Aplicação de Modelos de Linguagem de Grande Escala e Agentes Inteligentes com SUMO para Automatização de Simulações de Tráfego Urbano

Matheus Andrade [*], **Thaís Medeiros** [*], **Marianne Silva** [**], **Ivanovitch Silva** [*]

[*] *Universidade Federal do Rio Grande do Norte/PPgEEC, Natal-RN*
[**] *Universidade Federal de Alagoas/SI, Penedo-AL*

E-mails: {matheus.diniz.122, thais.araujo.707}@ufrn.edu.br, marianne.silva@penedo.ufal.br, ivanovitch.silva@ufrn.br

Abstract: Simulations are important for addressing urban mobility challenges in cities and the automotive industry, allowing for an understanding of transportation systems. However, the complexity of modeling urban scenarios and the volume of data involved pose some challenges, highlighting the need for automated methods. Thus, the aim of this work is to explore the capability of Large Language Models (LLMs) in automating the urban simulation process, providing an alternative to traditional methods. To achieve this goal, the integration of LLMs with the Simulation of Urban MObility (SUMO) is proposed, using the LangChain tool. This integration allows for the construction of agents capable of configuring simulations, analyzing results, and proposing adaptive solutions. A case study is conducted to validate the proposed approach, using real data from the urban area of Natal-RN city. The results highlight the importance and relevance of applying language models in conducting simulations and interpreting generated data. These findings reinforce the significance of LLMs in automating and optimizing processes, contributing to the development of more effective and sustainable solutions for smart cities.

Resumo:
As simulações são importantes para enfrentar os desafios de mobilidade urbana nas cidades e indústria automotiva, permitindo uma compreensão dos sistemas de transporte. No entanto, a complexidade na modelagem de cenários urbanos e o volume de dados envolvidos representam alguns desafios, destacando a necessidade de métodos automatizados. Deste modo, o objetivo deste trabalho é explorar a capacidade dos Modelos de Linguagem de Grande Escala (LLMs) em automatizar o processo de simulação urbana, oferecendo uma alternativa para os métodos tradicionais. Para alcançar esse objetivo, propõe-se a integração de LLMs com o *Simulation of Urban MObility* (SUMO), utilizando a ferramenta LangChain. Essa integração permite a construção de agentes capazes de configurar simulações, analisar resultados e propor soluções adaptativas. Um estudo de caso é conduzido para validar a abordagem proposta, utilizando dados reais da área urbana da cidade de Natal-RN. Os resultados destacam a importância e a pertinência da aplicação de modelos de linguagem na condução de simulações e na interpretação dos dados gerados. Essas descobertas reforçam a relevância dos LLMs na automatização e otimização de processos, contribuindo para o desenvolvimento de soluções mais eficazes e sustentáveis para as cidades inteligentes.

Keywords: Smart Cities; Automotive Industry; Urban Mobility; SUMO; LLM.
Palavras-chaves: Cidades Inteligentes; Indústria Automotiva; Mobilidade Urbana; SUMO; LLM.

1. INTRODUÇÃO

As Cidades Inteligentes e indústrias automotivas tem enfrentado desafios relacionados à simulação de tráfego urbano (Savastano et al., 2023; Guo et al., 2024). Com o aumento da urbanização e a crescente demanda por sistemas de transporte eficientes, tornou-se essencial desenvolver soluções para modelar e entender o comportamento do tráfego em ambientes urbanos (WHIG, 2023). O *Simulation of Urban MObility* (SUMO) é uma ferramenta que se insere nesse contexto, permitindo a coleta, análise e interpretação de dados de tráfego (Behrisch et al., 2011).

Desse modo, uma das dificuldades encontradas nesses ambientes é a complexidade e o custo envolvido na criação das simulações realistas de tráfego (Kayisu et al., 2024). As abordagens tradicionais exigem a configuração manual de parâmetros e a coleta extensiva de dados, o que é demorado e propenso a erros (Rocco Di Torrepadula et al., 2023).

