (2002, p. 44).

Para este autor os "livros constituem as fontes bibliográficas por excelência" (Gil, 2012, p. 44). Ele explica que os livros se subdividem em livros que proporcionam leitura corrente ou de referência. O primeiro refere-se as obras literárias, de divulgação científica, didáticos entre outros. O segundo são os livros de consulta como, por exemplo, enciclopédias, dicionários, almanaques, entre outros. No escopo deste trabalho trabalhamos com os livros de leitura corrente da produção de jogos digitais. Uma produção textual destinada a um grupo especifico da sociedade, que são os desenvolvedores de jogos digitais.

Desse ponto de vista, para a realização deste estudo, foi utilizado como fonte primária de pesquisa, o livro *Programming Game AI by Example* de Mat Buckland, (BUCKLAND, 2005). A fonte selecionada serviu como base de investigação da educação matemática e dos elementos que caracterizam um jogo com IA. E, também, o livro *Matemática Aplicada aos Games: uma abordagem teórica e prática para desenvolvedores*, (TONÉIS, 2015). Essa fonte serviu para auxiliar, comparar e discutir os modelos matemáticos e métodos numéricos no desenvolvimento dos jogos digitais.

Para Tonéis (2015, p. 13) “Os modelos matemáticos são compostos por fórmulas, os teoremas, as teorias matemáticas que estão registradas na cultura acadêmica e escolar enquanto que os modelos computacionais formam as estruturas ligadas a máquina como o algoritmo e a lógica utilizada na programação (lógica booleana; lógica fuzzy; etc)”. Para Biembengut (1999, p. 20) um modelo matemático “[...] é um conjunto de símbolos e relações matemáticas que traduzem, de alguma forma, um fenômeno em questão”. Desses dizeres nossos dados, a ser investigados nas fontes primarias foram os modelos matemáticos que, ao longo da história dos jogos digitais, proporcionaram formas de resolver os problemas computacionais ou seria traduzir o fenômeno real para um fenômeno computacional. Mas qual sobre qual fenômeno real estamos a falar? Nosso trabalho focou em dois fenômenos reais: o movimento (aleatórios, perseguição e evasão); e os encontros (detecção de colisões). Segundo Tecmundo (2008, p. 1):

> Um efeito extremamente básico, mas muito importante nesse tipo de simulação é a detecção de colisões, que traduz em termos computacionais uma das verdades mais básicas da realidade: dois corpos não podem ocupar o mesmo lugar no espaço em um determinado instante. Essencialmente, quando um corpo tenta entrar no espaço delimitado pelas extremidades de outro corpo, ambos entram em contato e um impede o movimento do outro (TECMUNDO, 2008, p. 1).

Os dizeres nos apresenta a importância da detecção de colisões e movimento nas simulações computacionais, um dos tipos desse é o jogo digital. Mas também nos mostra que devemos nos atentar a relação entre movimento e detecção de colisões. Desse ponto de vista, procuramos obter dados que representasse o desenvolvimento dos modelos matemáticos ao longo da história da criação de jogos digitais, o quais no Brasil é relativamente novo, pois a formação específica para a produção de jogos digitais só começa em 2003 com o curso de Tecnologia em Jogos Digita oferecido pela Universidade Anhembi Morumbi (TONÉIS, 2015).

Ao obter os dados, nossa preocupação passou a ser com qual técnica iríamos analisar tais dados. Escolhemos Análise Textual Discursiva (ATD) de Moraes e Galiazzi (2011). Para Morais a ATD é

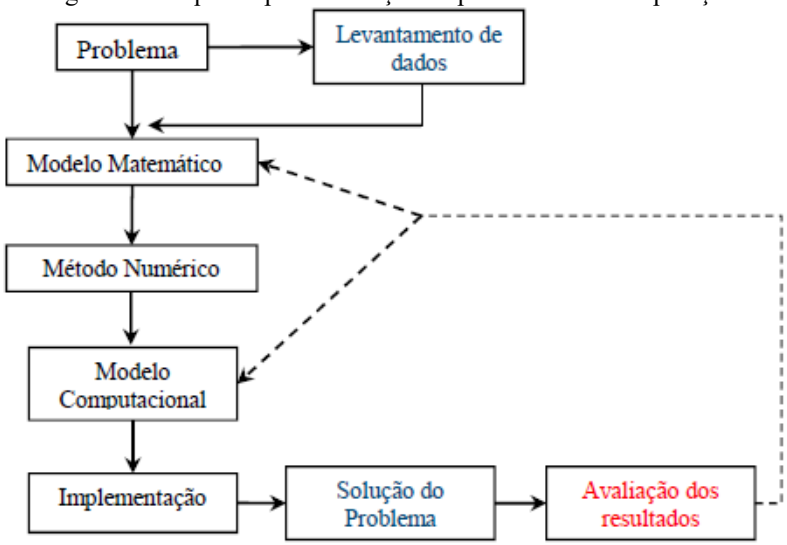
> [...] um processo auto-organizado de construção de compreensão em que novos entendimentos emergem de uma seqüência recursiva de três componentes: desconstrução dos textos do *corpus,* a *unitarização*; estabelecimento de relações entre os elementos unitários, a categorização; o captar do novo emergente em que a nova compreensão é comunicada e validada. Esse processo em seu todo pode ser comparado com *uma tempestade de luz*. O processo analítico consiste em criar as condições de formação dessa tempestade em que, emergindo do meio caótico e desordenado, formam-se *flashes* fugazes de raios de luz iluminando os fenômenos investigados, que possibilitam, por meio de um esforço de comunicação intenso, expressar novas compreensões atingidas ao longo da análise (MORAIS, 2003, p. 192, grifos do autor).

Nesse processo analítico cíclico, destacamos no processo de Desconstrução e unitarização os elementos constituintes dos textos que nos serviram de base de dados (*corpus*). Como já dissemos, os modelos matemáticos de movimento e detecção de colisões na historicidade dos jogos digitais (unidades de significado ou unidades de sentido). Desse contexto construímos a categorização dos jogos digitais de terem ou não uma Inteligência Artificial, a qual é o novo emergente da criação deles. Assim, buscamos por esse processo, na validação da construção da análise captar e expressar o movimento de produção de jogos digitais com e sem Inteligência Artificial.

4- ANÁLISE DE DADOS

Antes de criar um jogo, é necessário todo um planejamento para o seu desenvolvimento. Um dos primeiros itens em que se deve pensar é na sua mecânica, isto é, se ele será um jogo de ação, esporte, corrida, etc.. Nessa etapa surgirão alguns problemas que devem ser pensados e discutidos, como por exemplo, o tipo de movimentação que o agente (personagem) e a colisão entre esses agentes. Esses problemas poderão ser representados através de um fluxograma, Figura 12, para a resolução de problemas por meio do computador (TONÉIS, 2015).

Figura 11: Esquema para resolução de problemas de computação.



Fonte: Tonéis (2015).

Realizado o levantamento de dados, parte-se então para a construção do modelo matemático, o objetivo aqui é fornecer um modelo que solucione os problemas, nesse sentido Tonéis (2015) afirma que

> Então a Matemática aplicada aos Jogos Digitais procura fornecer modelos matemáticos que possam ampliar as soluções para problemas envolvidos na produção de jogos digitais – aspectos de *design* e da programação – com o auxílio também dos modelos computacionais. (TONÉIS, 2015, p. 16, grifos do autor).

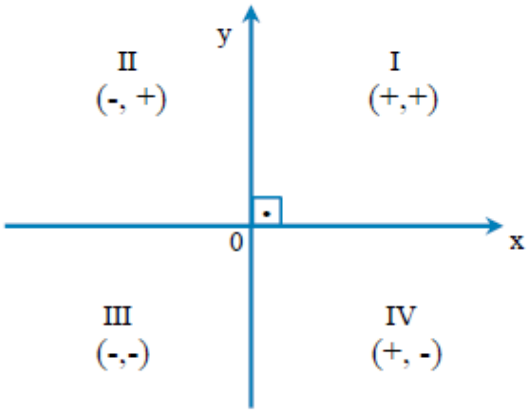
Desse contexto, sendo os principais problemas o movimento do agente e sua colisão, buscamos modelos matemáticos que responda aos problemas. Mas para tal é necessário explicar, de forma suscinta, a utilização do sistema de coordenadas no desenvolvimento dos jogos digitais.

O plano cartesiano[3] é composto por duas retas bases chamadas de eixos. A reta x chamamos de eixo das abscissas e a reta y de eixo das ordenadas. Os dois eixos dividem o plano em quatro partes (quadrantes), que são determinados pelo sentido anti-horário do relógio. Um

[3] Sistema cartesiano ou plano ortogonal cartesiano mostra a localização de um ponto em um plano e recebeu esse nome em homenagem a Descartes (do latin *Cartesius*).

ponto P(x, y) possui coordenadas (x, y), sendo x a abscissa e y o elemento da ordenada. O sinal de cada eixo no plano cartesiano é determinado de acordo com o quadrante, conforme a figura 13. (TONÉIS, 2015).

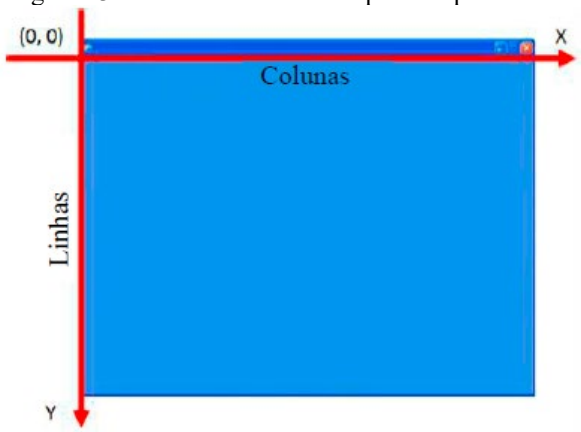
Figura 12: Plano ortogonal cartesiano com seus quadrantes e sinais.



Fonte: Tonéis (2015).

Quando transportamos o plano cartesiano para a programação, o sistema na tela é um pouco diferente, principalmente em relação a direção na qual o eixo das ordenadas cresce, enquanto que no plano cartesiano o valor cresce para cima, no computador o sentido cresce para baixo, Figura 14, ou seja, o eixo das ordenadas são invertidos (TONÉIS, 2015).

