APLICAÇÃO DO ALGORITMO KNN PARA CONTROLE DE MOVIMENTOS DE NPCS EM UM AMBIENTE DINÂMICO (JOGO)

Bruno Roberto Bricce[1]
Patrick Pedreira Silva [2]
Henrique Pachioni Martins[3]
Elvio Gilberto da Silva[4]

RESUMO
O projeto consistiu na criação de um jogo que foi desenvolvido na linguagem JavaScript utilizando o motor gráfico Unity 3D, com intuito de aplicar o algoritmo KNN para movimentação de um personagem não controlado (NPC). O jogo foi feito em 2D, com interface simples e clara, para qualquer tipo de jogador (usuário), poupando de uma longa aprendizagem para começar a jogar. O projeto consiste em armazenar as jogadas do usuário e realizar o aprendizado de máquina para que, no fim, o jogador possa ver como o NPC se sairá jogando sozinho, com base nas jogadas realizadas anteriormente, ou seja, quanto mais jogar mais o computador aprenderá sobre o jogo. Este trabalho também apresenta a importância de avaliar formas de inteligência artificial para jogos eletrônicos.
PALAVRAS-CHAVE: Inteligência Artificial. K-NN. NPC. Jogo Eletrônico.

ABSTRACT
The project consists in creating a game that will be developed in JavaScript language using the Unity 3D graphics engine, aiming to apply the KNN algorithm for moving an Non Player Character (NPC). The game will be 2D, simple and clear interface for any type of player (user), saving a long apprenticeship to start playing. The project is to store the user moves and perform machine learning that at the end the player can see how the NPC will do playing alone, based on the moves made earlier, that is, the further play computer learn about game. This work also shows the importance of evaluating forms of artificial intelligence for computer games.
KEYWORDS: Artificial Intelligence. KNN. NPC. Electronic game.

INTRODUÇÃO

A Inteligência Artificial (IA) faz com que a máquina tenha a capacidade de criar estratégias, aprender, reconhecer padrões ou encontrar as melhores soluções possíveis, simulando a forma de raciocínio de um ser humano, porém

[1] Bacharel em Ciência da Computação pela Universidade do Sagrado Coração - Bauru. E-mail: bruno.bricce@gmail.com
[2] Mestrado em Ciência da Computação pela Universidade Federal de São Carlos, Brasil (2006). E-mail: patrick.silva@usc.br
[3] Mestrado em Ciência da Computação pela Universidade Estadual Paulista Júlio de Mesquita Filho, Brasil (2014). E-mail: henmartins@gmail.com
[4] Doutorado em Agronomia (Energia na Agricultura) pela Universidade Estadual Paulista Júlio de Mesquita Filho, Brasil(2009). E-mail: egsilva@usc.br

este raciocínio é implementado em um computador. (RUSSEL; NORVIG, 2003).

No âmbito de jogos computacionais a Inteligência Artificial tem forte presença, permitindo uma interação mais dinâmica ao jogador, transmitindo a sensação de competitividade entre homem e máquina. (SANTAELLA; FEITOZA, 2009). Essa é certamente uma característica desejável, até mesmo porque, pela própria evolução dos jogos digitais, que antes tinham um alto grau de previsibilidade, atualmente contam com jogadores que cada vez mais exigem um nível de dificuldade compatível com a de um ser humano. É comum a utilização de IA em jogos que possuem NPCs (Non Player Characters), que são personagens que têm movimentação própria, ou seja, não são controlados pelo jogador (humano). Isso vem ocorrendo devido à necessidade cada vez maior de realismo para o usuário, pois se não existe uma inteligência nos NPCs, o jogo poderá se tornar monótono, limitando a experiência do jogador em relação aos desafios de um jogo. (OSORIO, 2007).

Uma das formas de se modelar os NPCs como agentes inteligentes para a resolução de problemas é encará-los como uma entidade que procura raciocinar, tentando atingir sua meta, por meio de ações inteligentes. Uma das técnicas que podem ser usadas para atingir esse objetivo é o Algoritmo KNN ou Nearest Neighbour Retrieval (Vizinho Mais Próximo). Trata-se de uma técnica muito utilizada para comparação e estimativa de casos existentes. Através de uma base de casos de ações ocorridas, por exemplo, o KNN consegue supor qual seria o resultado de determinado movimento apenas comparando o que já foi feito anteriormente.

Segundo Lagemann (1998), um Raciocínio Baseado em Casos, acontece principalmente ao acumularem-se novas experiências em sua memória e na correta indexação dos problemas. Em jogos eletrônicos ele pode ser usado para leitura do cenário e aprendizado do computador, o que motiva pesquisas e implementações envolvendo esta técnica.