Logo, a análise manual dos resultados dessas simulações podem ser trabalhosas e sujeitas a viés humano (Andrade et al., 2024).

Diante desse cenário, surge a necessidade de desenvolver métodos automatizados para a simulação de tráfego urbano (Güzay et al., 2023). Para isso, uma solução é a utilização dos Modelos de Linguagem de Grande Escala (LLMs), como uma promissora alternativa para otimizar e automatizar esse processo (Güzay et al., 2023; Tang et al., 2024). Os LLMs são modelos de inteligência artificial treinados em grandes volumes de texto, capazes de entender e gerar texto em linguagem natural com precisão e fluidez (Yao et al., 2024).

O objetivo deste trabalho é explorar a capacidade dos LLMs em automatizar o processo de simulação urbana, oferecendo uma alternativa efetiva, precisa e econômica para os métodos tradicionais. Para alcançar esse objetivo, propõe-se uma metodologia que integra os LLMs com o SUMO, utilizando a ferramenta `LangChain` [1] . Essa integração permite a construção de agentes capazes de configurar simulações de forma autônoma, analisar os resultados e propor soluções adaptativas com base nos padrões identificados, representando um avanço significativo na modelagem e na gestão do tráfego urbano.

Os resultados deste artigo destacam a importância e a pertinência da aplicação de modelos de linguagem na condução de simulações e na interpretabilidade dos dados gerados. Essas descobertas reforçam a relevância dos LLMs, na automatização e otimização de processos, como a simulação de tráfego urbano. Assim, pode contribuir para a compreensão e aprimoramento dos sistemas de transporte urbano, além de proporcionar o desenvolvimento de soluções mais eficazes e sustentáveis para as cidades inteligentes.

Por fim, o restante deste artigo encontra-se organizado da seguinte forma: a Seção 2 apresenta trabalhos relacionados; a Seção 3 especifica a abordagem proposta; a Seção 4 descreve o estudo de caso realizado para comparar as soluções; a Seção 5 discute os principais resultados obtidos; e, finalmente, a Seção 6 apresenta as considerações finais e sugere caminhos promissores para trabalhos futuros.

2. TRABALHOS RELACIONADOS

Esta seção oferece uma visão geral dos trabalhos relacionados que influenciaram a pesquisa realizada e contribuíram para o desenvolvimento da solução proposta. A utilização da simulação desempenha um papel importante na aceleração do desenvolvimento de plataformas de veículos, fornecendo uma alternativa segura, econômica e eficaz aos métodos de teste tradicionais (Liu et al., 2023; Alghodhaifi and Lakshmanan, 2021). Além disso, a integração de Modelos de Linguagem de Grande Escala (LLMs) pode facilitar a modelagem e execução das simulações, oferecendo novas perspectivas para a análise de dados e a geração de cenários realistas de tráfego (Güzay et al., 2023; Tang et al., 2024).

No trabalho de Tang et al. (2024), apresentam um método que utiliza Modelos LLM, como o GPT-4, para implementar o controle de onda verde em vias urbanas. A abordagem combina LLM com políticas de controle de semáforos, visando explorar o potencial na área de controle de tráfego. Deste modo, utilizaram o SUMO para construir o problema de controle de semáforos e verificaram que o LLM pode realizar a análise e solução desse problema. A política de controle de semáforos é gerada de forma interativa por meio de linguagem natural, reduzindo a carga de análise de dados e computação dos gestores de tráfego. Os resultados experimentais demonstraram que o processo gera um controle de onda verde na via que pode melhorar a velocidade média da via.

