Simulação de Tráfego Veicular Baseada em Mapas Reais Utilizando o Algoritmo de Colonização do Espaço

Vehicle Traffic Simulation Based on Real Maps Using the Space Colonization Algorithm

Rafael Neves Romeu[1], **Alessandro de Lima Bicho**[1]

[1]Centro de Ciências Computacionais (C3) – Universidade Federal do Rio Grande (FURG)
Rio Grande, RS – Brasil – CEP 96203-900

rafaelromeufurg@gmail.com, albicho@furg.br

Abstract. *This work presents a methodology for the virtual generation of real maps of road networks through OpenStreetMap and the World Geodetic System 1984 (WGS84), enabling the simulation of vehicle traffic through a microscopic computational model called BioTraffic. The proposal consists of developing a graph data structure that allows colonizing the road space through elements called markers, structures necessary for executing the BioTraffic algorithm. The results demonstrate the feasibility of the proposal for simulation based on the extraction of real information from traffic roads.*
Keywords *microscopic model, vehicle traffic simulation, real maps, BioTraffic*

Resumo. *Este trabalho apresenta uma metodologia para a geração virtual de mapas reais de malhas viárias através do OpenStreetMap e da norma geográfica WGS 84, possibilitando simular o tráfego de veículos através de um modelo computacional microscópico denominado BioTraffic. A proposta consiste no desenvolvimento de uma estrutura de dados em grafo que permite colonizar o espaço das vias através de elementos denominados marcadores, estruturas necessárias para a execução do algoritmo BioTraffic. Os resultados demonstram a viabilidade da proposta para a simulação a partir da extração de informações reais de vias de trânsito.*
Palavras-Chave *modelo microscópico, simulação de tráfego veicular, mapas reais, BioTraffic*

1. Introdução

Um dos problemas contemporâneos é a mobilidade urbana, que aflige cidades que cresceram sem estruturar seus Planos de Mobilidade Urbana (PMUs). Esta questão pode ser minimizada, ou mesmo solucionada, através do planejamento urbano. Neste contexto, a utilização da simulação de tráfego veicular possibilita, por exemplo, a detecção de regiões com potencial de congestionamento na malha viária de grandes centros urbanos.

A pesquisa na área da simulação de tráfego veicular também traz contribuições significativas ao ramo de desenvolvimento de jogos digitais, que tem apresentado um público crescente, principalmente a partir da década de 90 [Rabin 2011]. Houve a evolução das plataformas e o surgimento de grandes estúdios de produção de jogos massivos. Um dos principais objetivos dos grandes estúdios é o realismo tanto na modelagem geométrica e renderização dos cenários, como também no comportamento dos NPCs (*Non-Player Characters*), aumentando a imersão do jogador no jogo. Uma das

formas de produzir comportamentos realistas de um indivíduo é o estudo e a proposição de algoritmos que permitem simular computacionalmente o comportamento humano.

Existem algoritmos consolidados em áreas de pesquisa distintas à área da simulação de tráfego veicular que serviram de inspiração para a proposição de modelos comportamentais de pedestres e condutores. O algoritmo proposto por [Runions et al. 2005] para a geração de imagens sintetizadas de padrões de nervura em folhas de plantas sofreu adaptações que produziram, por exemplo, o modelo *BioCrowds* [de L. Bicho et al. 2012], o modelo *BioTraffic* [de Quadros et al. 2021] e o modelo procedural de cidades via algoritmo de colonização do espaço [Vila Nova 2010].

Um aspecto importante na simulação de tráfego veicular é o design das malhas viárias a serem simuladas. O presente trabalho propõe uma metodologia capaz de reproduzir virtualmente mapas reais permitindo simular o tráfego de veículos utilizando o modelo BioTraffic [de Quadros et al. 2021]. O modelo fará uso do *OpenStreetMap*[1] e da norma geográfica WGS 84 [ICAO 2002] para permitir a utilização de mapas reais em simulações.

O artigo está assim organizado. Na Seção 2 são discutidas diferentes abordagens metodológicas para a simulação de veículos, assim como trabalhos relacionados. Na Seção 3 é descrita a metodologia proposta neste trabalho. Na Seção 4 são apresentados os resultados. Por fim, na Seção 5 são destacadas as contribuições e trabalhos futuros.