Deste modo, o estudo da IA aplicada aos jogos torna-se relevante do ponto de vista acadêmico, por permitir a criação de jogos muito mais interativos e interessantes.

1. INTELIGÊNCIA ARTIFICIAL APLICADA A JOGOS

Marília tem uma economia de grande força, com indústrias, comércio e prestadoras de serviços que são destaques no município, no país e internacionalmente, com empresas que distribuem seus produtos para o mercado nacional e internacional. Conhecida como “Capital Nacional do Alimento”, o parque industrial mariliense conta com cerca de mil e cem empresas do setor alimentício, metalúrgico, construção, têxtil, gráfico e plástico, entre outras. Nestlé, Marilan, Dori e Sasazaki são algumas das empresas que se destacam no município. Estes resultados são em decorrência da década de 70, que houve um ciclo industrial no município com a instalação de novas indústrias, principalmente na área alimentícia e metalúrgica. Com a posterior instalação de vários cursos universitários. A cidade atrai muitos jovens. (Dados ...,2014)

No setor comercial, Marília dispõe de lojas dos mais variados segmentos. O município possui dois shoppings centers, galeria, além de um centro comercial com calçadão híbrido, atraindo consumidores de toda a região, num raio de até 100 quilômetros. (Dados...,2014)

O setor agropecuário também tem participação no município com café, amendoim, melancia, borracha, coco, laranja, manga, maracujá, cana-de-açúcar, mandioca, milho, sendo culturas produzidas na zona rural. A suinocultura, a bovinocultura (corte e leite) e avicultura (corte e produção de ovos) também tem seu espaço na economia mariliense. (Dados...,2014)

2. ALGORITMO KNN

Segundo Fernandes (2005), o algoritmo KNN ou Nearest Neighbour Retrieval (Vizinho mais próximo) é uma técnica simples que pode resolver um problema se baseando na sua distância com os casos existentes.

Segundo Russel e Norvig (2003), a ideia chave desse modelo é que a propriedade de qualquer ponto de entrada especifica tem probabilidade de serem semelhantes às propriedades de pontos na vizinhança. Esse algoritmo se dá durante a fase de teste/classificação, onde o algoritmo faz uso dos K-

vizinhos mais próximos, para estimar a classe de um novo padrão X, o algoritmo KNN calcula os K-vizinhos mais próximos a X e classifica-o como sendo da classe que aparece com maior frequência dentre os seus K-vizinhos.

De acordo com Lagemann (1999 citado por FERNANDES, 2005, p. 43), "[...] os aspectos de definição e identificação dos índices é fator fundamental para uma recuperação de sucesso.". Garantindo estes aspectos, a técnica de busca irá indicar em qual região do espaço de busca o problema em questão está inserido, sendo o próximo passo encontrar os casos mais parecidos usando comparação e valorização.

Cabe destacar duas características importantes do algoritmo K-NN: a regra de classificação e a função que calcula a distância entre dois pontos (instâncias). A regra de classificação informa de que modo o algoritmo vai tratar a importância de cada um dos k elementos selecionados – os k mais próximos. Assim, procura-se classificar x atribuindo a ele o rótulo representado mais frequentemente dentre as k amostras mais próximas. Já a função de distância serve para medir a distância entre dois elementos de forma a poder identificar quais são os mais próximos.

Para efetuar o cálculo do algoritmo KNN podem ser utilizadas diversas medidas de distância, mas de acordo com Russel e Norvig (2003), a distância euclidiana (Figura 1) é a mais simples para ser usada, principalmente em casos bidimensionais. A formula é representada da seguinte maneira: D = Distância euclidiana; X e Y = pontos de entrada de dados que serão calculados; ∑ = somatória.

Figura 1. Fórmula da distância euclidiana.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Fonte: Elaborada pelo autor.

3. METODOLOGIA

Este trabalho de pesquisa foi dividido em duas etapas: fundamentação teórica e desenvolvimento de software. Na fundamentação teórica foram

abordadas, teorias, algoritmos e ferramentas computacionais necessárias ao desenvolvimento do jogo proposto. A metodologia adotada nesta etapa consistiu, basicamente, no estudo dos conceitos, teorias, algoritmos e métodos computacionais relacionados ao tema desta pesquisa: desenvolvimento de jogos utilizando a ferramenta gratuita Unity 3D e algoritmo KNN.