Similarmente, os autores Güzay et al. (2023) propõem uma abordagem que utiliza LLM, API do GPT-4, para gerar cenários de tráfegos. Para facilitar a interação, desenvolveram uma aplicação intermediária que se comunica com a Application Programming Interface (API) e processa as respostas do modelo. O processo começa com o usuário definindo o cenário de simulação de forma linguística, descrevendo características como número de interseções, posição das vias, comprimento das estradas, número de faixas e semáforos. Em seguida, a aplicação intermediária adiciona as diretivas necessárias ao *prompt* fornecido pelo usuário. Uma vez recebida a resposta do modelo, a aplicação processa e a transforma em arquivos de simulação SUMO formatados em *Extensible Markup Language* (XML). Com base nessa resposta válida, a aplicação gera os arquivos de simulação SUMO necessários para representar o cenário de tráfego descrito pelo usuário. Essa abordagem permite que os usuários criem cenários de simulação SUMO de forma mais intuitiva e flexível, utilizando linguagem natural para descrever suas necessidades, enquanto a aplicação intermediária garante a precisão e adequação dos *prompts* fornecidos para a geração dos arquivos de simulação. Nossa abordagem proposta de integração do SUMO com o LLM difere ao utilizar dados de um domínio real como entrada para o processo. Isso permite gerar automaticamente análises a partir desses dados, sem a necessidade de entender detalhes sobre como realizar-lá e/ou criar visualizações para faciliar o entendimento.

Complementar, Da et al. (2024) exploram a capacidade de inferência dos LLMs pré-treinados para compreender como a dinâmica do tráfego varia com condições climáticas, estados de tráfego e tipos de estrada. Ao estar ciente dessas variações, as ações das políticas são tomadas com base em dinâmicas realistas, auxiliando o agente a aprender de forma realista. Realizaram experimentos em quatro cenários diferentes para demonstrar a eficácia da abordagem proposta, conhecida como `PromptGAT`, em mitigar a lacuna de desempenho do aprendizado por reforço em ambientes reais.

No entanto, cada um desses trabalhos enfoca aspectos específicos dentro do domínio da simulação e LLM. O trabalho proposto visa preencher uma lacuna explorando uma abordagem que integra dados de um domínio real de tráfego e técnicas de modelagem de linguagem para a geração automatizada de cenários de simulação SUMO. Essa abordagem representa uma convergência entre a análise de dados do mundo real e as capacidades preditivas dos LLMs, permitindo uma simulação mais precisa e adaptável do tráfego urbano.

[1] https://www.langchain.com/

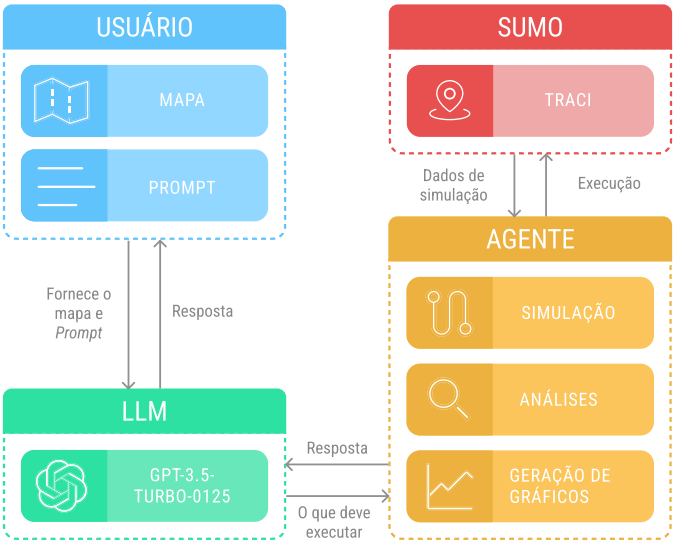

Figura 1. Fluxo da metodologia proposta.

3. ABORDAGEM PROPOSTA

A abordagem proposta visa fornecer uma metodologia para automatizar simulações urbanas por meio do uso de LLMs e SUMO (ver Figura 1). Ela é composta por quatro módulos:

- *Usuário* — responsável por configurar o cenário de tráfego e gerenciar as simulações utilizando *prompts* interpretados pelo modelo LLM;
- *LLM* — interpreta os comandos do usuário e direciona-o para as funcionalidades apropriadas;
- *Agente* — controla e traduz os comandos interpretados pelo modelo LLM em ações específicas;
- *SUMO* — responsável pela simulação do cenário configurado pelo usuário.