2. Uma breve fundamentação sobre simulação de tráfego veicular

As primeiras teorias acerca do fluxo de veículos datam da década de 30. O pioneiro no ramo foi o engenheiro americano Bruce Douglas Greenshields em seu trabalho "*A Study of Traffic Capacity*" onde desenvolveu a teoria do fluxo de tráfego, estudo da relação entre o fluxo do tráfego e a velocidade do percurso [Greenshields et al. 1935]. [Lighthill e Whitham 1955] também desenvolveram um modelo para descrever o tráfego de veículos adotando a cinemática. Após estes modelos precursores, o avanço tecnológico permitiu a validação mais precisa em relação aos modelos adotados na época [Neto 2017].

Um dos modelos precursores na área de simulação de tráfego foi o [Nagel e Schreckenberg 1992], que propuseram um modelo microscópico baseado em autômatos celulares. O modelo visa simular uma única via, utilizando um vetor onde cada célula pode ser ocupada por apenas um único veículo. Simulações são realizadas considerando a representação discretizada através de um ambiente constituído por células. Por meio desta abordagem bastante simples foi possível verificar o efeito "para e anda", comumente observado no trânsito real de veículos. Há diversas variações na literatura que propõem melhorias ao modelo original, acrescentando robustez e diversificando o conjunto de situações e cenários [Rickert et al. 1996, Knospe et al. 2000, de Quadros et al. 2021].

Nos modelos mesoscópicos um conjunto de veículos pode ser tratado como uma única unidade dentro sistema. A visão interna a essa unidade, a interação entre os veículos, poderia ser vista como um modelo microscópico enquanto a visão externa adotaria o comportamento de um modelo macroscópico. Nos últimos anos, de forma geral, houve um grande crescimento no interesse e utilização dos

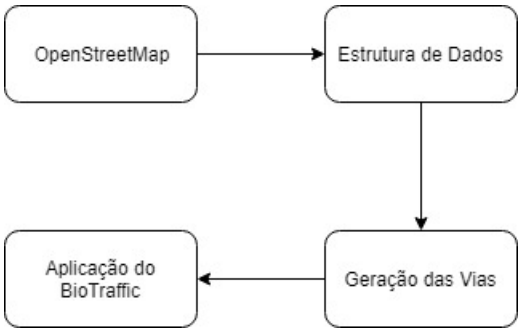
[1]https://www.openstreetmap.org/about, acessado em julho/2024.

modelos mesoscópicos. Com o aumento do poder computacional, os pesquisadores estão experimentando as vantagens de ambos modelos presentes em metodologias mesoscópicas [de Quadros et al. 2021].

3. Metodologia proposta para geração virtual de mapas reais

A metodologia proposta neste trabalho está representa na Figura 1. Nesta figura são apresentadas as principais fases do método: extração de informações de mapas reais, a adequação destas informações em uma estrutura de dados proposta, a geração das vias de trânsito e a aplicação do modelo BioTraffic para simulação do tráfego de veículos [de Quadros et al. 2021].

Figura 1. Diagrama que apresenta as principais fases da metodologia proposta.



3.1. Descrição das vias urbanas por meio do OpenStreetMap

O arquivo XML (*eXtensible Markup Language*)[2] exportado do OpenStreetMap é dividido em três principais estruturas: *nodes*, *ways* e *relations*. *Nodes* são pontos pertencentes a uma ou mais vias, possuem atributos como *id*, *timestamp*, usuário que adicionou o nodo, *id* do usuário, versão, latitude e longitude na norma geografica WGS 84 (*World Geodetic System*, ver Subseção 3.1.1). *Ways* descrevem as vias, além de conter múltiplos *nodes*, também podem conter *tags* que descrevem a via de maneira mais detalhada; as *tags* podem conter o nome das ruas, o tipo de estrada das mesmas e até mesmo o tipo da área: residencial, comercial, rural, etc. E, por fim, as *relations*, que são estruturas utilizadas para organizar um conjunto de *ways* e/ou *nodes* em um todo maior, são utilizadas para descrever áreas como praças, espaços públicos ou até mesmo descrever linhas de ônibus. No modelo as estruturas utilizadas foram as duas primeiras mencionadas. Tendo como objetivo criar a malha das vias baseado nas informações extraídas do OpenStreetMap, uma série de passos foram implementados para atingir a este fim.

