UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Johnata Brayan Souza Soares

NAVWARESET: UM CONJUNTO DE DADOS DE NAVEGAÇÃO ROBÓTICA SOCIALMENTE CONFORMISTA E NÃO CONFORMISTA

Belo Horizonte
2025

Johnata Brayan Souza Soares

NAVWARESET: UM CONJUNTO DE DADOS DE NAVEGAÇÃO ROBÓTICA SOCIALMENTE CONFORMISTA E NÃO CONFORMISTA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Armando Alves Neto

Belo Horizonte
2025

Soares, Johnata Brayan Souza.
S676n Navwareset [recurso eletrônico] : um conjunto de dados de navegação robótica socialmente conformista e não conformista / Johnata Brayan Souza Soares. – 2025.
1 recurso online (72 f. : il., color.) : pdf.

Orientador: Armando Alves Neto.

Dissertação (mestrado) – Universidade Federal de Minas Gerais, Escola de Engenharia.

Inclui bibliografia.

1. Engenharia elétrica – Teses. 2. Pesquisa na Internet – Teses. 3. Robótica – Teses. 4. Inteligência artificial – Teses. 5. Interação homem-máquina – Teses. I. Alves Neto, Armando. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pelo bibliotecário Marcio A. A. Gomes CRB/6 2812
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS

Escola de Engenharia

COLEGIADO DO CURSO DE GRADUAÇÃO / PÓS-GRADUAÇÃO EM Engenharia Elétrica

FOLHA DE APROVAÇÃO

"Navwareset: Um Conjunto de Dados de Navegação Robótica Socialmente Conformista e Não Conformista"

Johnata Brayan Souza Soares

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 04 de agosto de 2025.

Por:

Prof. Dr. Armando Alves Neto
DELT (UFMG) - Orientador

Prof. Dr. Vinícius Mariano Gonçalves
DEE (UFMG)

Prof. Dr. Douglas Guimarães Macharet
DCC (UFMG)

Documento assinado eletronicamente por **Armando Alves Neto**, **Professor do Magistério Superior**, em 04/08/2025, às 17:18, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Vinicius Mariano Goncalves**, **Membro**, em 04/08/2025, às 17:26, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Douglas Guimaraes Macharet**, **Professor do**

Magistério Superior, em 06/08/2025, às 08:31, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4416831** e o código CRC **D11201A1**.

Referência: Processo nº 23072.246102/2025-98 SEI nº 4416831

Este trabalho é dedicado a todas as crianças
que sonham em serem cientistas.

Agradecimentos

A realização desta dissertação de mestrado não teria sido possível sem o apoio e a colaboração de muitas pessoas e instituições, às quais expresso minha profunda gratidão.

Agradeço, primeiramente, ao meu orientador, cujas orientações, paciência e incentivo foram fundamentais ao longo desta jornada. Suas contribuições e conhecimento enriqueceram este trabalho de maneira significativa.

À minha família, por seu amor incondicional, apoio e compreensão em todos os momentos. Vocês são minha base e minha maior inspiração.

Aos colegas e amigos que fiz durante este percurso, agradeço pela troca de ideias, pelo suporte emocional e pelas discussões enriquecedoras que contribuíram para o meu crescimento acadêmico e pessoal.

Expresso também minha gratidão à instituição que me acolheu durante este período, fornecendo os recursos e o ambiente necessários para o desenvolvimento deste trabalho. Agradeço especialmente às agências de fomento que apoiaram financeiramente esta pesquisa.

Por fim, agradeço a todos que, direta ou indiretamente, contribuíram para a realização deste trabalho. A cada um de vocês, meu mais sincero obrigado.

“Em algum lugar, algo incrível está esperando para ser descoberto.” - Carl Sagan

Resumo

Este trabalho apresenta a construção e aplicação do **NavWareSet**, um novo conjunto de dados voltado para o estudo da navegação social de robôs em ambientes internos. O dataset foi obtido a partir de experimentos controlados envolvendo dois robôs (Toyota HSR e Clearpath Jackal) e 17 participantes humanos, englobando sete cenários distintos de interação social. As gravações incluem dados multimodais como nuvens de pontos 3D, imagens RGB-D, vídeo, odometria e comandos de velocidade, além de anotações manuais das trajetórias dos pedestres, totalizando mais de 1000 trilhas humanas e 600 robóticas.

Para tornar os dados úteis na modelagem de interações humano-robô, o modelo de forças sociais (SFM, do inglês Social Force Model) foi implementado no simulador UAIbot, permitindo a simulação física de pedestres em ambientes com obstáculos. Em seguida, foi realizada a calibração dos parâmetros do modelo com base em trajetórias reais extraídas do NavWareSet, por meio de um processo de otimização que minimiza o erro entre as trajetórias simuladas e observadas. Os resultados obtidos mostram que os parâmetros ajustados reproduzem com boa precisão o comportamento dos participantes humanos, e se alinham, com variações justificáveis, a valores encontrados na literatura.

As principais contribuições desta dissertação incluem a disponibilização pública de um conjunto de dados inédito, a implementação prática de um modelo clássico de navegação social, e o desenvolvimento de uma metodologia objetiva para sua calibração com base em dados reais. O trabalho visa fortalecer a base empírica para pesquisas em navegação social de robôs, promovendo reprodutibilidade e comparação padronizada entre algoritmos.

Palavras-chave: navegação social, robótica, aprendizado supervisionado, simulação, Social Forces Model.

Abstract

This work presents the development and application of **NavWareSet**, a new dataset designed for the study of socially-aware robot navigation in indoor environments. The dataset was collected through controlled experiments involving two robots (Toyota HSR and Clearpath Jackal) and 17 human participants, covering seven distinct social interaction scenarios. The recordings include multimodal data such as 3D point clouds, RGB-D images, video, odometry, and velocity commands, along with manually annotated pedestrian trajectories, totaling over 1000 human tracks and 600 robot tracks.

To enable the modeling of human-robot interactions, the Social Force Model (SFM) was implemented within the UAIbot simulator, allowing physical simulation of pedestrians in environments with obstacles. Subsequently, the model parameters were calibrated based on real trajectories extracted from NavWareSet, through an optimization process that minimizes the error between simulated and observed trajectories. The results show that the optimized parameters accurately reproduce human behavior and are consistent—with justifiable variations—with values reported in the literature.