Figura 13:Modelo da tela do computador por default.



Fonte: Tonéis (2015).

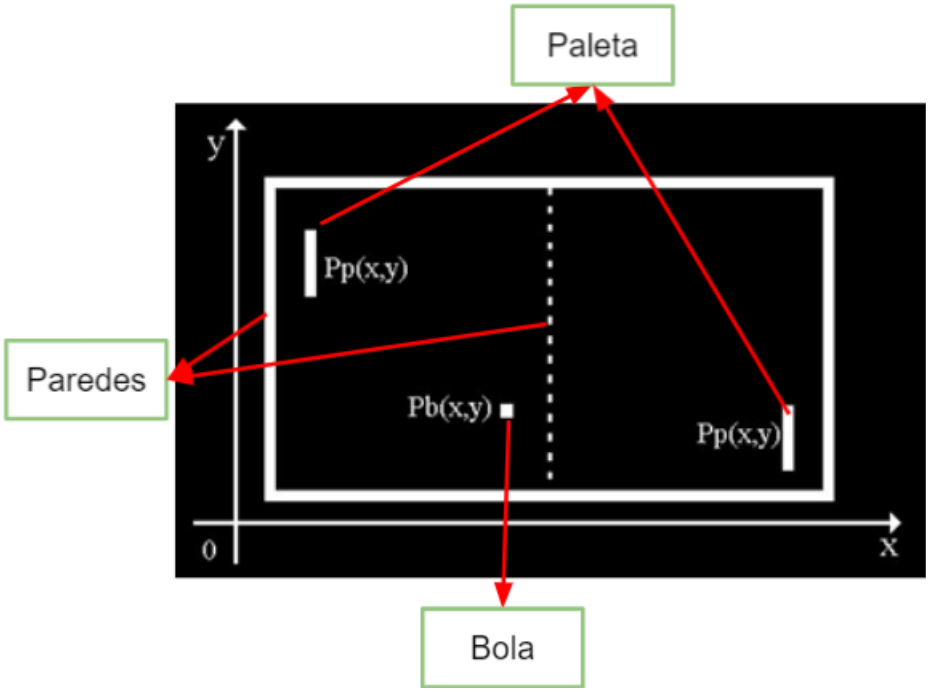
Por *default* (padrão) o monitor do computador possui apenas o quadrante positivo (+, +) e para interpretar as coordenadas, o computador parte da origem (0, 0) e avança para o primeiro quadrante até atingir o limite total da tela. Por exemplo, imagine que o tamanho da nossa tela é 9x9, então tudo que estiver entre o ponto (0, 0) e (9, 9) será visível, no entanto qualquer elemento fora desta área, não será visível, mesmo que esteja no primeiro quadrante (TONÉIS, 2015).

Outra diferença é que o plano cartesiano é composto por retas que pertencem aos números reais (R^2) e assim formam infinitos pontos, mas a tela do computador é limitada ao seu tamanho. Quando trabalhamos com definição gráfico-computacional, utilizamos o sistema métrico *pixel,* se uma tela tem a resolução de 1080*p* (1920x1080), isso implica dizer que ela tem uma resolução de 1920 *pixels* na horizontal e 1080 pixels na vertical, ou seja, sua resolução é de 2.073.600 *pixels* no total (TONÉIS, 2015).

4.1- Jogos Digitais Sem Inteligência Artificial

Os primeiros jogos lançados comercialmente no início da década de 70 seguiam basicamente o mesmo padrão em relação a programação e a linguagem gráfica, os jogos eram desenvolvidos basicamente por programadores e devido a limitação técnica da época os gráficos eram postos em segundo plano. A mecânica do jogo PONG[4], por exemplo, permite que o jogador controle uma barra vertical (paleta) movendo-a verticalmente com o objetivo de acertar a bola, quando esta acerta a paleta ela recebe uma nova direção de acordo com a distância máxima do centro da paleta ao local da colisão. Perceba que em relação a representação do plano e espaço, o jogo PONG utiliza-se do modelo de plano cartesiano. Na Figura 15 temos a definição das posições iniciais e finais, em coordenadas cartesianas (x, y) de cada elemento.

Figura 14: Desenho do jogo.



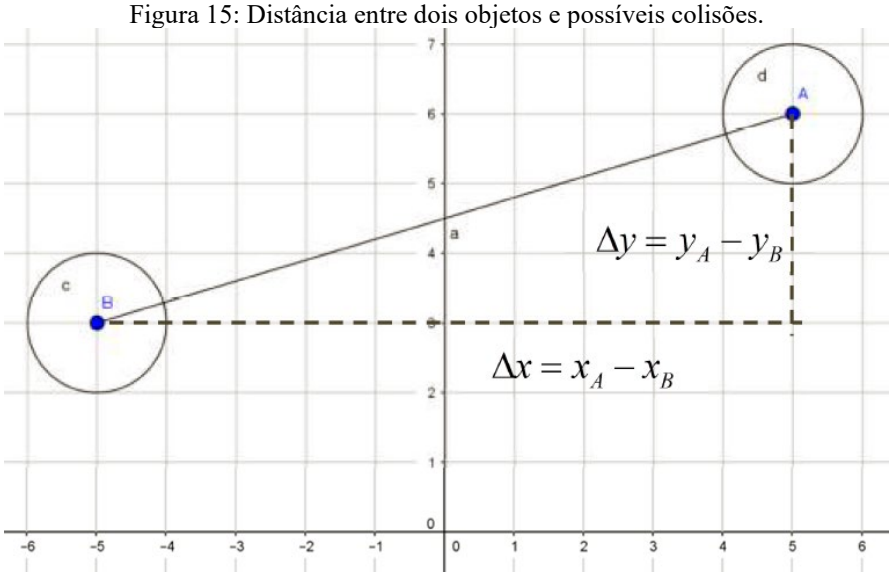
Fonte: Adaptado de Harris; Gorasia (2008).

[4] Pong é um jogo digital que simula um jogo de tênis de mesa, com gráficos bidimensionais simples, fabricado pela Atari e originalmente lançado em 1972, uns dos primeiros jogos digitais, criado por Allan Alcorn como um exercício de treinamento (HARRIS; GORAS, 2008).

Uma vez que temos o desenho do jogo, precisamos definir como será realizado o controle de movimento da paleta, e isso nos leva aos seguintes problemas: Como calcular a colisão entre a bola e a paleta? De que forma ocorre a colisão bola-parede? Qual ângulo devo utilizar para rebater a bola na direção do ponto P(x, y)?

Nos jogos digitais uma das formas que podemos calcular colisões é por meio da distância de dois pontos no plano cartesiano, conforme ilustrado na figura 15.

Figura 15: Distância entre dois objetos e possíveis colisões.



Fonte: Tonéis (2015).

A ideia é que se caso a distância entre os dois objetos seja maior que o raio A + raio B, então não haverá colisão, mas caso a distância seja menor ou igual a soma dos raios, então haverá colisão (TONÉIS, 2015). O modelo matemático é dado pela aplicação do Teorema de Pitágoras: “Em qualquer triângulo retângulo, a área do quadrado cujo lado é a hipotenusa é igual à soma das áreas dos quadrados que tem como lados cada um dos catetos” (LIMA et al., 2006, p.64), fazendo uso dele para o cálculo da distância (d) entre dois pontos, temos:

$$d^2 = (\Delta x)^2 + (\Delta y)^2$$

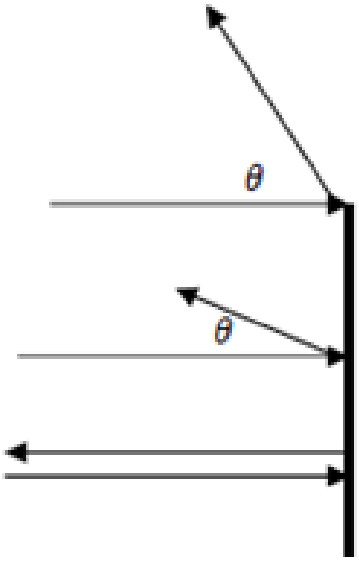
$$d = \sqrt{(\Delta x)^2 + (\Delta y)^2}$$

$$d = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}$$

Em que $\mathbf{x_A e\ y_A}$ são as coordenadas do ponto A e $\mathbf{x_B e\ y_B}$ as coordenadas do ponto B. Assim se d > raio A + raio B não há colisão e se d ≤ raio A + raio B há colisão.

Em relação a paleta, caso a bola atinja o seu centro ela será lançada exatamente em um ângulo de 90º, se ela atinge outra região a paleta muda o ângulo de reflexão. O tamanho do ângulo é proporcional à distância do centro aos extremos da paleta a ação descrita é demonstrado na Figura 16.

Figura 16: Progressão de incidência do ângulo de colisão



Fonte: Harris e Gorasia (2008)

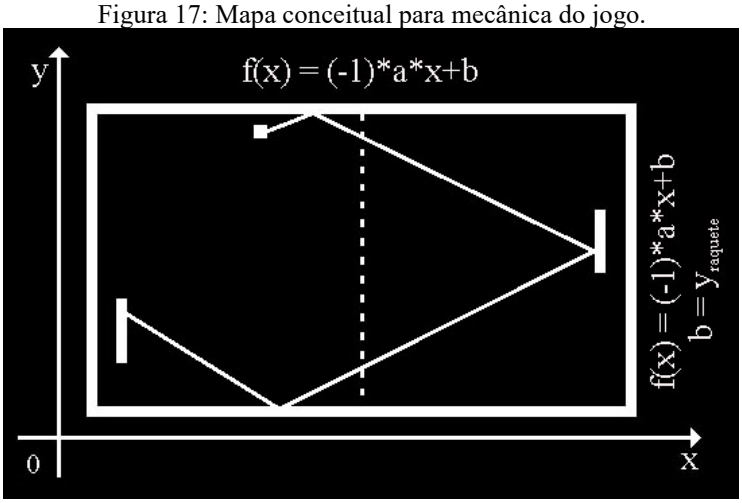
Para mudar o ângulo de reflexão dependendo de onde ocorre a colisão com a paleta, é necessário calcular a posição da bola em relação ao centro da paleta, caso a bola acerte a paleta, basta inverter o valor de y. No caso da colisão entre bola e parede, leva-se em consideração o sentido da bola e resulta em decremento das coordenadas y da bola ao colidir com a parede superior e incremento ao colidir com a parede inferior.