3.1.1. O uso de normas cartográficas para a construção das malhas viárias

Para obter uma malha com maior precisão, a latitude e a longitude de todos os nodos foram convertidos da norma cartográfica WGS 84 para o sistemas de coordenadas UTM (*Universal Transverse Mercator*). O WGS 84 é uma norma utilizada em cartografia, atualmente utilizada pelo GPS, Google Earth entre outros, concebido pelo departamento

[2]https://www.w3.org/XML/, acessado em julho/2024.

de defesa dos Estados Unidos. O sistema geodésico foi constituído com base no modelo gravitacional da terra [ICAO 2002].

Para os cálculos que serão explicados posteriormente, havia a necessidade de uma norma cartográfica bidimensional, para que os cálculos necessários para a construção da malha fossem simplificados e para a malha não sofrer possíveis distorções. No propósito de converter a latitude e a longitude dos nodos, foi implementado uma *API Rest*[3] em *nodeJS*, que consome o microserviço especializado em conversões de normas cartográficas *LatLong.net*. A arquitetura da API é extremamente simples, onde o sistema consiste em três passos:

1. Receber o arquivo XML do mapa extraído e ler os campos de latitude e longitude;
2. Enviar múltiplas requisições ao microserviço responsável pelo cálculo da conversão de WGS84 para UTM;
3. Receber a resposta do serviço com os campos latitude, longitude e *time zone*, substituindo no arquivo XML.

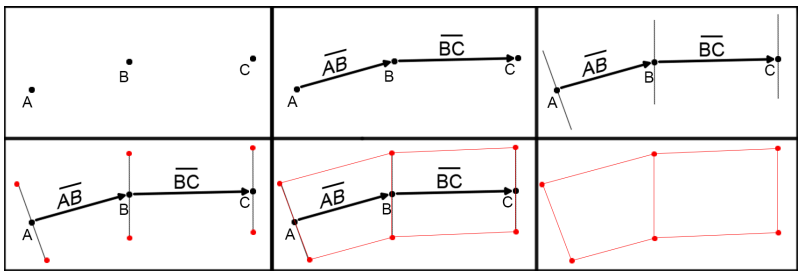
Sendo assim, o primeiro tratamento a ser dado ao extrair o arquivo do OpenStreetMap, para que possa ser corretamente utilizado pelo modelo, é enviá-lo para a API responsável por converter as latitudes e longitudes.

3.1.2. Construção das malhas viárias

A partir dos pontos geográficos dos nodos que descrevem as vias e respectivas larguras das mesmas, é possível calcular os pontos que descrevem a malha da via. O algoritmo responsável por esses cálculos possui os seguintes passos:

1. Calcular o vetor direção que “aponta” para o próximo nodo da sequência da via;
2. Calcular o vetor normalizado ortogonal ao vetor direção;
3. Dado o ponto, latitude e longitude do nodo, e o vetor ortogonal, é então descrita uma reta;
4. A partir da equação paramétrica da reta, é possível extrair dois pontos que descrevem parte da via, utilizando-se da informação da sua largura.

Figura 2. Mapeamento de vias simples.

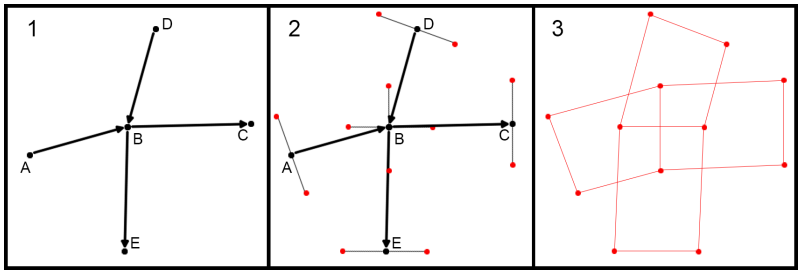


Ao repetir os passos acima para todos os nodos de uma via, todos os pontos necessários para sua descrição são obtidos, conforme a Figura 2. Entretanto, apesar

[3]Uma interface de programação de aplicativos (API) que segue os princípios de design Rest (*Representational State Transfer*).

de bastante preciso para vias simples, o algoritmo supracitado possui limitações para o cruzamento de vias. Quando duas vias se interceptam formando uma bifurcação, o modelo sobrepõe as malhas (Figura 3), impossibilitando o lançamento uniforme dos marcadores sobre as vias com o algoritmo *dart-throwing* (ver Seção 3.2.4).