The main contributions of this dissertation include the public release of a novel dataset, the practical implementation of a classical social navigation model, and the development of an objective methodology for parameter identification using real-world data. This work aims to strengthen the empirical foundation for research in socially-aware robot navigation, promoting reproducibility and standardized algorithm comparison.

Keywords: social navigation, robotics, supervised learning, simulation, Social Forces Model.

Lista de Figuras

Lista de Tabelas

Lista de Siglas e Símbolos

Siglas

AMCL	Adaptive Monte Carlo Localization
CVAT	Computer Vision Annotation Tool
GRS	Ground-Truth Recording Station
HRI	Human-Robot Interaction
HSR	Human Support Robot
LiDAR	Light Detection and Ranging
ODOM	Odometry
PCD	Point Cloud Data
RGB-D	Red Green Blue-Depth
ROS	Robot Operating System
SCAND	Socially Compliant Navigation Dataset
SFM	Social Forces Model
SLAM	Simultaneous Localization and Mapping
THÖR	Trajectories and Human-Object Interactions Dataset
UAIbot	Simulador de robótica baseado em navegador

Símbolos

$\vec{r}_i$	Posição do agente i no espaço [m]
$\vec{v}_i$	Velocidade atual do agente i [m/s]
$\vec{v}_i^{\text{des}}$	Velocidade desejada do agente i [m/s]
$\vec{e}_i$	Direção desejada (vetor unitário) do agente i
m_i	Massa do agente i [kg]
τ_i	Tempo de relaxamento do agente i [s]
$\vec{f}_i$	Força social total atuando no agente i [N]
$\vec{f}_{ig}$	Força de atração do agente i ao objetivo [N]
$\vec{f}_{ij}$	Força de interação entre os agentes i e j [N]
$\vec{f}_{ik}$	Força de interação entre o agente i e o obstáculo k [N]

A_i	Intensidade da força repulsiva [N]
B_i	Alcance da força repulsiva [m]
d_{ij}	Distância escalar entre os agentes i e j [m]
$\vec{d}_{ij}$	Vetor de distância entre os agentes i e j
ϕ_{ij}	Ângulo de interação entre os agentes i e j [rad]
$w(\phi)$	Função de ponderação angular [-]
λ_i	Fator de anisotropia direcional [-]
ξ_i	Perturbação estocástica no movimento do agente i
Δt	Intervalo de tempo discreto [s]
x, y, z	Coordenadas cartesianas [m]

Sumário

Capítulo 1

Introdução

O avanço da robótica móvel tem ampliado significativamente o papel dos robôs em ambientes compartilhados com seres humanos, como hospitais, aeroportos, centros comerciais e residências. Para além de executar uma tarefa com eficiência, espera-se que esses robôs convivam harmoniosamente com as pessoas, respeitando normas sociais implícitas – manter distâncias interpessoais adequadas, evitar movimentos bruscos, sinalizar intenções – de modo a não causar desconforto ou insegurança. Diversos autores têm enfatizado essas exigências em revisões recentes sobre navegação social (Ferrer et al. (2013); Singamaneni et al. (2024); Sisbot et al. (2007)). Esse campo de estudo, conhecido como *navegação social de robôs*, envolve desafios multidisciplinares relacionados à percepção, à previsão do movimento humano, ao planejamento de trajetórias e à interação social.

Apesar dos avanços, a avaliação e o desenvolvimento de algoritmos de navegação social ainda enfrentam limitações práticas e metodológicas. A literatura tem apontado a ausência de conjuntos de dados abrangentes que capturem a gama completa de comportamentos de pedestres na presença de robôs – desde evitar, ignorar até se aproximar – o que dificulta a aprendizagem de modelos de interação realistas (Agrawal et al. (2026)). Muitos datasets existentes não anotam explicitamente as reações dos pedestres nem oferecem versões contrastantes com robô ausente, estacionário ou em movimento. Além disso, boa parte dos modelos de comportamento humano, como o *Modelo de Forças Sociais* (SFM), é empregada com parâmetros genéricos ou ajustados subjetivamente, comprometendo a reprodutibilidade e a precisão dos experimentos.

Outro desafio relevante é o chamado *efeito de novidade*: pessoas tendem a reagir de maneira anormalmente curiosa ou defensiva quando interagem com um robô pela primeira vez. Pesquisas em interação humano-robô mostram que esse efeito não deve ser tratado apenas como "ruído" a ser eliminado, pois ele pode influenciar de forma significativa as trajetórias e avaliações de conforto. Para reduzir tal viés, é importante expor os participantes ao robô antes da coleta ou distribuir as coletas em grupos distintos, garantindo que as observações reflitam comportamentos estáveis.

Para contextualizar este trabalho, distinguimos dois padrões de navegação robótica: **navegação conformista**, na qual o robô respeita normas sociais implícitas (mantém distância