Este levantamento bibliográfico foi baseado em consultas à literatura especializada, incluindo: livros e artigos científicos, bem como consultas a sites da Internet. Basicamente são abordados os seguintes tópicos: histórico dos jogos e Game IA (breve histórico sobre jogos computacionais e para consoles com o intuito de mostrar a evolução destes jogos e das técnicas de inteligência artificiais utilizadas), algoritmo K-NN (conceitos de Inteligência Artificial com foco em algoritmos aprendizado de máquina com o intuito de contextualizar e mostrar a sua utilidade para o projeto), Planejamento de Jogos (caracterizando os passos para criação de um jogo e abordando características do motor de jogos, Unity 3D).

A metodologia utilizada na etapa de desenvolvimento do jogo envolveu as etapas seguintes: Definição das especificações do jogo, produção artística, definição da engine e integração dos elementos artísticos e computacionais.

Segundo Clua e Bittencourt (2005) a parte de especificação do jogo é uma das partes mais importantes no desenvolvimento, assim como não é possível criar um filme antes de ter um roteiro bem elaborado, é impossível desenvolver um jogo sem antes documentar todas as suas especificações. Nesta etapa são definidos os seguintes elementos: roteiro (definição história e estilo do jogo), game design (conceituação artística do jogo, envolvendo principais características dos cenários, personagens, elementos comportamentais e descrições das fases), jogabilidade (regras e desafio proporcionados aos usuários), interface gráfica (modos de interação do usuário com o jogo). Portanto, nesta etapa, foi realizado todo o planejamento envolvendo os elementos bidimensionais como cenário, objetos e os personagens (avatares). Para o desenvolvimento do conteúdo 2D foram utilizadas modelagens 2D, chamadas Sprites, providas por bases gratuitas e algumas delas fornecidas pela própria ferramenta de desenvolvimento, o Unity 3D.

Na etapa de produção artística foram definidas e utilizadas ferramentas computacionais para a produção de elementos como imagens e áudio utilizados durante o jogo bem como elementos de interface da aplicação tais como, botões, janelas e outros componentes gráficos.

A etapa de definição do engine consistiu na escolha do motor de jogo que é o componente de software responsável manipular o hardware gráfico, controlando toda a parte de renderização de elementos, bem como, as entradas do usuário e respostas da aplicação. Este elemento de software lida com o processamento de baixo nível, facilitando o desenvolvimento já que abstrai do desenvolvedor de jogos a necessidade de lidar diretamente com elementos complexos envolvidos na arquitetura de jogos. Neste projeto optou-se pelo motor de jogo Unity3D para o desenvolvimento dos scripts de Inteligência Artificial e controle da aplicação. O projeto foi desenvolvido utilizando Sprites 2D e os scripts para o desenvolvimento dos algoritmos e do controle da aplicação foram programados em linguagem JavaScripit.

3.1 Definições das Especificações do Jogo

3.1.1. Roteiro

O jogo desenvolvido por esse projeto é do gênero tabuleiro, com sua mecânica baseada em turnos, podendo ser jogado pelo usuário ou pelo computador, no caso onde será aplicada a inteligência artificial.

O jogador controla um herói pelo ambiente do tabuleiro e seu objetivo é reunir todos os diamantes da tela, evitando obstáculos.

Cada movimento feito pelo jogador é registrado, após isso o algoritmo K-NN faz uma comparação entre o cenário atual e os registrados no histórico de jogadas, buscando uma similaridade entre os ambientes para assim verificar qual o melhor caminho a ser utilizado pelo NPC, sempre se baseando no modelo mais próximo do que o jogador efetuou. A hipótese é que a reprodução dos movimentos feitos pelos jogadores humanos possa contribuir para que o NPC atinja os objetivos no jogo, desde que haja similaridade entre os cenários considerados.

3.1.2. Definição do Problema

Os diamantes pertencentes ao rei foram roubados e levados a um castelo abandonado.

Para conseguir cumprir sua missão, foi enviado o melhor herói do reino e ele terá que usar toda sua estratégia, desviando sempre dos obstáculos, fazer o menor caminho.

3.1.3. Personagens

A história conta com um protagonista, o herói, que poderá ser controlado pelo jogador ou pelo computador.

Esse personagem terá que se movimentar pelo tabuleiro em movimentos horizontais e verticais, evitando obstáculos predefinidos no cenário.