Primeiramente, o usuário define o cenário de tráfego por meio da ferramenta *OpenStreetMap Web Wizard* — a partir da execução do script Python `osmWebWizard.py`, disponibilizado no *toolkit* do SUMO. Ao selecionar a região desejada para a simulação, os arquivos resultantes, gerados pelo *OpenStreetMap Web Wizard*, são armazenados em uma pasta acessível tanto para o módulo *Agente* quanto para a biblioteca `TraCI`.

Além disso, o usuário também realiza o gerenciamento das simulações por meio de *prompts*, que são interpretados pelo modelo de linguagem da família GPT-3.5 Turbo da OpenAI. Esse conjunto de modelos foi selecionado devido a sua capacidade de interpretar comandos e custo-benefício aceitável para a automação na metodologia proposta, em termos de requisições de sua API.

Em seguida, ao processar o *prompt* fornecido pelo usuário, o modelo *LLM* facilita a seleção das funcionalidades desenvolvidas com o auxílio da biblioteca `LangChain`. Ele atua como um intermediário inteligente, interpretando os comandos e direcionando-os para as funcionalidades apropriadas.

Além do *prompt*, também utilizou-se a técnica de *few-shot* (*prompt engineering*), a qual visa fornecer exemplos específicos para orientar o modelo na compreensão dos diferentes tipos de comandos (Medeiros et al., 2023). Ao fornecer tais amostras pré-definidas, o modelo consegue compreender as instruções do usuário com mais precisão. Essa estratégia garante que o módulo *Agente* utilize a funcionalidade adequada para lidar com o *prompt* do usuário.

Nessa abordagem, utilizou-se o seguinte *few-shot*:

```
Você é um assistente muito poderoso. Se a con-
sulta do usuário for sobre informações da si-
mulação, você classificará a consulta exa-
tamente entre 'análise', 'estatísticas veí-
culo id' ou 'plotar coluna veículo id'. Exemplo:
- 'Analise os dados da simulação.': 'análise'
- 'Obtenha as estatísticas do veí-
culo com id 1.': 'estatísticas veículo id 1'
- 'Plote os dados de velocidade do veí-
culo com id 1.': 'plotar velocidade veículo id 1'
```

Por sua vez, o módulo *Agente* atua como um controlador central, recebendo os comandos interpretados pelo modelo de linguagem e traduzindo-os em ações específicas. Ele também é responsável por acionar as funcionalidade desenvolvidas com a biblioteca `LangChain`, assegurando que cada uma seja aplicada da forma correta.

Desse modo, foram concebidas funcionalidades utilizando a classe `BaseTool` como base para o desenvolvimento delas. Essa classe facilita a análise e validação dos parâmetros de entrada das funcionalidades criadas, além de lidar com erros de validação, caso as entradas fornecidas não sejam adequadas. Assim, essa abordagem proporciona uma estrutura para a criação e implementação de novas funcionalidades, garantindo a consistência e confiabilidade das operações realizadas dentro do sistema.

Outrossim, voltadas para automação, análise e visualização, essas funcionalidades foram desenvolvidas com o objetivo de diminuir o tempo de obtenção de informações sobre as simulações realizadas. São elas:

a) `SumoInitializationTool`: configura e executa simulações no SUMO de acordo com os *prompts* fornecidos. Para essa funcionalidade, deve ser especificado o caminho do arquivo de configuração da simulação, bem como podem ser determinados os parâmetros a serem coletados dos veículos. Desse modo, ela utilizará a biblioteca `TraCI` para executar a tarefa e gerenciar a simulação.
b) `SimulationDataAnalysisTool`: coleta e interpreta os dados gerados durante as simulações (*dataset* e *logs*). Caso haja mais de uma simulação gerada, a ferramenta busca a mais recente.
c) `SimulationDataPlottingTool`: utiliza o *dataset* criado ao final da simulação para confeccionar os gráficos de um veículo especificado via *prompt*. Da mesma forma que *SimulationDataAnalysisTool*, em caso de mais de uma simulação ocorrida, escolhe-se a mais recente.