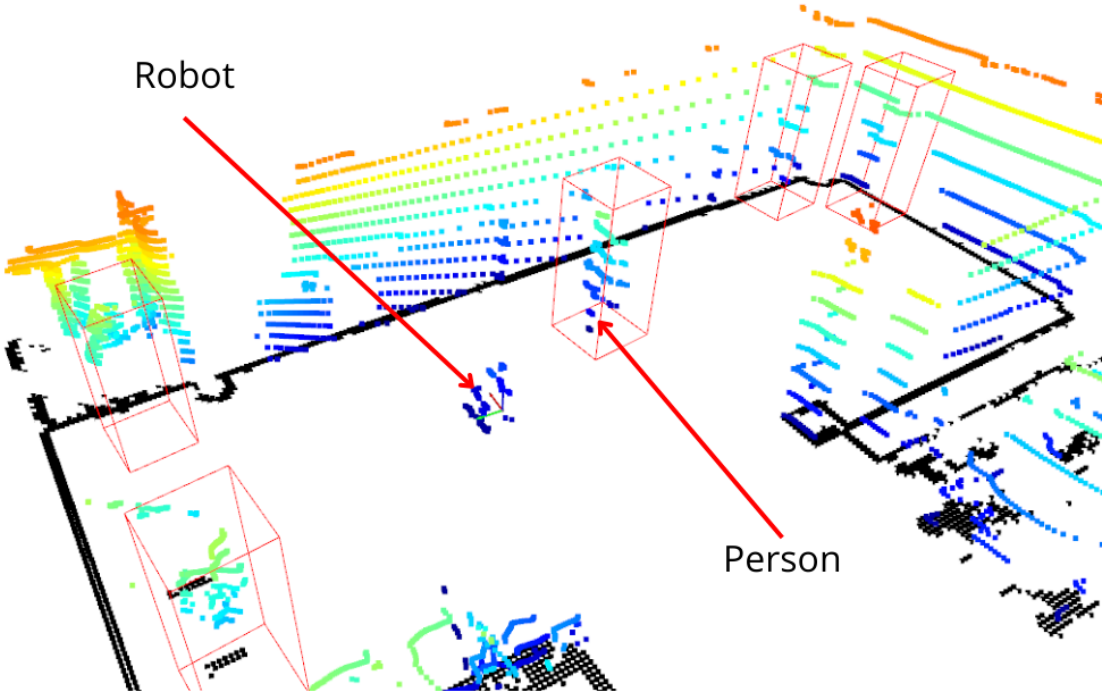

Figura 1.1: Imagem de um dos tipos de dados disponivel no NavWareSet (Nuvem de pontos anotada)

interpessoal, desvia suavemente e cede passagem), e **navegação não conformista**, onde o robô segue diretamente sua rota ignorando a presença e o conforto das pessoas. Esses comportamentos contrastantes servem para evidenciar a influência do respeito às normas sociais nas respostas humanas.

Tendo em vista essas lacunas, esta dissertação propõe a construção de um novo conjunto de dados, denominado **NavWareSet**, que registra interações sociais e não sociais entre humanos e dois robôs (Toyota HSR e Clearpath Jackal) em um ambiente interno padronizado. O NavWareSet foi concebido como resposta à falta de dados contrastivos e à necessidade de bases reprodutíveis. A metodologia inclui a coleta, anotação manual e organização de dados multimodais em cenários padronizados de navegação social, bem como a implementação do modelo de forças sociais no simulador **UAIbot**. O SFM é utilizado aqui apenas como estudo de caso para ilustrar o uso do dataset na calibração de modelos de comportamento humano.

1.1 Objetivos Gerais e Específicos

O objetivo geral desta dissertação é contribuir para o avanço da navegação social de robôs por meio da construção de um conjunto de dados realista e da calibração de modelos de comportamento humano.

Os objetivos específicos incluem:

- Desenvolver um ambiente experimental controlado capaz de reproduzir diferentes cenários

de navegação social;
- Coletar dados multimodais (vídeo, LiDAR, imagens RGB-D, odometria) em interações reais entre humanos e robôs, nas versões *conformista* e *não conformista* de cada cenário;
- Anotar manualmente as posições dos participantes humanos ao longo do tempo e documentar as decisões do teleoperador, garantindo precisão e transparência;
- Organizar os dados coletados em um formato reutilizável, com documentação e metadados claros que facilitem a reprodutibilidade;
- Implementar o modelo de forças sociais no simulador UAIbot e demonstrar como a base de dados pode ser utilizada para calibrar modelos de comportamento humano;
- Ajustar os parâmetros do modelo de forças sociais com base em trajetórias reais obtidas no experimento, tratando essa otimização como um estudo de caso, e não como foco principal do trabalho.

1.2 Contribuições e Originalidade

As principais contribuições deste trabalho podem ser destacadas da seguinte forma:

- **Construção do NavWareSet**: criação de um conjunto de dados multimodal e anotado que contempla sete cenários distintos de navegação social em ambiente interno, com versões *conformista* e *não conformista* para quase todos os cenários e utilização de dois robôs distintos. Todos os arquivos estão organizados com metadados detalhados, mapas de ocupação e poses sincronizadas;
- **Metodologia de coleta e anotação transparente**: descrição minuciosa do ambiente, dos robôs, dos sensores utilizados e do processo de teleoperação. Ao relatar como o operador decide o lado e o momento do desvio em cada condição, o trabalho facilita a compreensão das decisões que geraram as trajetórias e torna a base de dados auditável;
- **Discussão crítica sobre o conjunto de dados**: análise dos efeitos de aprendizagem e variabilidade de trajetórias, comparação da duração e diversidade das trilhas com conjuntos existentes e reflexão sobre possíveis vieses introduzidos por um ambiente único e por um número limitado de participantes;
- **Implementação do Modelo de Forças Sociais no UAIbot**: desenvolvimento e disponibilização de uma extensão do simulador para suportar pedestres e obstáculos baseados em forças sociais, permitindo simulações interativas em 2D e 3D;
- **Estudo de caso de calibração de parâmetros**: aplicação de um processo objetivo de otimização para ajustar os parâmetros do SFM utilizando as trajetórias do NavWareSet, demonstrando a utilidade do dataset para validação de modelos e comparação com parâmetros da literatura.

A originalidade do trabalho reside na integração entre coleta realista de dados (comportamentos conformistas e não conformistas), anotação manual precisa, simulação física baseada em forças sociais e identificação quantitativa de parâmetros. Essa combinação fecha o ciclo

entre observação, modelagem e validação, fornecendo uma base sólida para o desenvolvimento e avaliação de algoritmos de navegação social e estimulando a criação de novos modelos que considerem explicitamente a presença do robô.

Este trabalho foi desenvolvido no contexto do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais (UFMG), com apoio de infraestrutura da Université de Technologie de Belfort-Montbéliard (França), onde os experimentos foram realizados. Os resultados obtidos deram origem a um artigo aceito na conferência internacional Robotics in Education 2025 (Brayan et al. (2025b)), além da preparação de um artigo intitulado *NavWareSet: A Dataset of Socially Compliant and Non-Compliant Robot Navigation*, que será submetido ao periódico *The International Journal of Robotics Research* (IJRR).

1.3 Organização do Trabalho

Este documento está organizado da seguinte forma:

- O **Capítulo 2** apresenta a fundamentação teórica e os principais trabalhos relacionados ao tema da navegação social, modelagem de comportamento humano e construção de datasets.
- O **Capítulo 3** descreve em detalhes a metodologia adotada para a construção do NavWareSet, incluindo o ambiente experimental, os equipamentos utilizados, os cenários implementados e a pipeline de processamento de dados. Também são apresentados a implementação do modelo de forças sociais no UAIbot e o processo de otimização dos parâmetros.
- O **Capítulo 4** apresenta os principais resultados obtidos, incluindo a estrutura final do dataset, os testes de simulação realizados com o modelo SFM no UAIbot, os parâmetros identificados e a comparação com a literatura.
- O **Capítulo 5** conclui o trabalho, discutindo as principais contribuições, limitações e possíveis extensões futuras.