3.2 A Utilização do Algoritmo K-NN no Controle de Movimento do Personagem

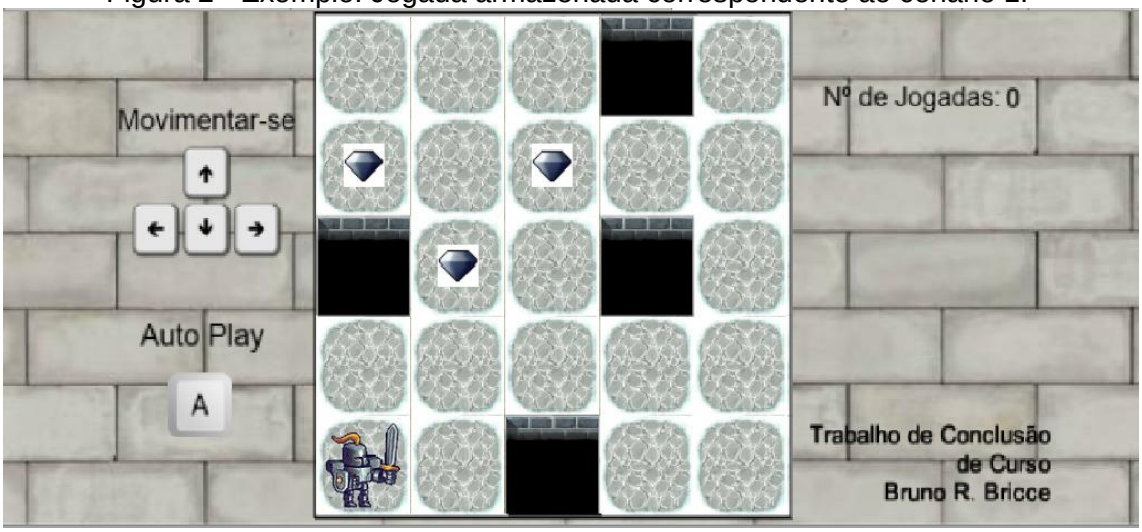
Assim como descrito no capítulo 2, segundo Fernandes (2005), o algoritmo K-NN ou Nearest Neighbour Retrieval (Vizinho mais próximo) é uma técnica simples que pode resolver um problema se baseando na sua distância com os casos existentes.

No jogo o algoritmo KNN se encaixa no Aprendizado de Máquina, pois a partir dos movimentos feitos pelo jogador humano, a máquina irá calcular a distância (similaridade) entre cenários desses para descobrir qual o melhor caminho a ser utilizado.

Detalhando com um exemplo, o jogador humano irá ter as opções de se movimentar na vertical e na horizontal apenas, cada passo que ele der será registrado, após concluir a fase, o jogador terá a opção de solicitar que o computador jogue, selecionando “Auto Play”, através da tecla “A” do teclado. Quanto mais o usuário jogar, mais a máquina irá aprender os seus movimentos em diferentes configurações de cenários, pois na opção do computador jogar, será feita uma comparação utilizando a distância euclidiana para saber qual

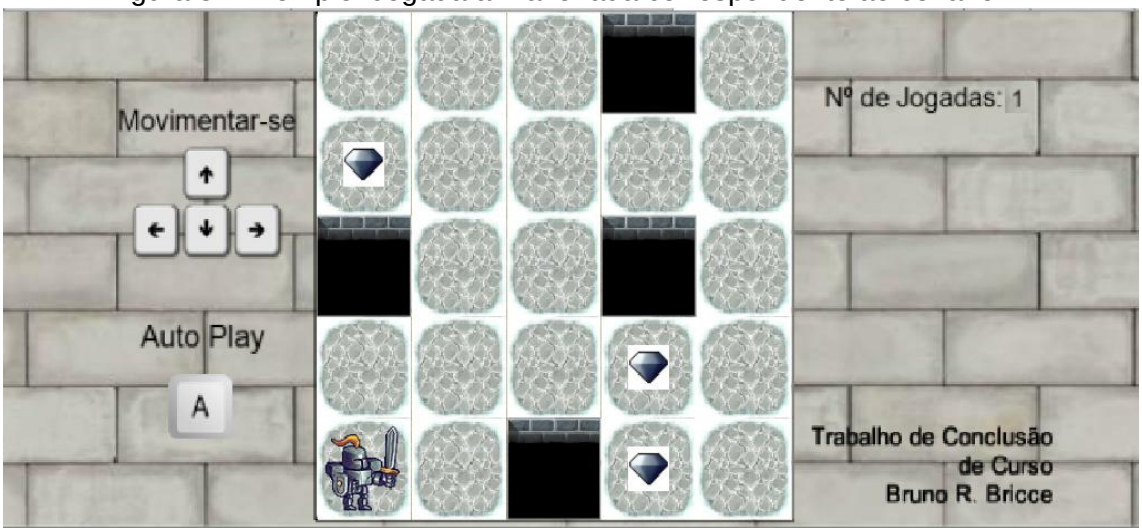
rota é a mais adequada de acordo com a disposição dos elementos no cenário (diamantes).