Uma outra forma de recriar essa mecânica é através da equação da reta. Vamos assumir o plano cartesiano com (0,0) na base da rede, conforme apresentado na Figura 18. Dessa forma é possível generalizar algumas funções que podem ser implementadas no jogo conforme exemplificado por Tonéis (2015).

- Definir a largura (y) e o comprimento que será dado pelo intervalo de x;
- As raquetes b = y;
- O coeficiente *a* é alterado nas bordas invertendo o sinal, assim $a = a*(-1)$;
- A mecânica tem como base o desenho de losangos, conforme podemos observar na figura, eles variam de acordo com a posição em que a bola bate na raquete e nas bordas.

Para posicionar a bola utiliza-se a função inicial $f(x) = ax + b$

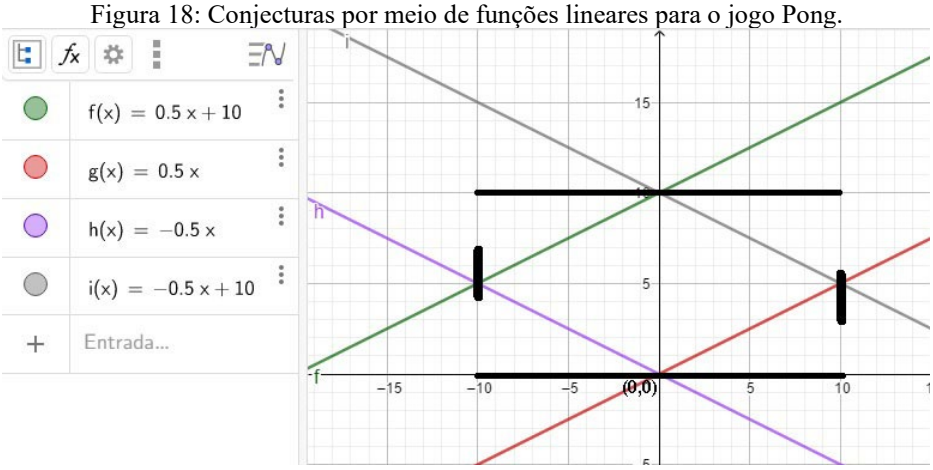
Figura 17: Mapa conceitual para mecânica do jogo.



Fonte: Adaptado de Tonéis (2015).

Utilizando o programa GeoGebra é possível construir diferentes gráficos de funções lineares e assim representar as alterações no gráfico a partir da taxa de variação (coeficiente angular).

Figura 18: Conjecturas por meio de funções lineares para o jogo Pong.



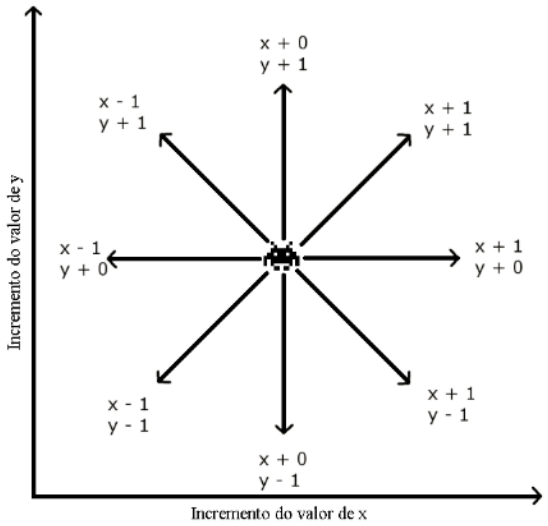
Fonte: Adaptado de Tonéis (2015).

Exemplos como esses podem ser aplicados a diversos jogos que exploram esse tipo de mecânica. No entanto, existem várias outras funções ou recursos que podemos utilizar para transformar o *level design* e as mecânicas de um jogo, e para isso conhecer a Matemática é essencial (TONÉIS, 2015).

Dentro da programação de jogos, funções trigonométricas como seno e cosseno são utilizadas para as mais variadas tarefas. Não há limites para a sua utilização e entender a matemática contida na função e os resultados retornados por essa função, pode reduzir significativamente os esforços na resolução de um determinado problema. Pegando o jogo

Space Invaders[5], um jogo de tiro com gráfico bidimensional cujo objetivo principal era destruir as naves alienígenas com uma espaçonave humana e fazer o maior número de pontos possíveis, como exemplo, podemos utilizar as funções trigonométricas para determinar a velocidade da nave com base no ângulo em que o agente está se movendo, girar, mover para cima, baixo, direita, esquerda ou diagonal. Suponha, por exemplo, que temos um sistema de coordenadas cartesianas onde o valor de y recebe 1 de incremento à medida que a nossa nave avança para cima e o valor de x recebe 1 de incremento medida que nos deslocamos para a direita. Por conseguinte, se quisermos mover para a esquerda, basta decrementar 1 em x e assim por diante. Como podemos observar na Figura 20, também é possível realizar movimentos diagonais, se quisermos ir 45º para cima e para a direita, basta incrementar 1 a x e também a y, se pretendemos ir para cima e para a esquerda, basta decrementar 1 de x e incrementar 1 em y (HORTON, 2016).

Figura 19: Diagrama de direção da nave espacial em *Space Invaders.*



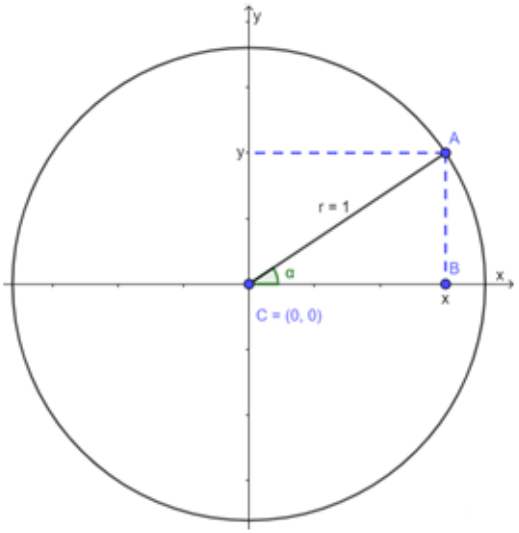
Fonte: Adaptado de Horton (2016).

A partir da Figura 20 podemos pensar o movimento do agente em um movimento circular, dessa forma o modelo matemático. Para isto Tonéis (2015) afirma que podemos ter as seguintes situações:

[5] O filme *Star Wars,* do diretor norte-americano George Lucas, lançado um ano antes e que se tornou um sucesso de bilheteria e um fenômeno da cultura pop e *The War of the Worlds,* célebre romance de ficção científica do escritor britânico H. G. Wells serviram de inspiração para Nishikado construir o jogo.

1- “Quando o centro da circunferência coincide com a origem do sistema (0, 0) e o raio é unitário (1) [...]” (TONÉIS, 2015, p. 134), a Figura 2**0** ilustra tais dizeres:

Figura 20: Direções da nave espacial em *Space Invaders*.



Fonte: Elaborada pelo autor.

Nessa figura o ponto C (0, 0) representaria o jogador, o ponto A (x, y) o inimigo e α é a medida da rotação, em radianos, a partir da origem do sistema cartesiano C (0, 0) no sentido anti-horário (TONÉIS, 2015). Desse ponto de vista o modelo matemático para movimento de A seria:

Para movimento em y: $\operatorname{sen}(\alpha) = \frac{y}{1}$

$y = \operatorname{sen}(\alpha)$

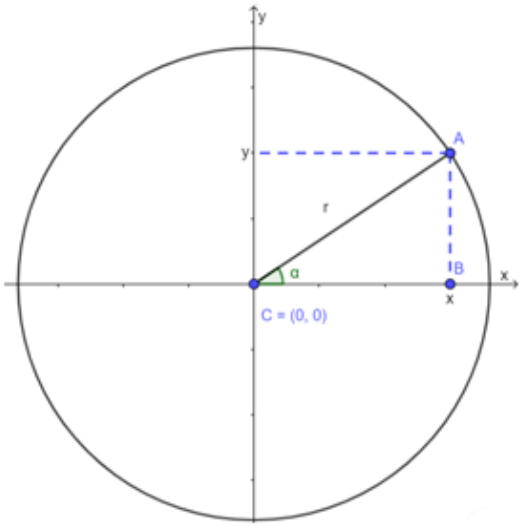
Para movimento em x: $\cos(\alpha) = \frac{x}{1}$

$x = \cos(\alpha)$

Dessa forma temos o seguinte ponto: A (sen (α), cos (α)). Deve-se considerar, também, segundo Tonéis (2015, 134):

2- “Quando o centro da circunferência coincide com a origem do sistema (0, 0) e o raio r qualquer (sendo r≠0)”, a Figura 21 ilustra tais dizeres:

Figura 21: Direções da nave espacial em *Space Invaders*.



Fonte: Elaborada pelo autor.

Desse contexto o modelo matemático para movimento de A seria:

Para movimento em y: $\text{sen}\,(\alpha) = \frac{y}{r}$

$y = r * \text{sen}\,(\alpha)$

Para movimento em x: $\cos(\alpha) = \frac{x}{r}$

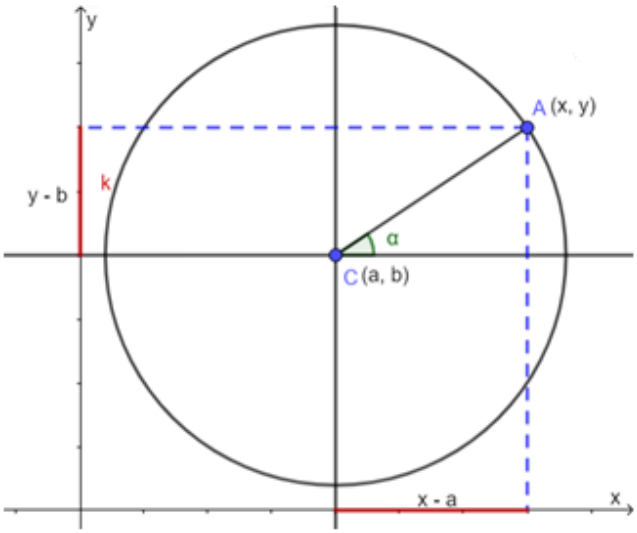
$x = r * \cos(\alpha)$

Dessa forma temos o seguinte ponto: A $(r * \text{sen}\,(\alpha), r * \cos(\alpha))$.