Capítulo 2

Revisão Bibliográfica

2.1 O Campo da Navegação Social

A navegação social é uma subárea emergente da robótica, essencial para o desenvolvimento de robôs capazes de operar em ambientes compartilhados com humanos de maneira eficiente e harmoniosa. Esse campo busca compreender e modelar interações humano-robô em cenários dinâmicos, promovendo comportamentos socialmente aceitáveis, seguros e eficazes (Francis et al. (2023)).

Nos últimos anos, diversas revisões sistemáticas foram publicadas, consolidando os desafios e avanços na área de navegação social de robôs. Dentre elas, destacam-se Karwowski et al. (2024); Mavrogiannis et al. (2021); Mirsky et al. (2022); Singamaneni et al. (2024), que propõem taxonomias abrangentes, discutem critérios de avaliação e apontam lacunas metodológicas persistentes. Essas revisões ressaltam a importância de modelos interpretáveis, dados realistas e métricas padronizadas como pilares para o avanço sustentável da área.

Um aspecto central da navegação social é o respeito às normas sociais e às preferências humanas. Estudos destacam a necessidade de robôs que possam navegar sem causar desconforto aos humanos, evitando colisões, respeitando espaços pessoais e exibindo comportamentos previsíveis e legíveis. Essas características são particularmente importantes em ambientes como hospitais, shoppings e residências, onde interações sociais são frequentes e variadas.

Diversos modelos e algoritmos têm sido propostos para lidar com os desafios da navegação social. O Modelo de Forças Sociais (*Social Forces Model* - SFM), por exemplo, utiliza forças abstratas para simular interações sociais e físicas entre robôs, humanos e obstáculos (Helbing & Molnár (1995)). Esse modelo tem sido amplamente utilizado devido à sua simplicidade e eficácia em reproduzir padrões de movimento humano (Zanlungo et al. (2011)). Alternativas mais recentes incluem abordagens baseadas em aprendizado de máquina, como redes neurais convolucionais e modelos generativos, que aprendem diretamente a partir de dados empíricos (Karnan et al. (2022)).

Apesar dos avanços, ainda existem lacunas importantes. Por exemplo, muitos dos al-

goritmos atuais não conseguem generalizar para cenários novos ou desconhecidos (Robicquet et al. (2016)). Além disso, há uma falta de consenso sobre métricas objetivas para avaliar comportamentos sociais em navegação. Trabalhos como Rudenko et al. (2020) propõem datasets controlados, mas sua aplicabilidade é limitada pela diversidade de trajetórias e interações disponíveis.

Outro desafio é a integração de dados multimodais para melhorar a percepção e a tomada de decisão do robô. Estudos mostram que sensores como LiDAR e câmeras RGB-D podem complementar-se, proporcionando uma visão mais abrangente do ambiente (Yan et al. (2018)). No entanto, essa integração exige modelos robustos e eficientes, capazes de processar grandes volumes de dados em tempo real.

Ao longo dos anos, o campo da navegação social tem avançado significativamente, mas desafios persistem, especialmente no que diz respeito à modelagem de comportamentos sociais complexos, à avaliação objetiva de algoritmos e à generalização para cenários do mundo real. Este trabalho busca contribuir para essa área ao propor um conjunto integrado de soluções, incluindo novos datasets, ferramentas de simulação e parâmetros.

2.2 Datasets de Navegação Social

Datasets desempenham um papel essencial na pesquisa de navegação social (Figura 2.1), fornecendo a base para o treinamento e a avaliação de algoritmos que permitem aos robôs operar de forma eficiente e socialmente consciente em ambientes humanos. Diversos datasets foram desenvolvidos ao longo dos anos, cada um com características distintas, voltados para diferentes aspectos das interações humano-robô. Os datasets *SCAND* e *THÖR* são dois exemplos amplamente utilizados. O *SCAND* (*Socially Compliant Navigation Dataset*) oferece 8,7 horas de dados multimodais, coletados por robôs teleoperados em ambientes naturais (Karnan et al. (2022)). Este dataset destaca-se por capturar comportamentos sociais em cenários reais, mas sua falta de controle experimental limita sua aplicação para benchmarking rigoroso. Por outro lado, o *THÖR* foca em interações sociais precisas em ambientes controlados, com anotações de alta frequência (Rudenko et al. (2020)). Embora sua configuração controlada ofereça dados consistentes, o *THÖR* não explora uma grande diversidade de trajetórias e comportamentos, restringindo sua aplicabilidade em cenários mais gerais.

Outros datasets, como *ETH* e *UCY*, capturam trajetórias de pedestres em espaços públicos usando câmeras estáticas (Pellegrini et al. (2009), Lerner et al. (2007)). Embora úteis para o estudo do movimento humano, esses datasets carecem de interações robóticas, dificultando sua aplicação direta à navegação social de robôs. O *Stanford Drone Dataset* adiciona a perspectiva aérea, oferecendo dados em grande escala de interações humanas (Robicquet et al. (2016)), mas enfrenta desafios semelhantes no que diz respeito à ausência de robôs.

Datasets específicos para robôs, como *L-CAS* e *JRDB*, oferecem dados de longo prazo e visão centrada no robô, respectivamente. O *L-CAS* foca em percepções robóticas em ambientes