Figura 2 - Exemplo: Jogada armazenada correspondente ao cenário 1.



Fonte: Elaborada pelo autor.

Figura 3 - Exemplo: Jogada armazenada correspondente ao cenário 2.



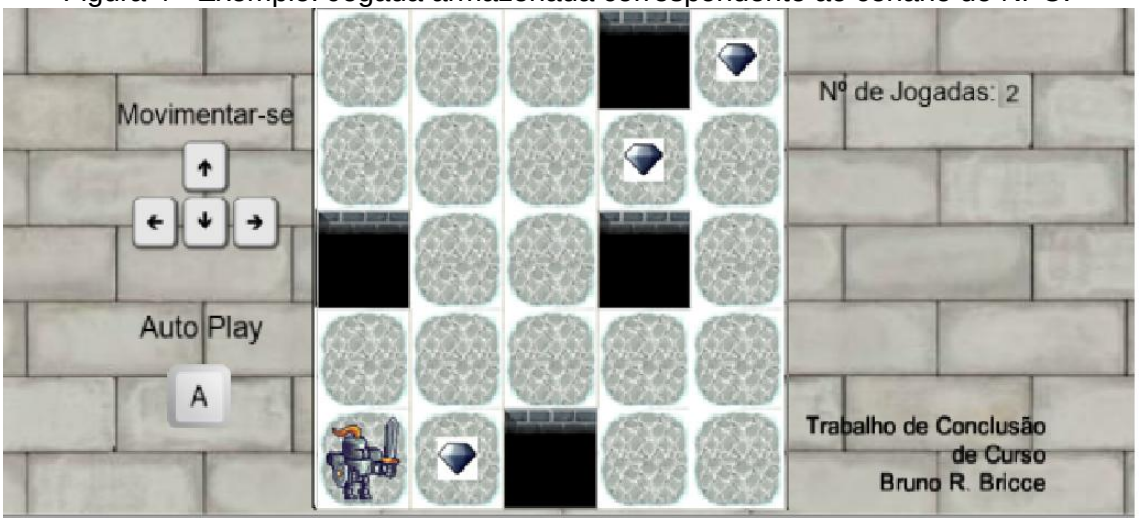
Fonte: Elaborada pelo autor.

Nos exemplos das Figuras 2 e 3 são ilustrados possíveis cenários iniciais encontrados por jogadores, cujos movimentos até a captura com sucesso de todos os diamantes foram armazenados. Cabe destacar que existirá um histórico de jogadas contendo vários cenários e rotas correspondentes armazenados. No código, estas informações vão sendo armazenadas em um Array, à medida que os jogadores usam a aplicação.

Após fechar o jogo, essas informações serão reiniciadas, fazendo com que o jogador inicie novamente a aprendizagem.

Na Figura 4 é mostrado um cenário através do qual o NPC deverá se movimentar automaticamente.

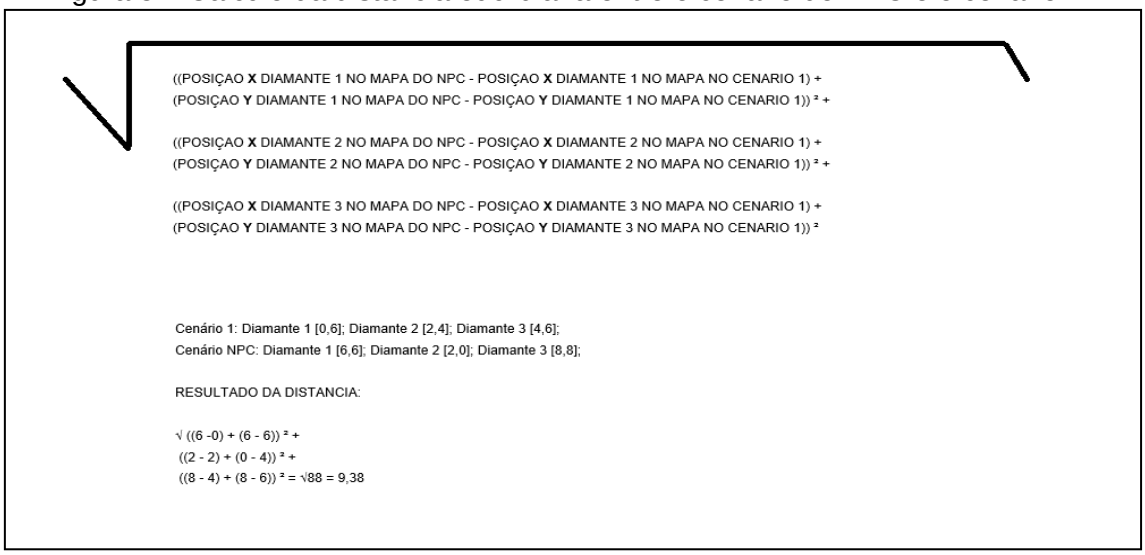
Figura 4 - Exemplo: Jogada armazenada correspondente ao cenário do NPC.