A última considerações feita por Tonéis (2015, p. 134) é:

3- "Quando o centro não está na origem do sistema e o raio r qualquer", a Figura 22 ilustra tais dizeres:

Figura 22: Direções da nave espacial em *Space Invaders.*



Fonte: Elaborada pelo autor.

Desse contexto o modelo matemático para movimento de A seria:

Para movimento em y: $\text{sen}\,(\alpha) = \dfrac{y-b}{r}$

$y = b + r * \text{sen}\,(\alpha)$

Para movimento em x: $\cos(\alpha) = \dfrac{x-a}{r}$

$x = a + r * \cos(\alpha)$

Dessa forma temos o seguinte ponto: A $(a + r * \text{sen}\,(\alpha), b + r * \cos(\alpha))$. Tonéis (2015), conclui que em movimento circular podemos localizar o objeto após a rotação por meio dessas duas equações paramétricas acima.

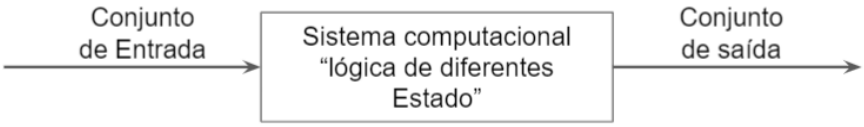
4.2- O Jogos Digitais com Inteligência Artificial

Existem diversas técnicas de inclusão da IA em jogos digitais, segundo Loureiro Júnior (2013, p. 6) "[...] dentre elas, destacam-se: Sistemas especialistas, Sistemas Baseados em Regras, Sistemas Fuzzy, Raciocínio Baseado em Casos, redes neurais, máquinas de estados, algoritmos genéticos, algoritmos de busca, etc.". No escopo deste trabalho abordamos duas técnicas para o desenvolvimento de jogos digitais: a máquinas de estado finita (FSM – *Finite State Machine*) e a Lógica Fuzzy.

4.2.1- Máquina de Estado Finita

Uma máquina de estado finita ou autômato finito, ou ainda máquina de estados, é um modelo matemático composto de um número finito de estados e é amplamente utilizado para representar circuitos lógicos ou programas computacionais. Nas palavras de Buckland (2005), historicamente, uma máquina de estado finito é um dispositivo rigorosamente formalizado usado por matemáticos na resolução de problemas. Ainda segundo esse autor podemos compreender ela, ainda, como uma abstração de software, que nos permite descrever como se comporta um sistema computacional, Figura 23.

Figura 23: Conjunto de entrada e saída de uma máquina de estado finito



Fonte: Elaborada pelo autor

Podemos dizer, pela figura que uma Máquina de Estados Finitos é uma representação de um sistema computacional que vai pegar entradas, a qual deve modificar o estado em que o

sistema se encontra e depois dessa mudança de estado produz saídas. Segundo Oliveira (2005) uma definição para uma Máquina de Estado Finito seria:

> M= [S, I, O, f_s, f_o] é uma máquina de estado finito se:
> S for um conjunto finito de estados;
> I for um conjunto finito de símbolos de entrada (o alfabeto de entrada);
> O for o conjunto finito de saída (o alfabeto de saída);
> f_s e f_o forem funções onde f_s : S×I em S e f_o : S em O.
> A máquina de estado finito (autômato) é sempre iniciada em estado inicial fixo S_0. A função f_0 é uma função do próximo estado e leva pares (estados, entradas) em estados Portanto o estado no ciclo do relógio t_{i+1}, $s_{(ti+1)}$, é obtido pela aplicação da função do próximo estado ao estado no início t_I e à entrada do ciclo t_i, como melhor representado nas relações em seguida. $s(t_{i+1}) = fs(s(t_I), i(t_i))$, no qual i = entrada (input) A função f_o é uma função da saída: $O(t_i) = f_o(s(t_I))$. (OLIVEIRA, 2005, p. 17).

O dizeres desse pesquisar mostra que esse tipo de máquina é descrito cinco coisas (quíntupla):

1- conjunto de entradas;
2- conjuntos de saídas;
3- conjunto de estados;
4- transições entre estados (as mudanças de um estado para outro). Essas transições são visualizadas pelos Gráficos de Transição de Estados e/ou representados pela Matriz de Transição de Estados. Essas duas representações é que dão a forma de nossa máquina de estado;
5- Determinação das saídas.

Já para Millington e Funge (2009) uma máquina de estados é um modelo matemáticos que é usado para representar softwares, a qual pode ser descrita por três coisas: conjunto de estados, regras de transição entre os estados e estado atual.

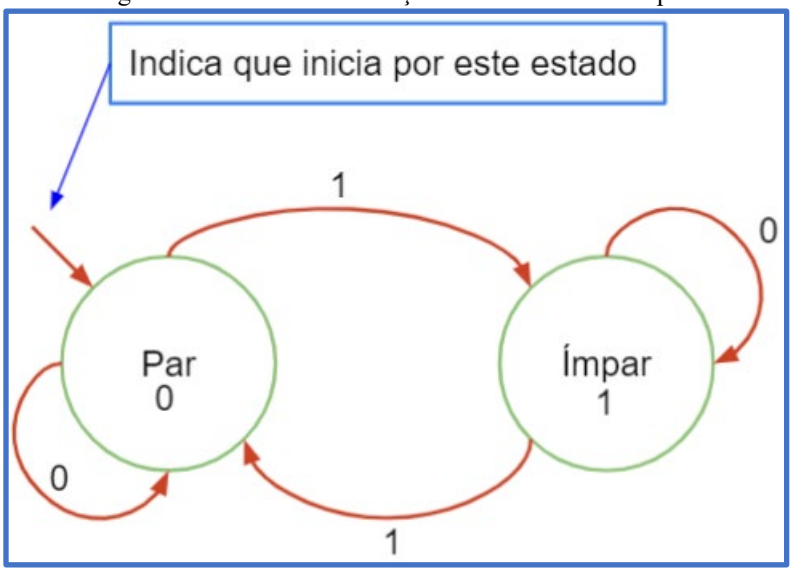
Esses autores afirmam que há dois tipos de FSM, as Máquinas de *Mealy* e de *Moore*. Segundo Vieira (2004, p. 103) "As máquinas de *Mealy* e de *Moore* são autômatos finitos com saída. Uma máquina de *Mealy* associa um símbolo de saída a cada transição. E uma máquina de *Moore* associa um símbolo de saída a cada estado". Isto é:

1- Nas máquinas de *Mealy*
    a. PE = f(estado atual, entrada). PE é o próximo estado e f é função de mundas do estado.
    b. S = g(estado atual, entrada), em que S é a saída e g a função de saída da máquina.
2- Nas máquinas de *Moore*
    a. PE = f(estado atual, entrada)
    b. S = g(estado atual), em que S é a saída e g a função de saída da máquina.

Há de se verificar que as máquinas de *Mealy* são mais complexas que as máquinas de *Moore*. Contudo elas permitem criar *softwares* que são descritos com as quíntuplas. Por exemplo, vamos criar uma máquina de estados que recebe uma cadeia de bits, e toda vez que receber um número par ela permanece com o número que está na máquina, mas quando for ímpar ela muda o número. Usando a representação da Figura 23 temos, por exemplo, a cadeia de bits 011001001, teríamos como saída a cadeia 001000111, pois o primeiro número a entrar na máquina seria um 1, ímpar, o que muda o número, como está no começo o número passa de zero para 1. Depois a entrada é zero, par, logo terá como saída outro 1, pois zero não muda o número que está na máquina, e assim sucessivamente. Por ser uma máquina de estado finito é possível extrairmos a quíntupla:

1- conjunto de entradas = 1 bit, isto é, um número na base dois 0 ou 1;
2- conjuntos de saídas = 1 bit, isto é, um número na base dois 0 ou 1;
3- conjunto de estados = há dois estados. Quando é par permanece o número e quando é ímpar muda-se o número;
4- transições entre estados = é visto na Figura 24 pelo Gráfico de Transição de Estados

Figura 24: Gráfico de Transição de Estados do exemplo.



Fonte: Elaborada pelo autor.

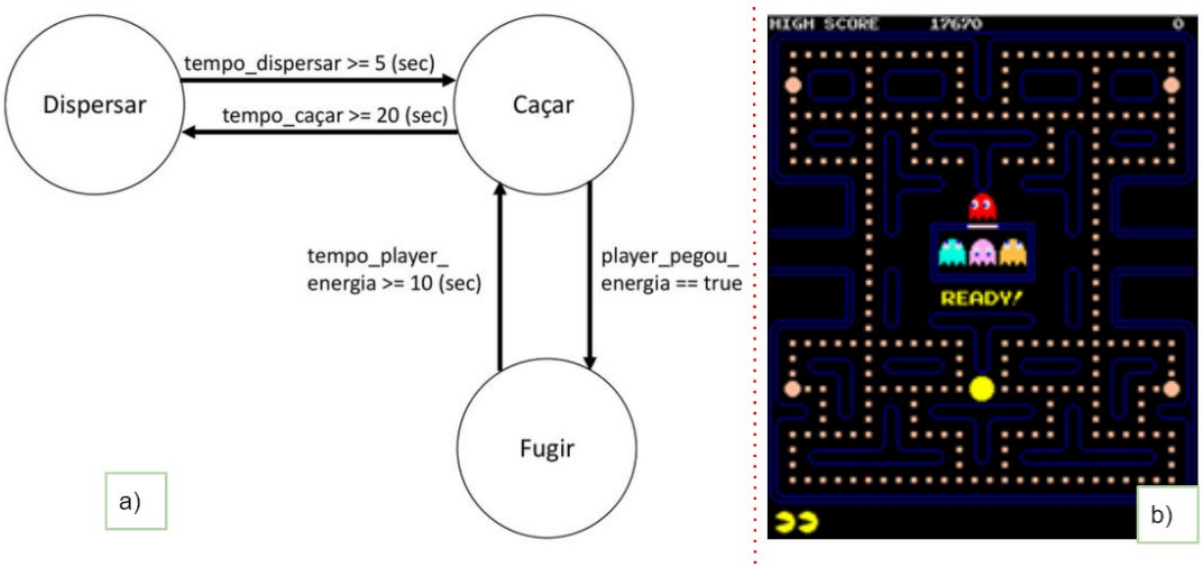
A Figura 24 mostra uma Máquina de Estados Finitos de Moore, a qual a saída depende do estado atual. Assim no estado Par a saída deve ser zero (0) e no estado ímpar a saída deve ser um (1). Na figura a flecha que chega ao círculo, do lado esquerdo, indica o estado inicial da máquina, o estado par. Como começa nesse estado e podemos ter uma entrada de um número zero, o qual nos mantém no próprio estado, dessa forma indicamos com flecha que saí e chega no mesmo estado. Contudo o número de entrada por ser um, então mudamos para outro estado,

o estado ímpar, ligamos com uma flecha saindo do estado par e chegando no estado ímpar. Agora no estado ímpar se tivermos zero, permanecemos neste estado, o qual indicamos com uma flecha saindo e chegando no mesmo estado. Mas se o número de entrada for um, então mudamos para outro estado, o estado o par.