Figura 3. Sobreposição das malhas das vias.



A fim de obter resultados mais precisos, e não haver a sobreposição de vias, houve algumas decisões de projeto. A primeira delas foi classificar os nodos de acordo com a quantidade vias a que ele pertence. Nodos que pertencem a uma única via foram definidos *nodos via* e os nodos que pertencem a mais de uma via foram definidos *nodos esquina* ou *nodos bifurcação*. Importante notar que os *nodos via* possuirão um único vetor direção, enquanto os nodos bifurcação possuirão dois vetores direção. Essa mudança acarreta em um aumento na complexidade do algoritmo responsável pelo cálculo dos pontos da malha das vias. Temos, portanto, o seguinte algoritmo:

1. Verifica a classificação do nodo; se for *nodo via*, segue os passos supracitados do algoritmo anterior. Porém, se for *nodo esquina*, calcula todos os vetores que apontam para os próximos nodos;
2. Calcular os vetores normalizados ortogonais aos vetores direcionais;
3. Dado o ponto e os vetores normalizados ortogonais, duas retas são criadas;
4. Após deslocar essas retas perpendicularmente em ambas direções, como mostra a Figura 4 parte 3, é possível calcular os quatro pontos de interseção entre essas retas, destacados nas partes 4 e 5.

Após todos os pontos calculados, temos as condições necessárias para construir a malha completa da via. A Figura 5 mostra, na parte superior, a maneira correta de agrupar os pontos para formar polígonos de quatro lados que descrevem as vias. A parte inferior da imagem retrata um problema comum ao gerar a malha de polígonos: enumerar os pontos em uma ordem incorreta pode acarretar uma malha "retorcida", e que não descreve o objeto desejado.

3.2. Desenvolvimento dos algoritmos responsáveis pela simulação veicular

A presente seção apresenta o conjunto de algoritmos que compõem o modelo responsável pela simulação. As subseções estão divididas em: preparação das estruturas de dados e ambiente; geração dos veículos e, por fim, o cálculo dos agentes (veículos) nas vias.

3.2.1. Construção da estrutura de dados: grafo

O objetivo desta etapa é extrair as informações do XML disponibilizado pelo OpenStreetMap para criar uma estrutura de dados que mantenha a forma das vias, assim

Figura 4. Algoritmo para vias que interseccionam (malha complexa).

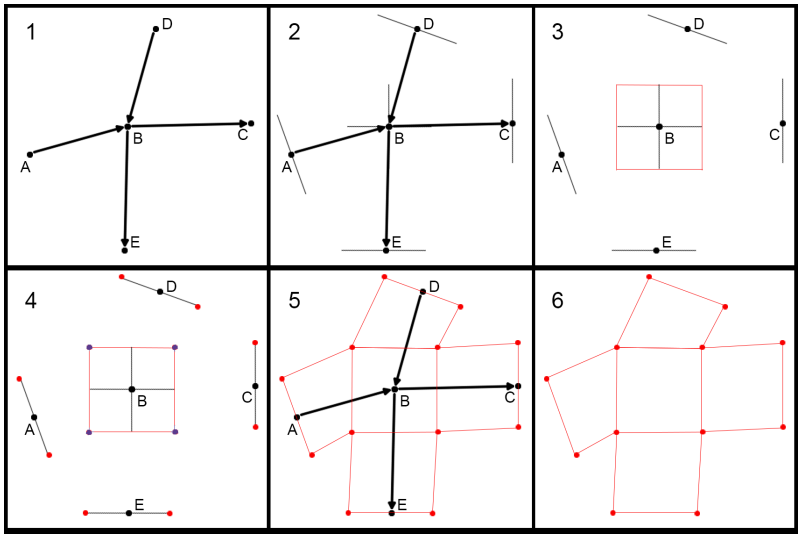
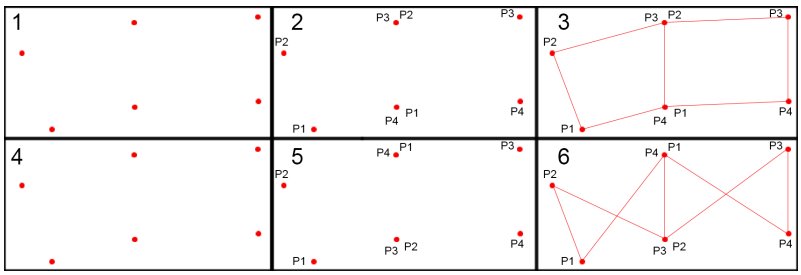