Fonte: Elaborada pelo autor.

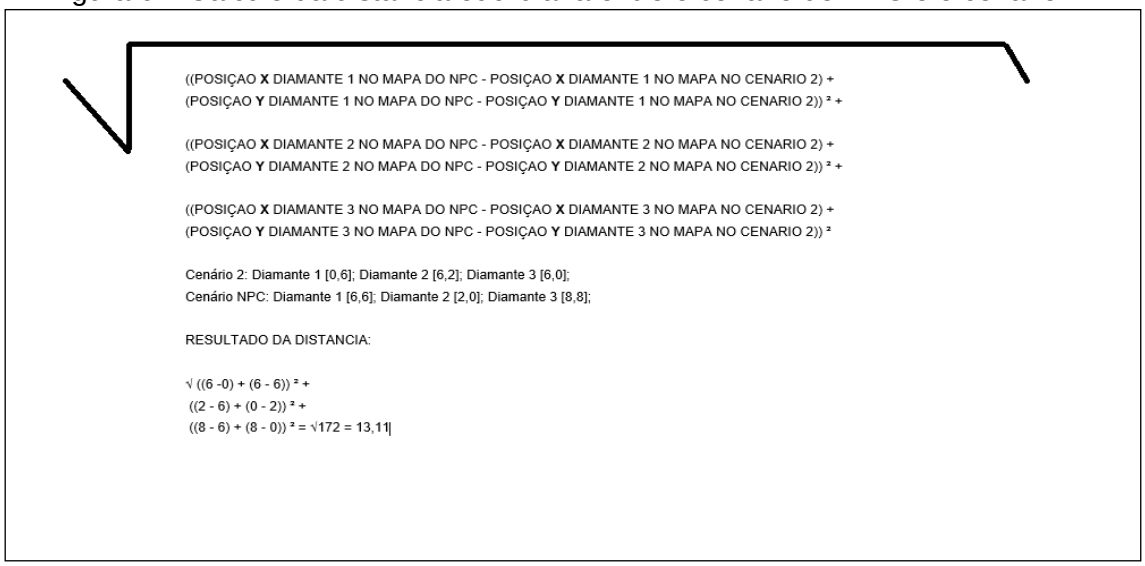
Neste caso, o algoritmo K-NN irá calcular a distância euclidiana entre o cenário atual do NPC (Figura 4) e todos os demais cenários armazenados, calculando a similaridade entre eles. A similaridade aqui adotada irá se basear na localização de cada elemento (diamantes) que compõe o cenário do NPC e os demais. Desta forma, será verificada a posição (x,y) de cada diamante nos dois cenários. Por exemplo, no cenário do NPC (Figura 4) o elemento “diamante 1” encontra-se na posição [6,6] já no cenário 1 (Figura 2) sua posição é [0,6]. A posição dos demais diamantes seria avaliada da mesma maneira e, neste caso a distância euclidiana (descrita na seção 2) calculada entre esses elementos correspondentes nos cenários seria realizada conforme Figuras 5 e 6.

Figura 5 – Cálculo da distância euclidiana entre o cenário do NPC e o cenário 1.



Fonte: Elaborada pelo autor.

Figura 6 – Cálculo da distância euclidiana entre o cenário do NPC e o cenário 2.



Fonte: Elaborada pelo autor.