5- Determinação das saídas = a saída 001000111

Obviamente o comportamento dos agentes de um jogo é um pouco mais complexo do que este exemplo apresentado, mas um exemplo clássico do uso de Máquinas de Estados Finitos são os comportamentos dos fantasmas Blinky, Pinky, Inky e Clyde do jogo *Pac-Man*. Os fantasmas possuem três tipos de estado, o *dispersar* que ocorre no início da fase com cada fantasma se posicionando em um dos cantos, existe também um estado de *fugir* comum a todos os fantasmas, e um estado *caçar* cujas ações são determinadas de forma diferente a cada fantasma. A ação *pegar energia* realizada pelo jogador é a condição para a transição *fugir*. O cronômetro zerado é a condição para a transição de *fugir* para *caçar*.

Figura 25: Máquina de Estado Finito e Interface do jogo *Pac-man*.



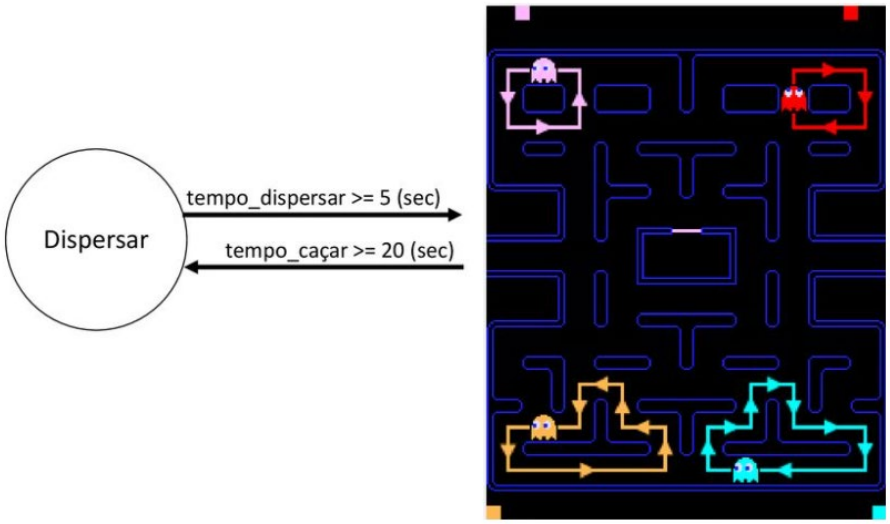
Fonte: Lima (2014) e Bandai Namco Entertainment (2018).

A Figura 25 temos como quíntupla que descreve a máquina:

1- conjunto de entradas = {tempo_dispersar, tempo_caçar, tempo_player_energia, player_pegou_energia};
2- conjuntos de saídas = {tempo_dispersar, tempo_caçar, tempo_player_energia, player_pegou_energia};
3- conjunto de estados = Dispersar, Caçar e Fugir;
4- transições entre estados = Figura 25 no item a);
5- Determinação das saídas = Figura 25 no item b).

Lima (2014) explica que quando o estado *dispersar* for acionado, os fantasmas se movimentam em direção aos cantos do mapa do jogo e ficam andando em círculo por um período de tempo, Figura 26.

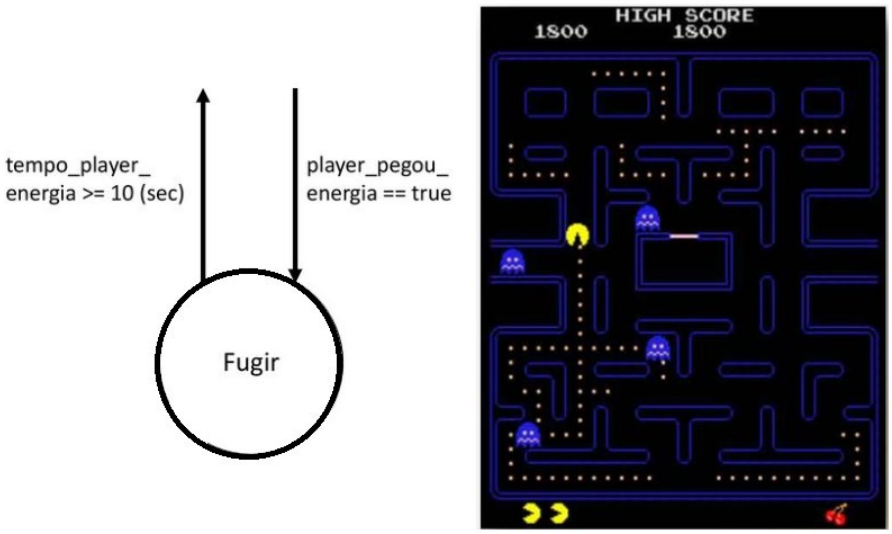
Figura 26: Máquina de Estado Finito e Interface do jogo Pac-man.



Fonte: Lima (2014).

Já ao acionar o estado de fugir os fantasmas movimentam-se mais lentamente com movimentos aleatórios, Figura 27.

Figura 27: Máquina de Estado Finito e Interface do jogo Pac-man.

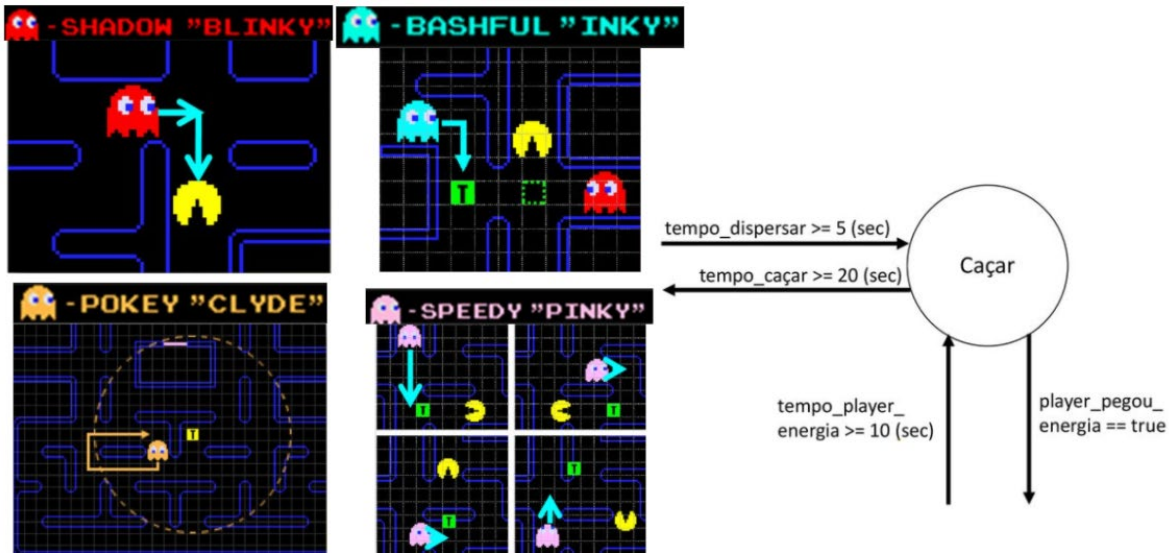


Fonte: Lima (2014).

Quando acionado o estado caçar, cria-se um comportamento diferente para cada um dos 4 fantasmas, o que os tornam de certa forma “únicos”, conforme exemplificado por Lima (2014). Sempre que o estado caçar for ligado, o fantasma Blink irá movimentar-se sempre mirando na posição em que o *Pac-man* se encontra, o Pinky movimenta-se mirando em uma das 4 posições titles a frente do *Pac-Man*, o Inky mira em uma posição que combina a

posição/direção do *Pac-man* e do Blinky e o Clyde, quando está longe movimenta-se em direção ao *Pac-man*, caso esteja perto irá tentar cerca-lo. Figura 28.

Figura 28: Estado de caçar.



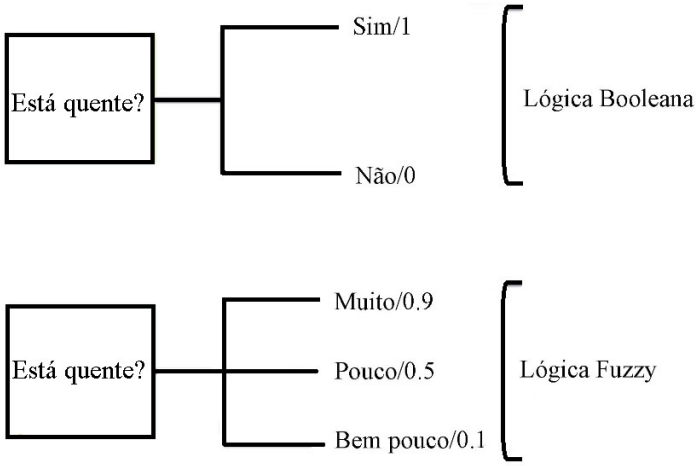
Fonte: Lima (2014).