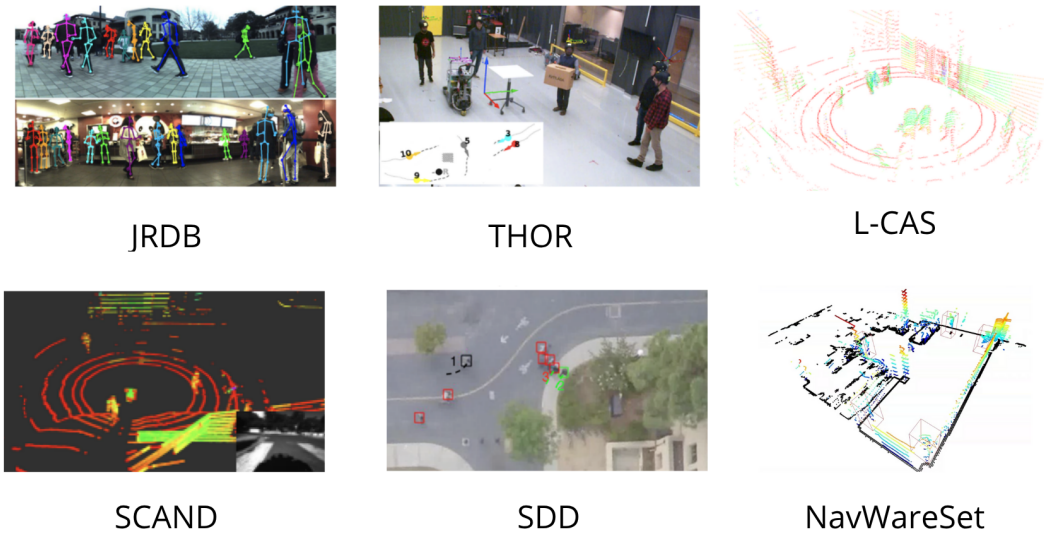

Figura 2.1: Datasets de Navegação Social

públicos (Yan et al. (2018)), enquanto o *JRDB* captura interações humano-robô em ambientes internos e externos (Martin-Martin et al. (2019)). Ambos contribuem significativamente para a pesquisa de navegação, mas apresentam limitações na modelagem de comportamentos sociais complexos.

O *NavWareSet*, desenvolvido no contexto deste trabalho, busca preencher as lacunas existentes nos datasets atuais (Brayan et al. (2025a)). Ele captura comportamentos sociais e não sociais em sete cenários cuidadosamente planejados, como aproximações frontais, tráfego perpendicular e entregas de objetos. Utilizando dois robôs distintos, o Toyota Human Support Robot (HSR) e o Clearpath Jackal, o *NavWareSet* oferece uma diversidade de dados multimodais, incluindo LiDAR, câmeras RGB-D e odometria. Além disso, combina interações sociais controladas e cenários variáveis, tornando-o uma ferramenta robusta para treinamento e avaliação de algoritmos de navegação social.

A Tabela 2.1 resume as principais características de alguns dos datasets mencionados, destacando suas aplicações e limitações.

2.2.1 Análise crítica dos datasets

A Tabela 2.1 resume características gerais de alguns bancos de dados utilizados para navegação social. A simples contagem de horas e trajetórias, no entanto, não evidencia as sutis diferenças de foco e propósito entre eles. O *SCAND*, por exemplo, registra aproximadamente 8,7 horas (cerca de 25 milhas) de dados coletados por robôs teleoperados em locais públicos ao longo de quinze dias, totalizando 138 trajetórias (Karnan et al. (2022)). Essa riqueza temporal é valiosa para treinar métodos de aprendizado profundo, mas a coleta é totalmente observacional:

Tabela 2.1: Comparação de Datasets de Navegação

Dataset	**Env.**	**Robot**	**Dur.**	**#Traj.**	**On-Robot Sensors**	**Off-Robot Sensors**	**Behavior**
SCAND	In/Out	Spot, Jackal	8.7 h	138	LiDAR, RGB	None	Compliant
THÖR	In	Mock-up	1 h	600+	RGB-D, LiDAR	Mocap, Eye-tracker	Compliant
ETH/UCY	Out	None	N/A	∼750	None	Static cameras	-
L-CAS	In/Out	Pioneer 3-AT	49 min	N/A	LiDAR	None	-
JRDB	In/Out	JackRabbot	64 min	2000+	LiDAR, RGB-D	None	Compliant
SDD	Out	None	1 h	20,000+	None	Drone RGB	-
NavWareSet	In	HSR, Jackal	3.2 h	1000+	LiDAR, RGB-D	LiDAR, RGB	Compliant & Non-Compliant

não há cenários contrabalanceados nem condições contrastantes que permitam isolar o efeito da presença do robô. Além disso, os robôs (Clearpath Jackal e Boston Dynamics Spot) interagem livremente com as pessoas sem um roteiro pré-estabelecido, de modo que as situações de aproximação frontal, ultrapassagem e tráfego perpendicular ocorrem de forma espontânea e imprevisível.

O *THÖR* adota uma estratégia complementar: em vez de registrar trajetórias espontâneas em ambientes naturais, ele é coletado em laboratório e foca na aquisição precisa de dados de navegação humano-robô. O dataset contém trajetórias de movimento humano e dados de orientação da cabeça e do olhar coletados em um ambiente interno com mapeamento de obstáculos e metas conhecidas. A posição, a orientação da cabeça, a direção do olhar e o agrupamento social dos participantes são anotados com alta frequência utilizando sistemas de captura de movimento e um sensor LiDAR 3D montado no robô. Segundo Rudenko et al. (2020), essas anotações ricas permitem analisar métricas como duração média de rastreamento, ruído do ground truth, curvatura e variação de velocidade das trajetórias. Embora ofereça maior variedade de comportamentos e qualidade de anotações que datasets anteriores, o THÖR carece de condições contrastivas (com e sem robô) e de variações sistemáticas de densidade de pedestres, o que limita sua utilidade para estudar respostas sociais específicas à presença do robô.

Já os datasets puramente pedestres, como *ETH/UCY* e o *Stanford Drone Dataset*, fornecem milhares de trajetórias em praças, universidades e estações ferroviárias. São úteis para modelar a dinâmica da multidão e para treinar preditores de caminhos, mas não incluem robôs, não diferenciam comportamentos conformistas e não conformistas e utilizam apenas câmeras

fixas como sensor. Por sua vez, *JRDB* e *L-CAS* oferecem uma perspectiva embarcada (sensorial) de robôs que se movem em ambientes urbanos ou internos: o *JRDB* reúne 64 minutos de dados anotados multimodais gravados a partir do robô JackRabbot, com 2,3 milhões de *bounding boxes* e 1,8 milhões de cubóides distribuídos em 3500 trajetórias (Martin-Martin et al. (2019)); o *L-CAS* foi capturado com um LiDAR VLP-16 em um edifício público e contém 935 trajetórias de pedestres extraídas em 19 minutos de gravação (Yan et al. (2018)). Embora esses datasets sejam valiosos para percepção e rastreamento, eles não são desenhados para estudar navegação social propriamente dita: não há teleoperação de robôs em diferentes modos e as métricas se concentram em detecção e segmentação.