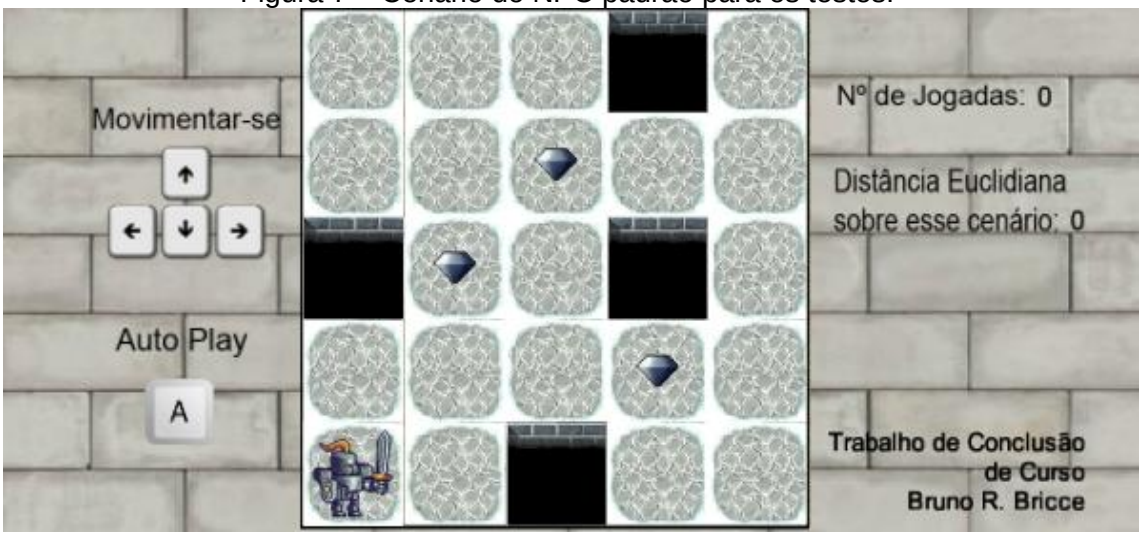
Com base nestes resultados e, considerando apenas os dois cenários, o NPC reproduziria os movimentos do jogador correspondentes ao cenário 1, pois a distância sendo menor, caracteriza que os dois mapas têm mais elementos em posições mais similares.

4. RESULTADOS

Para avaliar o algoritmo implementado, foram realizados quatro testes. No primeiro teste foram realizadas e armazenadas 5 jogadas. O segundo teste

adicionou mais 20 jogadas, totalizando 25 o terceiro contou com 75 jogadas e o último com 125 jogadas. Este aumento incremental de jogadas visou observar a influência do número de elementos armazenados no histórico na qualidade das rotas e na similaridade entre cenários. Os testes foram planejados com o intuito de verificar se houve o aprendizado do NPC durante as jogadas feitas pelo jogador humano e verificar o potencial da estratégia adotada pelo algoritmo proposto. Como base foi utilizado um cenário ilustrado na Figura 7.

Figura 7 – Cenário do NPC padrão para os testes.



Fonte: Elaborada pelo autor.

4.1. Primeiro Teste

No teste de número 1 foram realizadas 5 jogadas antes de iniciar o Auto Play. A distância gerada foi de 7,48 entre o cenário do NPC e o mapa mais similar dentre aqueles armazenados. Uma vez que o mapa mais similar tenha sido escolhido, a rota seguida pelo jogador foi, então, reproduzida no mapa do NPC, para verificar se essa sequência de movimentos permitiria atingir o objetivo do jogo (coletar todos os diamantes). Com a reprodução desta rota foram coletados dois diamantes. Pode-se concluir que o número de jogadas armazenadas não foi o suficiente para garantir a coleta de todos os diamantes.

4.2. Segundo Teste

No teste de número 2 foram realizadas 25 jogadas antes de iniciar o Auto Play do NPC. A distância euclidiana calculada foi de 6,92 entre o cenário do NPC e o mapa mais similar dentre aqueles armazenados. Uma vez que o mapa mais similar tenha sido escolhido, a rota seguida pelo jogador foi, então, reproduzida no mapa do NPC, permitindo a coleta de três diamantes. Pela quantidade de jogadas armazenadas (cinco vezes maior que no teste anterior), o algoritmo se melhor coletando todos os diamantes.

4.3. Terceiro Teste

No terceiro teste foram realizadas 75 jogadas gerando a distância euclidiana (entre o cenário do NPC e aqueles armazenados) no valor de 6,0, o que representa uma melhoria quando comparado dois testes anteriores, com relação à similaridade. A rota escolhida neste teste possibilitou também a coleta de dois diamantes.

4.4. Quarto Teste

No quarto e último teste foram realizadas 125 jogadas (um incremento de 66% de jogadas com relação ao teste anterior), gerando a menor distância euclidiana dentre todos os testes no valor de 4,89 entre o cenário do NPC e aquele escolhido do histórico. Neste teste, o NPC atingiu totalmente seu objetivo, coletando três diamantes.

5. DISCUSSÃO

Após o jogador solicitar que o NPC jogasse no primeiro teste, pode se verificar que o número de jogadas não era o suficiente para gerar cenários bem similares ao do NPC, de modo a permitir que fossem coletados todos os diamantes. Mesmo assim, dois deles (67%) foram coletados.

O segundo teste, além de também ter levado à coleta de 3 diamantes (100%) indica que houve melhor desempenho pela quantidade de mapas armazenados, o que pode ser notado pela diminuição da distância euclidiana.