A principal desvantagem de utilizar FSM é o fato de termos condições predeterminadas, tornando a estratégia utilizada pela máquina previsível, além disso, conforme o sistema vai ficando mais elaborado o número de estados e transições podem se tornar um grande problema. Ainda assim, é possível criar implementações mais complexas utilizando a Hierarquia de Máquinas de Estado Finito (*HFSM – Hierarchial Finite State Machines*).

4.2.2- Lógica Fuzzy

A lógica Fuzzy, também chamada lógica nebulosa ou lógica difusa, é um método de raciocínio criado pelo americano Lotfi Zadeh em meados da década de 60, que se assemelha bastante ao pensamento humano. É uma abordagem que permite que o computador realize tomadas de decisões bastante complexas sobre os termos e regras linguísticas de forma análoga ao homem. (BUCKLAND, 2005). Na lógica booleana tradicional tais decisões não são possíveis, uma vez que ao receber uma entrada, o computador produz uma saída definida como *verdadeiro* ou *falso,* que seria equivalente ao *sim* ou *não* de uma pessoa, Figura 29.

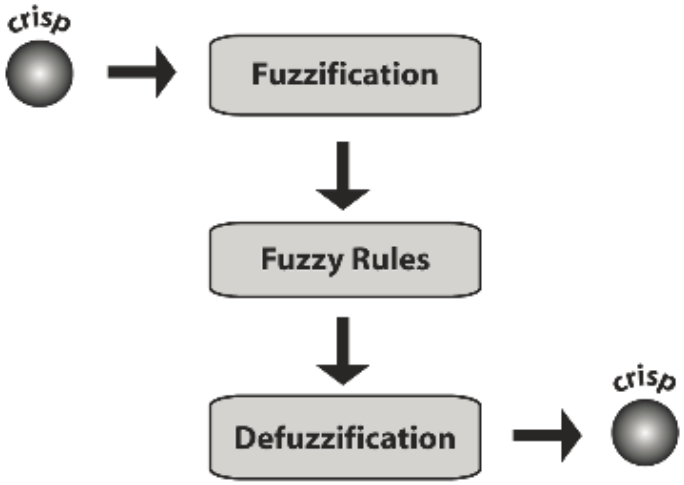
Figura 29: Níveis de possibilidades na lógica Fuzzy.



Fonte: Elaborada pelo autor.

A lógica Fuzzy é uma teoria matemática que leva em conta o aspecto da incerteza, ou seja, considera possibilidades intermediárias entre o sim e o não. Conceitos como "muito" ou "pouco" podem ser representados por conjuntos Fuzzy, permitindo assim que valores numéricos sejam convertidos para um conjunto nebuloso, esse processo é chamado de fuzificação. A arquitetura de lógica Fuzzy, segundo Buckland (2005), consiste em três partes principais, Figura 30.

Figura 30: Regra de fuzzy baseada na inferência[6]



Fonte: Buckland (2005)

- Fuzificação: Converte as entradas em valor númerico em conjuntos *Fuzzy*. Para realizar essa conversão usa-se as funções de inferência.

[6] Segundo Souza (2010, p. 3) na "[...] linguagem fuzzy um subconjunto clássico costuma ser denominado subconjunto crisp".

- Mecanismo de inferência: Determina o grau de correspondência entre a entrada e as regras. Fornece os parâmetros a uma máquina de inferência de acordo com os campos de entrada, processa uma série de regras do tipo SE/ENTÃO e decide quais regras devem ser acionadas.
- Defuzificação: Esse processo converte os conjuntos *Fuzzy* em um valor númerico. Existem vários tipos de técnicas disponíveis que podem ser escolhidas de acordo com a necessidade, podemos citar como exemplos os metódos centróide, média dos máximos, bissetor, entre outros.

4.2.3- Conjuntos Fuzzy

Na teoria clássica dos conjuntos, da lógica booleana, o processo que determina quais elementos do conjunto universo pertencem ou não a um conjunto é definido, segundo Souza (2010, p. 2), da seguinte forma: "Seja U um universo de discurso e A um subconjunto de U. A função característica de A é dada por"

$$\mu A\ (x) = \begin{cases} 1, \text{se x } \in \text{A} \\ 0, \text{se x } \notin \text{A} \end{cases}$$

Temos, então, que μA é uma função domínio U e imagem contida no conjunto {0, 1}. Nesse contexto, $\mu A\ (x) = 1$ significa afirmar que o elemento x está em A, enquanto $\mu A\ (x) = 0$ significa afirmar que x não é elemento de A (SOUZA, 2010). Esse autor constata que na teoria booleana a função característica, μA, "descreve completamente o conjunto A, já que tal função indica quais elementos do conjunto universo U são também elementos de A. **No entanto, existem casos em que [...] não sabemos dizer se um elemento pertence efetivamente a um conjunto ou não**" (SOUZA, 2010, p. 2, grifos nossos).

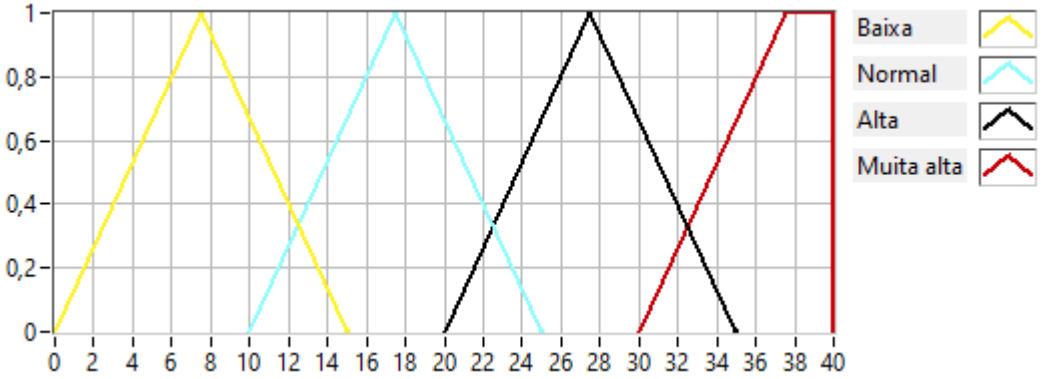
Desse ponto de vista, na teoria dos conjuntos Fuzzy, podemos definir conjunto Fuzzy, segundo Souza (2010, p. 3), da seguinte maneira: "Seja U um universo de discurso. Um subconjunto Fuzzy F de U é caracterizado por uma função $\boldsymbol{\varphi F}: \boldsymbol{U} \rightarrow [\mathbf{0}, \mathbf{1}]$, pré-fixada, chamada função de pertinência do subconjunto fuzzy F. [...] A classe de todos os subconjuntos Fuzzy de U é denominado por F(U)".

Portanto um conjunto Fuzzy pode ser modelado por meio de uma função de pertinência $\mu A\ (x)$: X → [0, 1], admitindo a existência de outros valores entre intervalo de 0 a 1. Esses valores indicam o grau de pertinência dos elementos de U em relação ao conjunto A, onde A é o conjunto Fuzzy. Vejamos o exemplo dado por Souza (2010, p. 3):

> Considere o subconjunto F dos números reais próximos de 2:
> F = {x ∈ R : x é próximo de 2} [...]
> Se definirmos a função $\boldsymbol{\varphi F}: \boldsymbol{R} \rightarrow [\boldsymbol{0}, \boldsymbol{1}]$ que associa cada x real ao valor de proximidade ao ponto 2 pela expressão
>
> $$\boldsymbol{\varphi F}(\boldsymbol{x}) = \begin{cases} (\boldsymbol{1} - |\boldsymbol{x} - \boldsymbol{2}|, & \boldsymbol{se\ x} \in [\boldsymbol{1}, \boldsymbol{3}] \\ \boldsymbol{0}, \boldsymbol{se\ x} \in [\boldsymbol{1}, \boldsymbol{3}] \end{cases}$$
>
> então o subconjunto fuzzy F dos pontos próximos de 2, caracterizado por $\boldsymbol{\varphi F}$, é tal que $\boldsymbol{\varphi F}$(2, 001) = 0, 999 e $\boldsymbol{\varphi F}$(7) = 0. Nesse caso, dizemos que x = 2, 001 é um ponto próximo de 2 com grau de proximidade 0,999 e x = 7 não é próximo de 2.

A função de pertinência pode ser representada por um gráfico que define a forma como cada ponto no espaço de entrada é mapeado para o valor de pertinência entre 0 e 1. Ela permite quantificar termos linguísticos (baixa, normal, alta, muito alta) e representar graficamente um conjunto Fuzzy. Conforme exibido na Figura 31, temos a temperatura exibida em ºC, representada graficamente por um conjunto Fuzzy, onde os elementos pertencentes ao conjunto são valores possíveis para a temperatura, representada no eixo °C.

Figura 31: Função de pertinência de temperatura.



Fonte: Adaptado de Buckland (2005).

A escolha da função de pertinência depende muito do contexto analisado, as funções podem ter variadas formas, mas as mais utilizadas são tipicamente triangulares ou trapezoidais. Um número fuzzy é triangular se a sua função de pertinência segue a seguinte forma:

$$\mu A(x) = \begin{cases} 0, \text{se } x \leq a, \\ \dfrac{(x-a)}{u-a}, \text{se } a < x \leq u, \\ \dfrac{(x-b)}{u-b}, \text{se } u \leq x < b, \\ 0, \text{se } x \geq b. \end{cases}$$

O gráfico da função de pertinência triangular representado na Figura 33, tem a forma de um triângulo, e tem como base o intervalo [a, b] e como vértice o ponto (*u*, 1). Dessa forma o triangulo é denotado pela seguinte ordenação (a; *u*; b).