Em contraste com esses trabalhos, o *NavWareSet* foi desenhado para estudar especificamente a influência do comportamento do robô sobre as respostas humanas. Ele combina duas plataformas robóticas distintas, sete cenários sistematicamente definidos e duas versões de navegação (*conformista* e *não conformista*). O teleoperador segue um roteiro prescrito, repetindo cada cenário múltiplas vezes para garantir comparabilidade estatística. Ao incluir versões com e sem respeito às normas sociais, controlando a direção do desvio e a proximidade interpessoal, o *NavWareSet* permite análises diferenciais que nenhum dataset anterior disponibiliza. Ademais, a base conta com anotações manuais precisas das posições humanas e metadados do teleoperador, possibilitando calibração objetiva de modelos como o *Social Force Model* e estudos sobre *novelty effect*. Por fim, a coleta em um único ambiente interno elimina variáveis externas (clima, iluminação), tornando os dados mais consistentes. Apesar de a duração total do NavWareSet (aprox. 3,2 horas) ser menor que a de SCAND, a diversidade de cenários, a estrutura contrastiva e a documentação detalhada tornam-no uma contribuição única e complementar à literatura.

Além dos datasets centrados na interação humano-robô, há iniciativas que analisam fluxos de pedestres e a atenção conjunta em contextos de trânsito humano ou de veículos. Embora esses estudos não incluam robôs móveis, eles oferecem visões complementares da dinâmica social urbana e inspiram métricas e metodologias que podem ser aproveitadas pela navegação social robótica. Este trabalho se apoia nessa literatura mais ampla, mas foca especificamente em interações humano-robô controladas, fornecendo dados contrastivos para investigar a influência do comportamento do robô sobre os pedestres.

2.3 Social Forces Model

O *Social Force Model* (SFM) é um framework conceitual para representar o comportamento de pedestres e outros agentes móveis como resultado de forças de *atração* (que direcionam o agente ao seu objetivo) e de *repulsão* (que evitam colisões com pessoas, robôs ou obstáculos). Cada agente tenta seguir uma velocidade e direção desejadas, mas ajusta sua trajetória de forma contínua ao sentir a presença dos demais. Diversas variações do SFM foram propostas desde a formulação original de Helbing e Molnár (Helbing & Molnár (1995)),

diferindo na forma das funções de força (exponencial, linear, anisotrópica), na escolha de parâmetros e na inclusão de termos adicionais. Essa diversidade fez com que os parâmetros $(A, B, \lambda, \tau, v_{\text{des}})$ reportados na literatura apresentem ordens de magnitude distintas (Johansson et al. (2008); Kretz et al. (2017)). No contexto deste trabalho, o SFM é utilizado como estudo de caso e não como modelo definitivo. Detalhes matemáticos e a calibração paramétrica são apresentados no Capítulo 3. Abaixo, mantemos a descrição completa do SFM apenas para referência.

A posição de um pedestre α no tempo t, denotada por $r_\alpha(t)$, evolui de acordo com sua velocidade $v_\alpha(t)$:

$$\frac{dr_\alpha(t)}{dt} = v_\alpha(t) \tag{1}$$

A aceleração, representada pela taxa de mudança de velocidade, é influenciada pela força social líquida $f_\alpha(t)$ e por flutuações aleatórias $\xi_\alpha(t)$:

$$\frac{dv_\alpha(t)}{dt} = f_\alpha(t) + \xi_\alpha(t) \tag{2}$$

A força social $f_\alpha(t)$ que age sobre o pedestre α é composta por três componentes principais: a força atrativa $f_{\alpha g}(t)$, que o direciona ao objetivo, as forças repulsivas $f_{\alpha\beta}(t)$ provenientes de outros pedestres e as forças repulsivas $f_{\alpha i}(t)$ decorrentes de obstáculos. Esta combinação é expressa como:

$$f_\alpha(t) = f_{\alpha g}(t) + \sum_{\beta \neq \alpha} f_{\alpha\beta}(t) + \sum_i f_{\alpha i}(t) \tag{3}$$

A força atrativa em direção ao objetivo, $f_{\alpha g}(t)$, é proporcional à diferença entre a velocidade atual v_α e a velocidade desejada $v^0_\alpha e^0_\alpha$. Essa força é dada por:

$$f_{\alpha g}(t) = m_\alpha \frac{1}{\tau_\alpha} \left(v^0_\alpha e^0_\alpha - v_\alpha \right) \tag{2.1}$$

Nesta equação, v^0_α é a velocidade desejada (indicando quão rápido o pedestre deseja se mover), e^0_α é a direção desejada (um vetor unitário apontando para o objetivo), m_α representa a massa do pedestre, e τ_α é o tempo de relaxamento, que controla a rapidez com que o pedestre ajusta sua velocidade. Em muitas simulações, a massa m_α é normalizada para 1, simplificando os cálculos.

As forças repulsivas entre o pedestre α e outras entidades q (sejam pedestres β ou obstáculos i) são moduladas por uma função de ponderação anisotrópica $w(\phi_{\alpha q})$, que considera o ângulo de interação:

$$f_{\alpha q}(t) = w(\phi_{\alpha q}) f(d_{\alpha q}(t)) \tag{4}$$

Aqui, $f(d_{\alpha q})$ é a força repulsiva dependente da distância, enquanto $w(\phi_{\alpha q})$ ajusta a

intensidade da força com base na orientação angular da interação, e $d_{\alpha q}$ é o vetor de distância apontando da entidade q para o pedestre α. A força repulsiva decai exponencialmente com a distância:

$$f(d_{\alpha q}) = A_\alpha e^{-d_{\alpha q}/B_\alpha} \frac{d_{\alpha q}}{|d_{\alpha q}|} \tag{5}$$

Onde A_α é a intensidade da interação, B_α é o alcance da interação. É importante lembrar que a força repulsiva só é calculada dessa forma na chamada **especificação circular** do modelo de força social (Helbing & Johansson (2013)) que difere de outras especificações na literatura, incluindo a especificação original apresentada por Helbing & Molnár (1995).