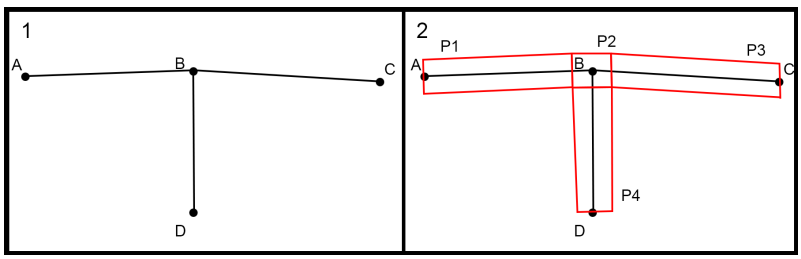
Figura 5. Geração dos polígonos para posterior preechimento pelo algoritmo *dart-throwing*.



como as informações importantes para a simulação e para o desenvolvimento de um ambiente real. Cabe salientar que a estrutura de dados grafo foi a decisão proposta.

Para o modelo foi utilizado um grafo não direcional onde os vértices são representados pelos nodos extraídos do mapa, e as arestas são mapeadas de acordo com os caminhos representados no arquivo XML obtido do OpenStreetMap. Os nodos seguem o mesmo padrão supracitado na seção anterior, podendo ser nodos do tipo *via* ou *bifurcação*. Os polígonos são associados aos arcos, e os polígonos gerados nas bifurcações são associados aos nodos do tipo *bifurcação*, como mostra a Figura 6.

Figura 6. Associação dos polígonos ao grafo.



3.2.2. Algoritmo Dijkstra para cálculo do caminho mínimo

O algoritmo Dijkstra resolve o problema de caminhos mais curtos de única origem em um grafo para o caso no qual todos os pesos de arestas são não negativos [Cormen et al. 2012].

Após calculado os menores caminhos de todos os nodos para todos os nodos, esses dados são gravados em um arquivo XML para que, futuramente, sejam utilizados pelos veículos para traçarem suas respectivas rotas.

3.2.3. Locais de lançamento dos veículos no grafo

Para o lançamento dos veículos, alguns nodos foram escolhidos como nodos iniciais (de partida). Todos os veículos lançados nas vias terão como partida algum desses nodos, definidos de maneira aleatória. Portanto, para um nodo ser escolhido como nodo inicial ele terá de cumprir dois pré-requisitos: não ser do tipo nodo *esquina*, e pertencer ao início de um *way*. Esses dois requisitos garantem que os veículos não surgirão no meio das vias, prejudicando a simulação.

3.2.4. Algoritmo para preenchimento de marcadores nas vias: *Dart-throwing*

Antes de inicializar os cálculos para deslocamento dos veículos nas vias, é necessário realizar algumas preparações na malha viária. A principal e mais custosa delas é preencher as vias com os marcadores; para tanto, foi implementado uma versão do algoritmo *Dart-throwing* [Cook 1986].