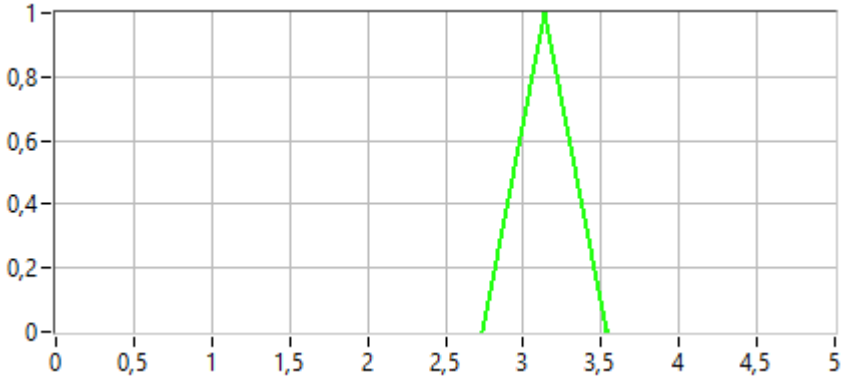
Figura 32: Gráfico da função de pertinência triangular.

Fonte: Adaptado de Bender (1996).

A partir dessa função podemos, por exemplo, modelar matematicamente o valor "aproximado" do número π, representado graficamente na figura 34. Dados que a = 2,74; u = 3,14 e b = 3,54. Temos a seguinte função de pertinência:

$$\mu A(x) = \begin{cases} 0, \text{se } x \leq 2{,}74; \\ \dfrac{(x - 2{,}74)}{3{,}14 - 2{,}74}, \text{se } 2{,}74 < x \leq 3{,}14; \\ \dfrac{(x - 3{,}54)}{3{,}14 - 3{,}54}, \text{se } 3{,}14 \leq x < 3{,}54; \\ 0, \text{se } x \geq 3{,}54. \end{cases}$$

Figura 33: Gráfico da função de pertinência do número Fuzzy "valor aproximado de π"



Fonte: Adaptado de Buckland (2005)

O gráfico da função de pertinência trapezoidal, representado na figura 35, tem a forma de um trapézio, e sua base maior e denotada pelo intervalo [a, d] e a base menor por [b, c]. Assim, o número *Fuzzy* trapezoidal é definido pelos números reais a, b, c e d, bem como se a sua função de pertinência segue a seguinte forma:

$$\mu A(x) = \begin{cases} \dfrac{(x-a)}{b-a}, \text{se } a \leq x < b, \\ 1, \text{se } b \leq x \leq c, \\ \dfrac{(d-x)}{d-c}, \text{se } c < x \leq d, \\ 0, \text{caso contrário.} \end{cases}$$

Figura 34: Gráfico da função de pertinência trapezoidal.

Fonte: Adaptado de Bender (1996).

É importante salientar que outros tipos de gráficos, conforme a definição de número *Fuzzy* e respectivas funções pertinência os possibilita criar. Segundo Souza (2010, p. 5) a definição de números *Fuzzy* é:

> Um subconjunto fuzzy A é chamado de número fuzzy quando o conjunto universo no qual $\boldsymbol{\varphi A}$ está definida, é o conjunto dos números reais R e satisfazem às seguintes condições:
> 1. todos os α-níveis de A são não-vazios, com $0 \leq \alpha \leq 1$;
> 2. todos os α-níveis de A são intervalos fechados de R;
> 3. $\text{supp}_A = \{x \in R : \boldsymbol{\varphi A}(x) > 0\}$ é limitado.

Esse autor explica, ainda, que supp_A é denominado suporte de A, um subconjunto do conjunto clássico de U, no qual $\text{supp}_A = \{x \in U: \boldsymbol{\varphi A}(x) > 0\}$. Além disso, é necessário explicar que α-nível de A "[...] é o subconjunto clássico de U definido por $[A]^{\alpha} = \{x \in U: \boldsymbol{\varphi A}(x) \geq \alpha\}$, se $0 < \alpha \leq 1$" (SOUZA, 2010, p. 5).

4.2.4- Jogos com Fuzzy

Durante muito tempo as técnicas de programação em jogos digitais estiveram restritas aos padrões de movimentos e FSM's. Essas técnicas tinham como principal fator limitante a interação player (jogador) e NPC (personagem não jogável, do inglês: *non-player character*), pois ao receber uma pergunta o NPC deveria apresentar uma resposta exata, ou seja, a decisão

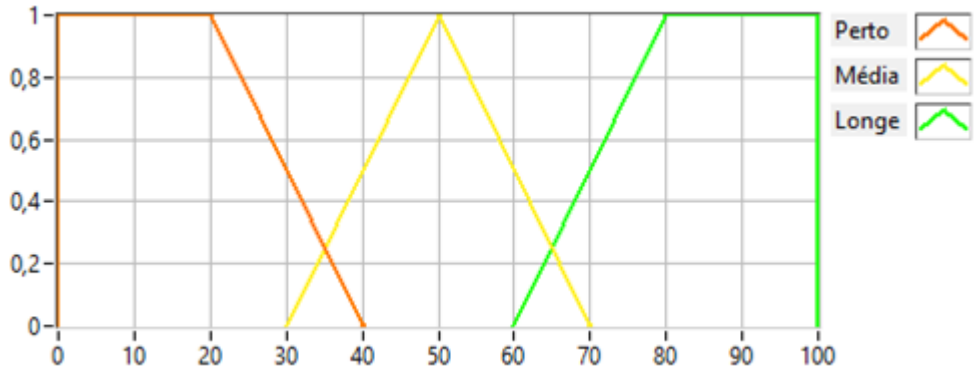
seria sim ou não. Por exemplo, no jogo Karate Champ [7], se o NPC quiser saber se vai atacar ou não o personagem do jogador ele irá analisar e tomará a decisão utilizando a lógica clássica, atacar (sim) ou não atacar (não), uma decisão baseada em regras pré-definidas. Se utilizarmos a FSM teremos o mesmo resultado, uma vez que a decisão de mudança de estado só pode ter como respostas o sim ou o não (BUCKLAND, 2005).

Uma das principais características da Lógica Fuzzy é a sua versatilidade e é isso que a torna uma excelente escolha para os jogos eletrônicos, uma vez que a mesma apresenta flexibilidade e capacidade de adaptação às aplicações. Em um jogo de combate um contra um, poderíamos identificar na modelagem a posição (eixos x e y) do jogador e a sua distância até o NPC e a quantidade de vida (saúde). Tomando como base essas entradas, podemos inferir as regras destas variáveis e a ação que deverá ser realizada pelo NPC, como exemplificado nas duas variáveis linguísticas: distância, Figura 35, e saúde, Figura 36. Os comportamentos do NPC são recuo, defesa e ataque e variam de acordo com o player e o seu estado. As explicações detalhas sobre as ações do NPC são:

1- Recuar: O NPC se move na direção oposta ao player, aumentando a distância do player para o NPC.
2- Defender: O NPC irá bloquear ataques do player, essa mitigação de dano pode corresponder, por exemplo, a 30% do dano normal.
3- Atacar: O NPC irá atacar o player, caso o golpe o acerte o jogador terá uma redução de 10% em seu HP.

A variável distância é o intervalo entre o player e o NPC, inimigo, variando de 0 a 100.

Figura 35: Conjunto de membros para variável distância.

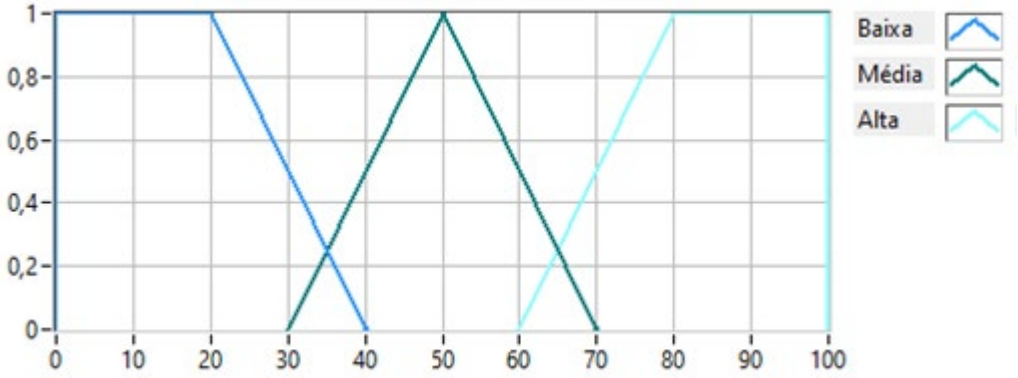


Fonte: Elaborada pelo autor.

A variável vida é a quantidade de "saúde" do NPC, inimigo, variando de 0 a 100. A vida do NPC pode ser reduzida em 10% a cada golpe recebido pelo player.

[7] Lançado em 1984 e precursor do gênero de combate um contra um (do inglês *one-on-one fight game*)

Figura 36: Conjuntos de membros para variável vida.



Fonte: Elaborada pelo autor.

Combinando as duas variáveis distância e saúde, observe que temos 9 regras no total conforme demonstrado abaixo.

SE (distância É PERTO) E (vida É BAIXA) ENTÃO (ação é RECUAR)

SE (distância É PERTO) E (vida É MÉDIA) ENTÃO (ação é ATACAR)

SE (distância É PERTO) E (vida É ALTA) ENTÃO (ação é ATACAR)

SE (distância É MÉDIA) E (vida É BAIXA) ENTÃO (ação é DEFENDER)

SE (distância É MÉDIA) E (vida É MÉDIA) ENTÃO (ação é ATACAR)

SE (distância É MÉDIA) E (vida É ALTA) ENTÃO (ação é ATACAR)

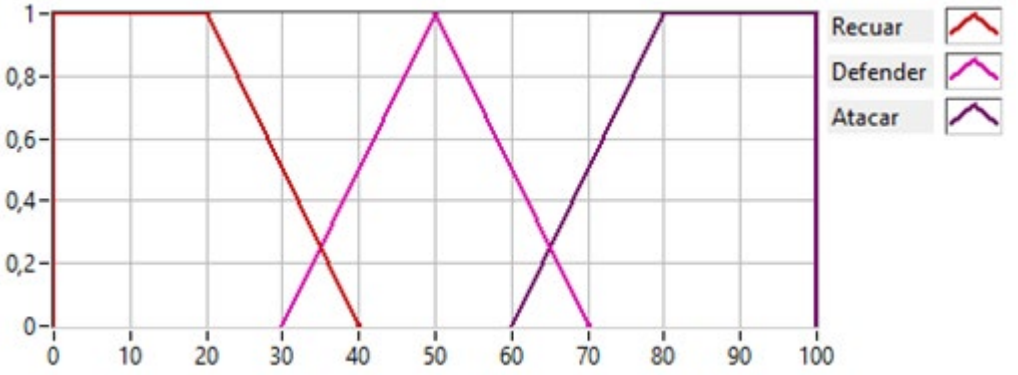
SE (distância É LONGE) E (vida É BAIXA) ENTÃO (ação é DEFENDER)

SE (distância É LONGE) E (vida É MÉDIA) ENTÃO (ação é ATACAR)

SE (distância É LONGE) E (vida É ALTA) ENTÃO (ação é ATACAR)

Essas ações podem ser visualizadas no gráfico triangular na Figura 37.