Os dois últimos testes indicaram melhor desempenho, pois o NPC havia uma maior quantidade de dados para verificação de cada diamante nos mapas anteriores, permitindo, inclusive, que ele recuperasse todos os elementos no último teste. Além disso, observa-se maior similaridade entre os cenários (valor de 4,89) o que nos leva a perceber que existe uma relação direta entre o número de jogadas armazenadas e o valor da similaridade, conforme pode ser visto na Tabela 1 que resume todos os testes realizados.

Analisando todos os testes, foi possível verificar que o algoritmo K-NN implementado no NPC, está condizendo com o que foi proposto, pois a medida que as quantidades de jogadas vão sendo feitas e armazenadas, as similaridades entre os mapas vão aumentando.

A distância euclidiana é calculada toda vez que o jogador pede para o NPC jogar, ativando o Auto Play, dessa maneira pode-se verificar que os valores, apesar de serem diferentes nos mapas jogados, podem ser melhorados à medida que os mapas vão sendo armazenados no histórico de jogadas.

Tabela 1. Resumo dos testes.

Nº de jogadas	Nº de Diamantes Encontrados	Menor distancia euclidiana (similaridade)
5	2	7,48
25	3	6,92
75	2	6,00
125	3	4,89

Fonte: Elaborada pelo autor.

CONSIDERAÇÕES FINAIS

Com base nos resultados obtidos, é possível afirmar que a utilização do algoritmo K-NN no desenvolvimento de jogos pode impulsionar o grau de interatividade destes tipos de aplicação em um novo nível, já que esta técnica

de Inteligência Artificial pode ser aplicada em um NPC, utilizando o Aprendizado de Máquina para, assim, identificar movimentos similares e executar uma ação dentro do jogo.

No geral, o projeto atingiu todos os objetivos propostos, conseguindo principalmente mostrar a utilização do K-NN no contexto de jogos e possibilitando mapear o nível de seu desempenho. Conclui-se, portanto, que esse algoritmo possui uma enorme capacidade e pode ser utilizado dentro de jogos com o intuito de simular e resolver os mais diferentes tipos de problemas, principalmente no Aprendizado de Máquina.

Para trabalhos futuros pode ser proposta a melhoria de alguns aspectos gráficos no jogo, como por exemplo, não fazer desaparecer do cenário o ícone do diamante quando existe a colisão com o jogador. Outra proposta seria aumentar o tamanho do cenário e gerar os diamantes com suas identificações ou mostrar qual está sendo coletado no momento, além de incrementar com outros tipos de obstáculos dinâmicos e utilizá-los também nos cálculos de similaridade. Novos testes também devem ser realizados, com quantidades bem maiores de dados armazenados, a fim de detectar mais claramente a influência do tamanho do histórico na qualidade dos movimentos do NPC.

REFERÊNCIAS BIBLIOGRÁFICAS

CLUA, E. W. G.; BITTENCOURT, J. R. **Desenvolvimento de Jogos 3D**: concepção, design e programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 25., 2005. São Leopoldo. **Anais...** São Leopoldo: UNISINOS, 2005. p. 1313-1356.

FERNANDES, A. M. da R. **Inteligência Artificial**: noções gerais. 2. ed. Visualbooks Florianopolis, 2005.

FUNGE, J. D. **Artificial Intelligence for Computer Games***:* an introduction. Natick: AK Peters, 2004.

KISHIMOTO, A. **Inteligência artificial em jogos eletrônicos**. 2004. 10 f. Trabalho apresentado à disciplina Inteligência Artificial - Universidade Presbiteriana Mackenzie, 2004.
Disponível em: < http://pt.slideshare.net/AndreKishimoto/inteligncia-artificial-em-jogos-eletrnicos-48108829?ref=http://kishimoto.com.br/publications.php>.
Acesso em 05 de maio de 2015.

OSORIO. F. et al. **Inteligência artificial para jogos**: agentes especiais com permissão para matar... e raciocinar! 2007. 4 f. Trabalho de Conclusão de Curso (Graduação em Jogos Digitais) - Universidade do Vale do Rio dos Sinos, São Leopoldo, 2007. Disponível em: <http://osorio.wait4.org/publications/Osorio-et-al-SBGames07-Tutorial.pdf>. Acesso em 09 de março de 2015.

RUSSEL, S.; NORVIG, P. **Inteligência artificial**. Tradução: Stuart Russel, Peter Norvig. Rio de Janeiro: Elsevier, 2011.

SANTAELLA, L.; FEITOZA M. **Mapa do Jogo**: a diversidade cultural dos games. São Paulo: Cengage Learning, 2009.