O algoritmo *Dart-throwing* [Cook 1986] foi o primeiro algoritmo a gerar pontos distribuídos de acordo com a distribuição de *Poisson*. O algoritmo rejeita novos pontos que não satisfazem a distância mínima entre os pontos já lançados na via. O processo continua até uma condição de parada seja satisfeita; essa condição pode ser baseada na densidade de pontos, no número total de pontos ou no tempo de execução do algoritmo *Dart-throwing*. Como parâmetros no algoritmo, devem ser informados a área a ser preenchida e a distância mínima entre os pontos (marcadores) [Lagae 2009]. Para aplicação no modelo, foi utilizada como condição de parada da execução do algoritmo a densidade de pontos nos polígonos que definem as vias. A utilização de um parâmetro como densidade garante que os polígonos de diferentes tamanhos possuam a mesma distribuição de marcadores, garantindo que a simulação dos veículos não seja prejudicada.

3.2.5. Geração dos veículos nas vias

Após todos os marcadores serem lançados nas vias, é necessário gerar os veículos a serem simulados. Todos os veículos são inicializados antes da simulação. A inicialização de cada veículo consiste em:

- Estabelecer aleatoriamente um nodo origem, baseado nas entradas das vias (Seção 3.2.3);
- Estabelecer aleatoriamente um nodo destino;

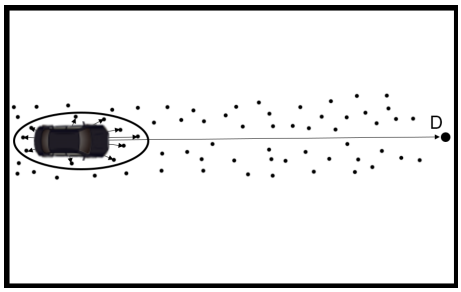
- Encontrar o menor caminho até o nodo destino, calculado na Seção 3.2.2;
- Definir aleatoriamente o tamanho do veículo.

No modelo proposto no presente trabalho, as áreas de percepção dos veículos são representadas por elipses, de modo a simplificar os cálculos e aumentar o desempenho das animações. Os veículos considerados "grandes" possuem o eixo maior 4.85 m e o eixo menor 1.85 m, enquanto os veículos "pequenos" possuem o eixo maior 3.85 m e o eixo menor 1.68 m. A área de percepção consiste em uma área elíptica ao redor dos veículos no qual os marcadores existentes dentro dessa área são utilizados para o cálculo da movimentação. A ideia de utilizar uma área de percepção advém do conceito de espaço pessoal desenvolvido no modelo [de L. Bicho et al. 2012].

3.2.6. Simulação do tráfego de veículos

Uma vez os veículos nas vias e os marcadores posicionados, é possível calcular a movimentação dos agentes (veículos). Os marcadores informam os espaços disponíveis para que os veículos possam trafegar; os pontos (marcadores) no interior da zona de percepção do agente são utilizados para o cálculo do vetor resultante da soma vetorial ponderada. Os vetores cujas orientações estejam alinhadas ao objetivo recebem maiores ponderações na soma vetorial, enquanto o contrário recebem menores ponderações. De forma similar ao que ocorre no modelo proposto por [de Quadros et al. 2021], os agentes aumentam a sua velocidade até o limite estabelecido na via (utilizando o princípio de *terrain reasoning*).

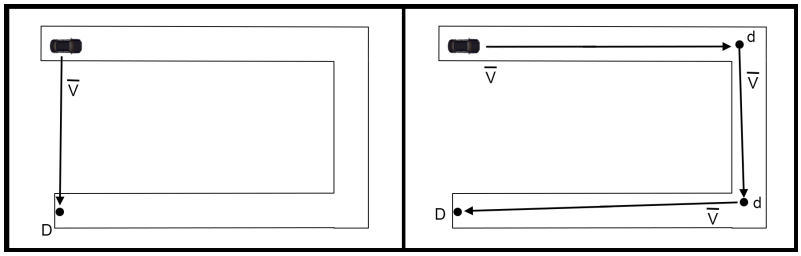
Figura 7. Relação entre o agente (veículo) e os marcadores na área de percepção.



No modelo *BioTraffic* proposto por Quadros, não há a utilização de malhas viárias complexas oriundas de mapas reais. Há apenas vias simples, possuindo um ponto de partida e um ponto de destino. Neste trabalho, foram inseridos destinos parciais que, quando percorridos em sequência, direcionam o agente (veículo) ao seu destino final. Essa foi a estratégia utilizada para trafegar por caminhos complexos, evitando o problema representado na Figura 8.