Figura 37: Conjuntos de membros para variável Ação.



Fonte: Elaborada pelo autor.

Combinando todas as regras obtemos uma tabela de regras conforme mostrado na Figura 38.

Figura 38: Regras Fuzzy.

	Baixa	Média	Alta
Perto	RECUAR	ATACAR	ATACAR
Médio	DEFENDER	ATACAR	ATACAR
Longe	DEFENDER	ATACAR	ATACAR

Fonte: Elaborada pelo autor.

Podemos observar que trabalhar nos jogos digitais com a lógica Fuzzy os artefatos computacionais (*bots*, NPC, entre outros) é possível trazer as variáveis matemáticas as imprecisões do comportamento humano. Se antes os desenvolvedores olhavam para detecção de colisão, para o teorema de Pitágoras, para que o inimigo atacasse o jogador, com *Fuzzy* organiza e entrelaça outros parâmetros para fazer a ação mais perto do que um humano faria. Como diz Souza (2010, p. 2) a Teoria dos Conjuntos Fuzzy é um passo na direção “de se programar e armazenar conceitos vagos em computadores, tornando possível a produção de cálculos com informações imprecisas, a exemplo do que faz o ser humano”.

5- CONCLUSÃO

Com este trabalho, buscamos evidenciar a importância da Matemática na implementação de jogos digitais com inteligência artificial. Mais especificamente, foram apresentados de forma sucinta alguns métodos e modelos matemáticos que facilitam o desenvolvimento, auxiliam na resolução de problemas e a sua evolução histórica através da linha do tempo. Devido ao momento atual e considerando os efeitos da pandemia da COVID-19, em especial na educação, decidimos limita-lo ao campo teórico.

No decorrer deste estudo, apresentamos alguns modelos matemáticos utilizados desde os primórdios dos jogos digitais. Consideramos o contexto histórico de grande relevância para a compreensão deste trabalho, por isso procuramos manter uma linearidade temporal, principalmente pela relevância apresentada pela Matemática na resolução de problemas numa época em que havia tanto a limitação de *hardware* quanto de metodologias de desenvolvimento.

Para alcançar os objetivos deste trabalho, foi necessário realizar a fundamentação teórica onde estudamos as definições de inteligência artificial, plano cartesiano, equação da reta, sistema de colisões, ângulos, conversões, máquina de estado finita (FSM) e a lógica fuzzy. Essa última, em especial se tornou um grande desafio por não ser ensinada na graduação, mas também nos trouxe uma nova perspectiva. Ao apresentarmos de modo formal alguns dos seus conceitos básicos, novos questionamentos foram surgindo, como por exemplo: *quais as possibilidades de aplicação da lógica fuzzy em sala de aula?*

Nesse sentido, entendemos que ainda há muito o que ser pesquisado, dessa forma estabelecem-se algumas recomendações para pesquisas futuras.

a) A construção de jogos digitais e sua aplicabilidade em sala de aula
b) Utilização do ambiente fuzzy para tomadas de decisões
c) Aplicação da lógica fuzzy para solução de problemas reais

Desse ponto de vista concluímos que o novo emergente para as criações computacionais é também a urgência de nos cursos de formação que trabalham com a Matemática introduziram no processo formativa a *Matemática Fuzzy*, pois a cultura digital desse mundo computacional está a exigir que o professor de matemática discuta desde a educação básica esses novos saberes da matemática.

REFERÊNCIAS

ARAÚJO, Anderson Souza. **A influência dos agentes inteligentes na navegação dos usuários em ambientes virtuais.** Tese (Mestrado em Ciências em Engenharia de Sistemas e Computação) – Coordenação dos Programas de Pós-graduação de Engenharia, Universidade Federal do Rio de Janeiro. Rio de Janeiro, p.94. 2004.

BANDAI NAMCO ENTERTAINMENT (Tóquio). PAC-Man: Official Website**.** 2018. Disponível em: https://pacman.com/en/games/pacman.php. Acesso em: 18 maio 2021.

BENDER, Edward A.. **Mathematical Methods in Artificial Intelligence**. Los Alamitos: Ieee Computer Society, 1996. 636p.

BIEMBENGUT, Maria Salett. Modelagem Matemática & Implicações no Ensino-Aprendizagem de Matemática. Editora da FURB: Blumenau, 1999.

BRASIL. Ministério da Saúde. Secretaria de Vigilância em Saúde. Departamento de Análise em Saúde e Doenças não Transmissíveis. Guia de vigilância epidemiológica Emergência de saúde pública de Importância nacional pela Doença pelo coronavírus 2019 – covid-19 [recurso eletrônico] / Ministério da Saúde, Secretaria de Vigilância em Saúde. – Brasília: Ministério da Saúde, 2021. 86p.

BUCKLAND, Mat. **Programming Game AI by Example**. Plano: Wordware Publishing, Inc., 2005. 495p.

DAMIÃO, Mateus Araujo; CAÇADOR, Rodigo Menezes Costa; LIMA, Sérgio Muinhos Barroso. Princípios e Aspectos Sobre Agentes Inteligentes. **Revista Eletrônica da Faculdade Metodista Granbery**, Juiz de Fora, v. X, n. 17, p. X, jul. 2014.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas, 2002. 171p.

HARRIS, Ryan A.; GORASIA, Jayesh B.. **Pong**, 2008. 4p. Disponível em: < http://www.jgorasia.com/Files/Spring08/ICB/Gorasia_Harris.pdf>. Acesso em 18 mar. 2021.

HORTON, John. **Calculating heading in 2d games**. Game Code School, 2016. Disponível em: <https://gamecodeschool.com/essentials/calculating-heading-in-2d-games-using-trigonometric-functions-part-1/>. Acesso em: 10 abr. 2021.

KRIPKA, Rosana Maria Luvezute; SCHELLER, Morgana; BONOTTO, Danusa de Lara. **Pesquisa Documental na pesquisa qualitativa: conceitos e caracterização**. Revista de Investigaciones UNAD, v.14, n. 2, p. 55-73, jul.-dez. 2015.

LUGER, George F.. **Inteligência Artificial**. 6. ed. São Paulo: Pearson Education do Brasil, 2013. 614p. Tradução de Daniel Vieira.

LIMA, Edirlei Soares de. **Máquinas de Estados Finitos**. Edirlei.com. 2014. Disponível em: <https://edirlei.com/aulas/ia_2014_1/IA_Aula_21_Maquinas_de_Estados_Finitos_2014.html >. Acesso em: 15 mar. 2021.

LIMA, Elon Lages. *et al*., Temas e Problemas Elementares. Rio de Janeiro: SBM, 2006. 256 p.

MILINGTON, Ian; FUNGE, John. Artificial Intelligence for games. 2. ed. Burlington: Elsevier, 2009. 870p.

MORAES, R. Uma tempestade de luz: A compreensão possibilitada pela análise textual discursiva. Ciência e Educação, Bauru, v. 9, n. 2, p. 191-211, 2003. Disponível em: < http://www.scielo.br/pdf/ciedu/v9n2/04.pdf>. Acesso em: 06 mai. 2021.

MORAES, R.; GALIAZZI, M. C. Análise textual: discursiva. 2. ed. Revisada. Ijuí: Editora Unijuí, 2011.

NAKAMITI, Eduardo Kiochi. **Agentes inteligentes artificiais.** Tese (Mestrado em Comunicação e Semiótica) – Pontifícia Universidade Católica de São Paulo. São Paulo, p.84. 2009.

OLIVEIRA, Luciano José de. A Modelagem de Máquinas de Estado Finito. Monografia (Bacharelado em Ciência da Computação) – Curso de Ciência da Computação da Faculdade de Jaguariúna. Jaguariúna, p.68. 2005.

OLIVEIRA, Thiago Rodrigues de; BORKS, João; NESTERIUK, Sérgio. **Inovando no Level Design: cenários dinâmicos de combate com Física Mecânica e Eletromagnética**. *In*: XVI Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames), Trilha de Arte & Design. Curitiba: PUCPR, 2017.

ROSA, João Luís Garcia. **Fundamentos da Inteligência Artificial**. Rio de Janeiro: Ltc, 2011. 212p.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 3. ed. Rio de Janeiro: Elsevier, 2013. 957 p. Tradução de Regina Célia Simille de Macedo.

SALEM, Katie; ZIMMERMAN, Eric. **Rules of Play**: game design fundamentals. Massachusetts: Mit Press, 2004. 670p.

SCHAWB, Brian. **AI Game Engine Programming**. 2. ed. Boston: Course Technology, 2009. 710p.

SOUZA, Osmar do Nascimento. **Introdução à Teoria dos Conjuntos Fuzzy**. Londrina, 2010. Disponível em: < http://www.ime.unicamp.br/~valle/PDFfiles/osmar10.pdf>. Acesso em 18 mai. 2021.

TECMUNDO. Tecmundo, 2018. **O que é detecção de colisões?** Disponível em: < https://www.tecmundo.com.br/video-game-e-jogos/1059-o-que-e-deteccao-de-colisoes-.htm>. Acesso em: 25 abr. 2021.

TONÉIS, Cristiano Natal. **Matemática Aplicada aos Games**. São Paulo: ArteSam, 2015. 255p.

VIEIRA, Newton José. Linguagens e Máquinas: Uma introdução aos fundamentos da computação. Belo Horizonte: UFMG, 2004. 297p.

WOLF, Mark J. P.. **The Video Game Explosion**: a history from pong to playstation and beyond. Westport: Greenwood Press, 2008. 380p.