No módulo *SUMO*, ocorre a simulação propriamente dita, na qual o cenário previamente configurado pelo usuário é reproduzido, considerando diversas variáveis — a exemplo de tráfego, tipos de veículos e padrões de movimentação. A integração com a biblioteca `TraCI` permite que *Agente* controle e colete os dados relevantes sobre a simulação.

Dessa forma, *SUMO* atua como um meio pelo qual as funcionalidades podem operar.

4. ESTUDO DE CASO

O estudo de caso tem como objetivo avaliar a abordagem proposta, explorando a eficácia do uso de agentes com funcionalidades personalizadas da biblioteca `LangChain` para automatizar as simulações de tráfego urbano utilizando o SUMO.

4.1 Preparação

Para configurar o cenário de tráfego, a cidade de Natal, no estado do Rio Grande do Norte, Brasil, foi escolhida como local de estudo. A região delimitada na ferramenta *OpenStreetMap Web Wizard* [2] pode ser observada na Figura 2.

Figura 2. Região selecionada para simulação.

Posteriormente, foram definidas perguntas específicas para analisar cada funcionalidade implementada, as quais foram utilizadas nos *prompts*. As perguntas específicas estão detalhadas na Tabela 1.

Tabela 1. Funcionalidades e Perguntas Correspondentes.

Funcionalidade	**Pergunta no *Prompt***
`SumoInitializationTool`	- "Inicialize uma simulação com o arquivo de configuração em ./data/sumo_files/osm.sumocfg." - "Inicialize uma simulação com o arquivo de configuração em ./data/sumo_files/osm.sumocfg com 8 threads, iniciando no tempo 0 e terminando no tempo 100, com um máximo de 20.000 veículos simultaneamente. Obtenha dos veículos a velocidade, aceleração, consumo de combustível, geolocalização, odômetro e emissões de CO_2."
`SimulationDataAnalysisTool`	"Analise os dados da simulação." "Obtenha as estatísticas do veículo com id 1."
`SimulationDataPlottingTool`	"Plote os dados de velocidade para o veículo com id 1."

[2] https://sumo.dlr.de/docs/Tutorials/OSMWebWizard.html

4.2 Execução

Nesta subseção, descreve-se os passos necessários para a execução da abordagem, que consiste na criação e interação com um cenário de tráfego urbano utilizando o SUMO. Os passos detalhados para realizar essa tarefa, são:

(1) **Cenário Criado** — delimitação da área de interesse resultou na criação de um cenário de tráfego com configurações padrão ajustadas conforme necessário para atender aos requisitos específicos do estudo. A criação do cenário gerou vários arquivos, incluindo arquivos de viagens dos veículos e um arquivo .sumocfg contendo as configurações da simulação.

(2) **Preparação do arquivos para simulação e interação com o cenário** — com o cenário preparado, foi instalada a interface TraCI, que permite interagir com o SUMO via Python, necessária para controlar a simulação e realizar análises em tempo real.

(3) **Implementação das funcionalidades em Python** — a implementação das funcionalidades desejadas foi realizada em Python. Primeiramente, uma classe foi criada para representar como a entrada seria interpretada, utilizando a classe BaseModel da biblioteca Pydantic. Esta classe de entrada foi fornecida ao args_schema dentro da classe da funcionalidade a ser implementada. O método especial _run() foi então implementado para definir o comportamento da funcionalidade, utilizando a biblioteca TraCI para gerenciar a simulação.