De modo a otimizar a simulação, cada veículo considera apenas os marcadores presentes no polígono no qual ele está trafegando. Portanto, durante o deslocamento, o veículo deve verificar se está se aproximando da área de transição de um polígono para o próximo e, se estiver, os marcadores do próximo polígono devem ser levados em consideração no cálculo da movimentação do agente.

Figura 8. Destinos parciais e final de um agente (veículo).



3.2.7. Visualização gráfica da simulação

Durante a simulação, todos os dados referentes a movimentação de cada veículo são armazenados em um arquivo XML. O arquivo recebe uma descrição de cada veículo e a sua posição, orientação e velocidade em cada quadro da simulação. A visualização gráfica é reproduzida baseada nos dados armazenados no arquivo XML. A proposta da abordagem de simulação *offline* foi definida devido ao custo computacional do algoritmo de simulação utilizado, sendo executado em *single-thread*, tornando-se impeditivo para a realização dos cálculos e da visualização em tempo real.

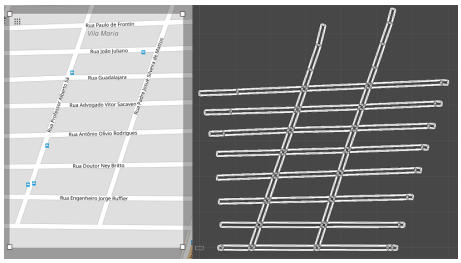
Para desenvolver a parte gráfica foi utilizada a Unity[4]. A *game engine* é responsável por ler o arquivo XML gerado durante a simulação, preparar o ambiente e reproduzir a movimentação dos agentes em cada quadro da animação.

4. Resultados

A visualização pode ser dividida em duas partes: a visualização das vias e da simulação. Durante o desenvolvimento deste trabalho, havia o planejamento de apresentar a visualização tridimensional do cenário e dos agentes na simulação; porém, o tempo de desenvolvimento permitiu apenas a apresentação de formas geométricas básicas bidimensionais dos veículos e do ambiente de simulação.

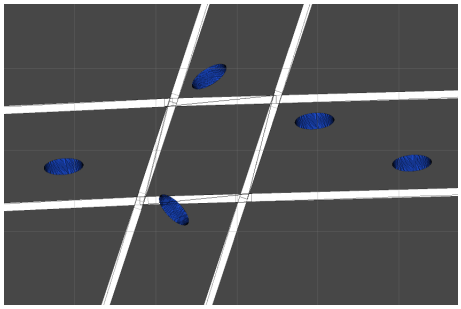
As vias são traçadas baseadas nos polígonos calculados na Seção 3.1.2, e os veículos são representados como elipses. A Figura 9 mostra uma comparação do mapa real obtido no OpenStreetMap com as vias calculadas a partir dos conceitos explicitados na Seção 3.1.2. A Figura 10 representa um *snapshot* de um quadro da simulação no ambiente Unity.

Figura 9. Mapa do OpenStreetMap e as vias geradas na Unity.



[4]https://unity.com/pt/products/unity-engine, acessado em julho/2024.

Figura 10. Simulação do tráfego de veículos na Unity.



5. Conclusão

A presente seção apresenta as contribuições do modelo, e analisa possíveis trabalhos futuros.

5.1. Contribuições

O modelo proposto neste trabalho apresenta as seguintes contribuições, sendo algumas possíveis de aperfeiçoamento:

- Extração e geração de vias a partir de mapas reais: a proposição de uma metodologia para a modelagem das vias urbanas baseada em mapas reais utilizando-se do OpenStreetMap e da norma cartográfica WGS 84, assim como os resultados obtidos em simulação, são contribuições importantes;
- Algoritmo para colonização do espaço adaptado para simulação de veículos (*BioTraffic*): a simulação de veículos utilizando-se como base o algoritmo de colonização do espaço adaptado não é novidade; porém, a utilização dessa abordagem na escala presente nesse artigo é um passo importante no amadurecimento deste conceito.