A função de ponderação anisotrópica $w(\phi_{\alpha q})$ introduz sensibilidade direcional às interações, considerando o ângulo $\phi_{\alpha q}$. O ângulo é calculado como:

$$\cos(\phi_{\alpha q}) = \frac{v_\alpha}{|v_\alpha|} \cdot \frac{-d_{\alpha q}}{|d_{\alpha q}|} \tag{10}$$

A função de ponderação é definida por:

$$w(\phi_{\alpha q}) = \lambda_\alpha + (1-\lambda_\alpha)\frac{1+\cos(\phi_{\alpha q})}{2} \tag{11}$$

O parâmetro λ_α controla a sensibilidade a interações provenientes de diferentes direções, especialmente as de trás. Esse modelo detalhado permite simular interações sociais complexas e reproduzir padrões de comportamento humano de maneira realista.

2.3.1 Critérios para a escolha do SFM

O *Social Forces Model* combina forças atrativas, repulsivas e dependências angulares, fornecendo uma abordagem extensível e eficaz para a análise e simulação de navegação social. Sua flexibilidade permite aplicações em uma ampla gama de cenários, tornando-o uma ferramenta fundamental para estudos na área. Apesar de o SFM apresentar algumas limitações conhecidas, como a suposição de forças simétricas entre os agentes e a ausência de mecanismos explícitos de antecipação, ele ainda é amplamente utilizado como referência em estudos de navegação social. Nesta seção, discutimos como o modelo se compara com outras abordagens e justificamos sua escolha neste trabalho.

De forma geral, modelos microscópicos de navegação humana podem ser divididos em duas categorias principais: modelos baseados em velocidade (como RVO/ORCA) e modelos baseados em forças (como o SFM e o HSFM). Um estudo recente comparou o desempenho de políticas de navegação treinadas com diferentes modelos (ORCA, SFM e HSFM), considerando métricas de eficiência e de comportamento social. Os resultados indicam que o HSFM tende a gerar trajetórias mais curtas e rápidas, enquanto o SFM produz comportamentos mais suaves, com menores acelerações e maior respeito ao espaço interpessoal, principalmente em cenários

mais densos. Já o ORCA, embora eficiente em termos de desvio de colisões, pode resultar em movimentos mais bruscos. Outro ponto relevante observado nesse trabalho é que políticas treinadas com modelos de forças costumam generalizar melhor para outros cenários, mantendo um comportamento mais estável quando avaliadas em modelos diferentes. Isso sugere que, mesmo não sendo o mais eficiente em termos de trajetória, o SFM pode produzir comportamentos mais consistentes do ponto de vista social.

Com o avanço recente das redes neurais, surgiram diversos modelos de previsão de trajetória baseados em aprendizado, como Social-LSTM e Social-GAN. Esses métodos normalmente apresentam melhor desempenho em métricas quantitativas, como erro médio de previsão. No entanto, eles dependem de grandes quantidades de dados rotulados e, em geral, são menos interpretáveis. Uma revisão da literatura aponta que, apesar da alta precisão desses modelos, ainda há dúvidas sobre sua capacidade de reproduzir dinâmicas coletivas de forma consistente em simulações maiores (Korbmacher & Tordeux (2022)). Além disso, a falta de interpretabilidade pode dificultar a análise de falhas. Nesse contexto, o SFM continua sendo uma alternativa interessante por ser mais simples, interpretável e não depender de dados rotulados, o que facilita sua aplicação em cenários com menor disponibilidade de dados.

Outro aspecto importante diz respeito à robustez dos modelos. Trabalhos recentes mostram que mesmo modelos considerados "socialmente conscientes" podem falhar em situações adversariais. Em particular, pequenas perturbações nas trajetórias previstas podem levar a colisões, mesmo em modelos mais modernos (Saadatnejad et al. (2022)). Isso indica que um bom desempenho em métricas tradicionais não garante necessariamente um comportamento seguro ou socialmente adequado. Dessa forma, modelos mais simples e interpretáveis, como o SFM, podem ser úteis para entender melhor essas limitações.

Em resumo, a escolha do SFM neste trabalho se deve a alguns fatores principais: (i) modelos baseados em forças tendem a apresentar comportamentos mais compatíveis com interações sociais, mesmo que não sejam os mais eficientes; (ii) modelos baseados em aprendizado, apesar de mais precisos, exigem grandes volumes de dados e são menos interpretáveis; (iii) o SFM é relativamente simples e pode ser calibrado mesmo com poucos dados. Dessa forma, o SFM foi adotado como modelo inicial para análise do NavWareSet, servindo como base para comparações futuras com outras abordagens.

Os valores dos parâmetros utilizados no *Social Forces Model* (SFM) variam significativamente entre diferentes estudos, como mostrado na Tabela 2.2. Por exemplo, o parâmetro A, que representa a intensidade da interação, varia de 0.4 (Johansson et al. (2008)) a 100.0 (Luber et al. (2010)), enquanto o alcance da interação, B, apresenta variações entre 0.4 (Luber et al. (2010)) e 1.7 (Johansson et al. (2008)). Da mesma forma, o tempo de relaxamento τ, que controla a rapidez com que os pedestres ajustam suas velocidades, também varia amplamente de 0.2 (Helbing et al. (2005)) a 1.0 (Agrawal et al. (2024)). Essas discrepâncias refletem as diferenças nos contextos e metodologias de cada estudo, dificultando a generalização do modelo para novos cenários. Este trabalho contribui para reduzir essas incertezas ao parametrizar

Tabela 2.2: Parâmetros utilizados em diferentes estudos do *Social Forces Model*. P-P significa *Person to Person*. P-R significa *Person to Robot*. P-O significa *Person to Object*.