(4) **Configuração do agente inteligente** — para possibilitar a comunicação com o modelo gpt-3.5-turbo-0125, foi implementado o comportamento do agente utilizando a classe ChatOpenAI. A classe ChatPromptTemplate foi empregada para aplicar a técnica de few-shot, enquanto o método bind_tools() foi utilizado para permitir a chamada da funcionalidade implementada. Após a configuração, o agente foi criado e a classe OpenAIToolsAgentOutputParser foi adicionada para que a mensagem resultante da execução do agente fosse analisada e direcionada corretamente.

(5) **Execução da Simulação** — finalmente, uma instância de AgentExecutor foi criada para executar os *prompts* enviados pelo usuário, utilizando o método stream(). Com essa configuração, foi possível iniciar a simulação, coletar dados sobre o comportamento dos veículos e realizar as análises. Os dados incluíam informações sobre velocidade, aceleração, consumo de combustível, geolocalização, odômetro e emissões de CO2, permitindo uma compreensão do tráfego na área estudada.

5. RESULTADOS E DISCUSSÃO

Esta seção tem como objetivo discutir os resultados obtidos na aplicação do estudo de caso apresentado na Seção 4. Assim, busca-se analisar as respostas obtidas ao introduzir os *prompts* e avaliar a eficácia das funcionalidades desenvolvidas.

Primeiramente, foi analisada a funcionalidade responsável por realizar as simulações. Para isso, foram desenvolvidos dois *prompts*, (1) determina diversas característica a serem levadas em consideração no momento de executar a simulação. (2) determina somente o caminho para o arquivo

de configuração que deve ser utilizado, como pode ser observado na Tabela 2

Tabela 2. Descrição dos *prompts* e saídas para o SumoInitializationTool.

Nº	*Prompt*	Saída
1	Inicialize uma simulação com o arquivo de configuração em ./data/-sumo_files/osm.sumocfg com 8 *threads*, iniciando no tempo 0 e terminando no tempo 100, com um máximo de 20.000 veículos simultaneamente. Obtenha dos veículos a velocidade, aceleração, consumo de combustível, geolocalização, odômetro e emissões de CO2.	A simulação foi inicializada com a configuração fornecida. A simulação foi executada com sucesso usando 8 *threads*, começando no tempo 0 e terminando no tempo 100, simulando um máximo de 20.000 veículos. Dados como velocidade, aceleração, consumo de combustível, geolocalização, odômetro e emissões de CO_2 foram coletados dos veículos.
2	Inicialize uma simulação com o arquivo de configuração em ./data/-sumo_files/osm.sumocfg.	A simulação foi inicializada com uso do arquivo ./data/-sumo_files/osm.sumocfg.

Na Tabela 2, apresenta-se o primeiro caso de uso da funcionalidade `SumoInitializationTool`, no qual foram empregadas 8 *threads* para a execução da simulação. Complementarmente, foram gerados, após a simulação, três arquivos. São eles: `.CSV` com as informações dos veículos retornadas pelo SUMO; `main.log` criado pela abordagem, contém os registros de cada um dos 100 passos; e `simulation.log` também retornada pelo SUMO com os detalhes da simulação. O `simulation.log` dessa execução, é observado conforme estrutura apresentada abaixo.

```
Starting server on port 49067
Loading net-file from
'./data/sumo\_files/osm.net.xml.gz'
... done (7560ms).

Loading done.
Simulation version 1.19.0 started with time: 0.00.
Simulation ended at time: 100.00
Reason: TraCI requested termination.
Performance:
 Duration: 2723.00s
 TraCI-Duration: 2404.49s
 Real time factor: 0.0367243
 UPS: 922.239666
Vehicles:
 Inserted: 31080 (Loaded: 33682)
 Running: 31036
 Waiting: 2602
 Teleports: 51 (Collisions: 51)
 Emergency Braking: 12
Statistics (avg of 44):
 RouteLength: 777.93
 Speed: 10.98
 Duration: 70.95
 WaitingTime: 3.45
 TimeLoss: 21.26
 DepartDelay: 1.40
```

Os resultados da simulação pelo log indicam a execução bem-sucedida da abordagem. Além disso, são fornecidas estatísticas sobre os veículos, incluindo o número total de veículos inseridos e carregados, assim como aqueles que relataram ou experimentaram uma frenagem de emergência. Também são apresentadas estatísticas médias para os veículos que completaram suas rotas, incluindo a média do comprimento da rota, velocidade média, entre outras métricas relevantes.