5.2. Trabalhos Futuros

Considerando a metodologia e os resultados apresentados, foram identificadas possíveis funcionalidades que podem ser desenvolvidas em trabalhos futuros:

- Simular semáforos em vias: a localização dos semáforos nas vias é de fácil extração no OpenStreetMap. Uma possível abordagem para controle semafórico é a adoção de redes de Petri [Paravisi et al. 2006];
- Simular diferentes leis de trânsito: para que o modelo tenha maior realismo, é necessário que os agentes (veículos) apresentem comportamentos de acordo com as leis de trânsito do local simulado;
- Simular comportamentos para os condutores: para maior realismo, reproduzir nas simulações agentes que apresentem agressividade, altruísmo, egoísmo, etc;
- Implementar *multithreading* em GPU/CPU: para maior desempenho durante a simulação, a paralelização poderia ser abordada em futuras implementações, de modo a otimização dos cálculos para geração das vias e da movimentação dos agentes;
- Gerar ambientes tridimensionais: neste artigo tanto o ambiente quanto os agentes são bidimensionais; porém, é possível desenvolver visualizações tridimensionais.

Referências

Cook, R. L. (1986). Stochastic sampling in computer graphics. *ACM Trans. Graph.*, 5(1):51–72.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2012). *Algoritmos: Teoria e Prática*. GEN LTC, 3 edition.

de L. Bicho, A., Rodrigues, R. A., Musse, S. R., Jung, C. R., Paravisi, M., e Magalhães, L. P. (2012). Simulating crowds based on a space colonization algorithm. *Computers & Graphics*, 36(2):70–79. Virtual Reality in Brazil 2011.

de Quadros, C. E. P., Adamatti, D. F., e de L. Bicho, A. (2021). Biotraffic: a bio-inspired behavioral model to vehicle traffic simulation. In *20th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 29–38.

Greenshields, B. D., Bibbins, J. R., Channing, W. S., e Miller, H. H. (1935). A study of traffic capacity. In *Proceedings of Highway Research Board*, volume 14, pages 448–477. Washington, DC.

ICAO (2002). *World Geodetic System – 1984 (WGS-84) Manual*. International Civil Aviation Organization (ICAO), 2 edition.

Knospe, W., Santen, L., Schadschneider, A., e Schreckenberg, M. (2000). Towards a realistic microscopic description of highway traffic. *Journal of Physics A: Mathematical and General*, 33(48):L477.

Lagae, A. (2009). *Wang Tiles in Computer Graphics*. Springer Cham, 1 edition.

Lighthill, M. J. e Whitham, G. B. (1955). On kinematic waves II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society A*, 229(1178):317–345.

Nagel, K. e Schreckenberg, M. (1992). A cellular automaton model for freeway traffic. *Journal de Physique I France*, 2(12):2221–2229.

Neto, J. C. (2017). A Teoria do Fluxo de Tráfego. Notas de Engenharia de Tráfego Urbano, Escola de Engenharia, Univ. Presbiteriana Mackenzie.

Paravisi, M., Musse, S. R., e de Lima Bicho, A. (2006). Modelagem e simulação do tráfego de veículos e controle semafórico em um ambiente virtual. *VETOR – Revista de Ciências Exatas e Engenharias*, 16(2):16–37.

Rabin, S. (2011). *Introdução ao Desenvolvimento de Games. Volume 1: Entendendo o Universo dos Jogos*. Cengage Learning, 1 edition.

Rickert, M., Nagel, K., Schreckenberg, M., e Latour, A. (1996). Two lane traffic simulations using cellular automata. *Physica A: Statistical Mechanics and its Applications*, 231(4):534–550.

Runions, A., Fuhrer, M., Lane, B., Federl, P., Rolland-Lagan, A.-G., e Prusinkiewicz, P. (2005). Modeling and visualization of leaf venation patterns. In *Proceedings of ACM SIGGRAPH ´05*, pages 702–711, New York, NY, USA. Association for Computing Machinery.

Vila Nova, A. B. (2010). Modelagem procedural de cidades via algoritmo de colonização de espaço. Master's thesis, Programa de Pós-graduação em Ciência da Computação – Universidade Federal de Pernambuco.