Tabela 3. Descrição dos *prompts* e saídas para o SimulationDataAnalysisTool.

Nº	*Prompt*	Saída
1	Analise os dados da simulação.	Os dados de simulação foram analisados. • Arquivo: ./data/-sumo_files/osm.net.xml.gz • Versão: 1.19.0 • Duração: 2723.00s • Teleportes: 51 • Colisões: 51 • Frenagens de Emergência: 12
2	Obtenha as estatísticas do veículo com id 1.	Aqui estão as estatísticas para o veículo de ID 1: • Velocidade Média: 11.93 • Aceleração Média: 0.12 • Passos Totais: 100 • Distância Total: 1193.35 • Velocidade Máxima: 20.70 • Aceleração Máxima: 2.60 • Consumo de Combustível Total: 134712.81

Tabela 4. Descrição dos *prompts* e saídas para o SimulationDataPlottingTool.

Prompt	Saída
Plote os dados de velocidade para o veículo com id 1.	Eu plotei os dados de velocidade para o veículo com ID 1.

Em seguida, para analisar a funcionalidade de interpretação dos dados gerados durante a simulação, foram desenvolvidos os *prompts* apresentados na Tabela 3. Assim, é perceptível que o primeiro *prompt* capturou várias informações relacionadas às características da simulação. Na qual valida à capacidade da funcionalidade implementada de acessar o arquivo `simulation.log` gerado no primeiro caso da Tabela 2.

Além disso, no segundo *prompt* da Tabela 3, é evidente a presença de métricas estatísticas para alguns dos dados solicitados no primeiro *prompt* da Tabela 2. Corroborando mais uma vez com a implementação da funcionalidade e a maneira como ela foi implementada.

Por fim, para analisar a capacidade de geração de gráficos por parte do agente com uso da funcionalidade `SimulationDataPlottingTool`, foi criado o *prompt*, conforme Tabela 4.

A análise da saída registrada na Tabela 4 ratifica que o agente obteve com êxito as informações disponibilizadas

ao término da simulação. Portanto, viabilizou a produção do gráfico ilustrado na Figura 3, proporcionando uma representação visual dos dados coletados.

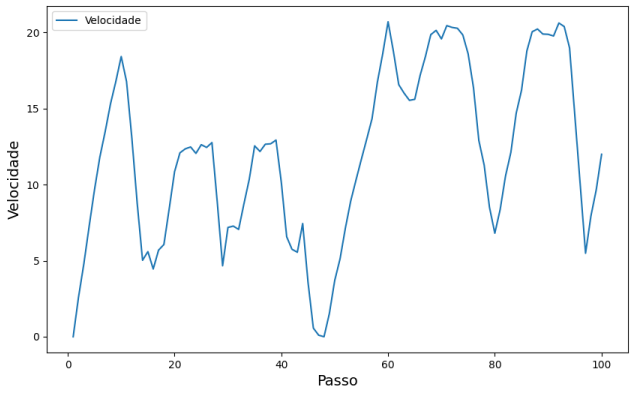

Figura 3. Gráfico de velocidade gerado pelo agente para o veículo de ID 1

6. CONCLUSÃO

Este artigo apresenta uma metodologia para automatizar simulações urbanas por meio do uso de LLMs, Agentes e SUMO. O estudo de caso realizado demonstrou que a metodologia pode ser um passo significativo rumo ao aprimoramento dos sistemas de transporte urbano, oferecendo suporte para o desenvolvimento de soluções mais eficazes e sustentáveis para as cidades inteligentes. Os *prompts* desenvolvidos permitiram configurar simulações de forma precisa e eficiente, evidenciando a capacidade do sistema em interpretar comandos e direcioná-los para as funcionalidades apropriadas.

Além disso, a análise das saídas geradas pelos *prompts* demonstrou que o agente foi capaz de acessar e interpretar os dados gerados durante a simulação, fornecendo informações detalhadas e estatísticas relevantes. Ademais, a geração de gráficos a partir dos dados coletados demonstrou a capacidade do sistema em apresentar visualmente os resultados da simulação, facilitando a compreensão e interpretação dos dados.

Trabalhos futuros incluem, mas não se limitam a: desenvolver novas funcionalidades para o agente, como a capacidade de lidar com situações mais complexas ou a integração de técnicas de aprendizado de máquina para uma tomada de decisão; explorar técnicas para otimizar o desempenho e a escalabilidade da simulação, visando reduzir o tempo de execução e os recursos computacionais necessários; e Investigar o uso de outros modelos de linguagem.

AGRADECIMENTOS

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001, do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) Processo nº 405531/2022-2 e da Fundação de Desenvolvimento da Pesquisa – Fundep Rota 2030/Linha VI 29271.01.01/2023.03-00.

REFERÊNCIAS

Alghodhaifi, H. and Lakshmanan, S. (2021). Autonomous vehicle evaluation: A comprehensive survey on modeling and simulation approaches. *IEEE Access*, 9, 151531–151566. doi:10.1109/ACCESS.2021.3125620.

Andrade, M., Medeiros, M., Medeiros, T., Azevedo, M., Silva, M., Costa, D.G., and Silva, I. (2024). On the use of biofuels for cleaner cities: Assessing vehicular pollution through digital twins and machine learning algorithms. *Sustainability*, 16(2), 708.

Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D. (2011). Sumo–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind.

Da, L., Gao, M., Mei, H., and Wei, H. (2024). Prompt to transfer: Sim-to-real transfer for traffic signal control with prompt learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 82–90.

Guo, P., Chen, Z., Yang, Y., and Miao, R. (2024). A multistage simulation-optimization-integrated methodology framework for user-oriented electric vehicle carsharing reallocation under dynamic price subsidy. *Energy*, 290, 130207.

Güzay, Ç., Özdemir, E., and Kara, Y. (2023). A generative ai-driven application: Use of large language models for traffic scenario generation. In *2023 14th International Conference on Electrical and Electronics Engineering (ELECO)*, 1–6. IEEE.

Kayisu, A.K., Kambale, W.V., Benarbia, T., Bokoro, P.N., and Kyamakya, K. (2024). A comprehensive literature review on artificial dataset generation for repositioning challenges in shared electric automated and connected mobility. *Symmetry*, 16(1), 128.

Liu, S.T., Chang, C., Huang, Y.H., Lin, T.H., Chiu, J., and Lee, J.L. (2023). Development and test of abs/tcs controller with dual-axis dynamometer hil platform. Technical report, SAE Technical Paper.

Medeiros, T., Medeiros, M., Azevedo, M., Silva, M., Silva, I., and Costa, D.G. (2023). Analysis of language-model-powered chatbots for query resolution in pdf-based automotive manuals. *Vehicles*, 5(4), 1384–1399.

Rocco Di Torrepadula, F., Russo, D., Di Martino, S., Mazzocca, N., and Sannino, P. (2023). Using sumo towards proactive public mobility: Some lessons learned. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Sustainable Mobility*, 51–58.

Savastano, M., Suciu, M.C., Gorelova, I., and Stativă, G.A. (2023). How smart is mobility in smart cities? an analysis of citizens' value perceptions through ict applications. *Cities*, 132, 104071.

Tang, Y., Dai, X., and Lv, Y. (2024). Large language model-assisted arterial traffic signal control. *IEEE Journal of Radio Frequency Identification*.

WHIG, P. (2023). Harnessing ai for sustainable traffic management: Enhancing efficiency, safety, and mobility in smart cities. *International Scientific Journal for Research*, 5(5).

Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., and Zhang, Y. (2024). A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 100211.