Relação	**A**	**B**	**d**	λ	τ	$v_{\mathbf{des}}$
P-R Agrawal et al. (2024)	7.9	1.0	N/A	0.4	0.6	N/A
P-R Ferrer et al. (2013)	2.7	0.8	0.4	0.6	0.4	N/A
P-P Agrawal et al. (2024)	2.0	0.9	N/A	0.4	0.6	N/A
P-P Zanlungo et al. (2011)	1.1	0.7	N/A	0.3	0.7	N/A
P-P Johansson et al. (2008)	0.4	1.7	N/A	0.1	N/A	Variável
P-P Luber et al. (2010)	70.0	0.4	0.4	0.5	0.5	N/A
P-P Kretz et al. (2017)	5.0	0.5	0.9	0.1	0.2	1.3
P-P Helbing et al. (2005)	3.0	0.2	0.6	0.8	1.0	1.3
P-O Helbing et al. (2005)	5.0	0.1	N/A	0.8	1.0	1.3
P-O Luber et al. (2010)	100.0	0.0	0.4	0.5	0.5	N/A

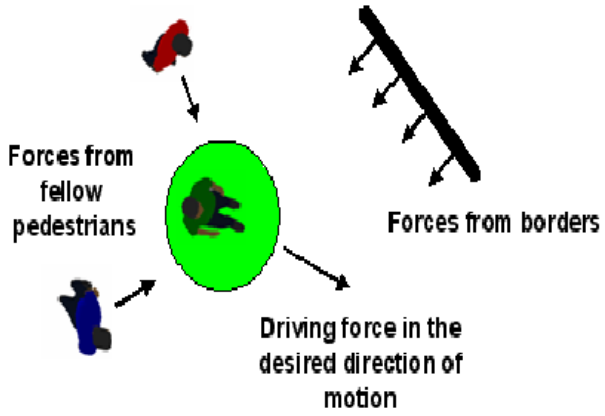

Figura 2.2: Atributos do modelo de Forças Sociais. Fonte: Laufer (2008)

o SFM com base em dados empíricos coletados no *NavWareSet*, oferecendo valores mais consistentes e adequados para cenários de navegação social específicos, permitindo maior precisão e realismo em simulações.

2.4 O Simulador UAIbot

O **UAIbot** (Brayan et al. (2025b)) é um simulador de robótica em código aberto desenvolvido com o objetivo de proporcionar uma plataforma acessível e interativa para o ensino, pesquisa e prototipagem rápida em robótica. Sua principal característica é a interface baseada em *browser*, o que o torna multiplataforma e fácil de utilizar, mesmo em dispositivos com recursos computacionais limitados. Na Figura 2.3 pode-se ver exemplos de simulações feitas com o UAIbot.

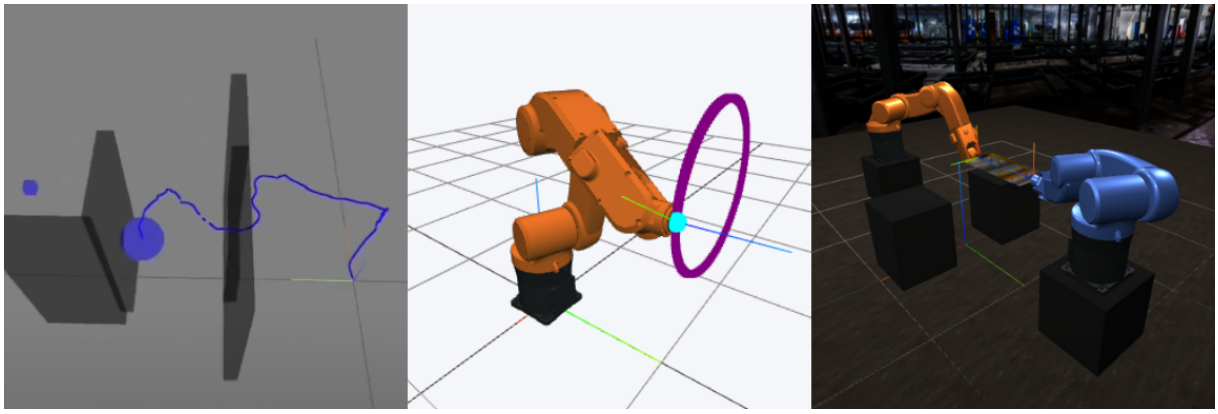

Figura 2.3: Simulações feitas com o UAIbot.

2.4.1 Objetivos e Motivação

O desenvolvimento do UAIbot foi motivado pela necessidade de uma ferramenta educacional simples, moderna e visualmente intuitiva, que permitisse a estudantes e pesquisadores simular e visualizar conceitos fundamentais de robótica, como cinemática direta, trajetórias, controle e ambientes interativos.

Ao contrário de simuladores tradicionais como Gazebo, Webots ou CoppeliaSim, que exigem configurações mais complexas e conhecimento prévio sobre ROS ou middlewares robóticos, o UAIbot foca na simplicidade e na curva de aprendizado reduzida, sem abrir mão de capacidade gráfica e flexibilidade.

2.4.2 Arquitetura

O núcleo do UAIbotPy (versão em Python do UAIbot) tem suporte a visualização 3D baseada em WebGL por meio de integração com `Three.js`. O simulador permite a criação e manipulação de objetos geométricos (como cilindros, esferas, caixas), robôs com múltiplos elos, articulações, sensores e obstáculos, todos controláveis por código. Ele é compatível com animações interativas e exportação de simulações como páginas web estáticas.

2.4.3 Aplicações

Desde sua publicação, o UAIbot tem sido utilizado em diferentes contextos:

- Ensino de robótica em cursos de graduação e pós-graduação;
- Demonstrações interativas de algoritmos de cinemática, planejamento e controle;
- Simulação de navegação social e interações humano-robô, como apresentado neste trabalho;
- Geração de animações para artigos, apresentações e materiais didáticos.

Seu código-fonte e exemplos estão disponíveis publicamente[1], incentivando a reutilização

[1] `https://uaibot.github.io/`

e personalização.

2.4.4 Relevância para Este Trabalho

Neste trabalho, o UAIbot foi estendido com uma implementação do *Social Force Model* (SFM) para simulação de pedestres, obstáculos e ambientes interativos. Essa extensão permitiu:

- Simular trajetórias humanas em resposta à presença de robôs móveis;
- Realizar experimentos controlados de calibração de parâmetros do SFM;
- Visualizar interações complexas entre múltiplos agentes em tempo real.

Sua natureza leve e programável tornou o UAIbot uma ferramenta ideal para conduzir as simulações necessárias na identificação dos parâmetros do modelo de forças sociais com base nos dados reais do *NavWareSet*.

Capítulo 3

Metodologia

Este capítulo descreve, de forma detalhada, os procedimentos metodológicos adotados ao longo da pesquisa. Inicialmente, é apresentada a configuração dos experimentos utilizados para a coleta dos dados que compõem o conjunto *NavWareSet*, incluindo a descrição do ambiente, dos sensores e dos robôs envolvidos. Em seguida, são descritos os diferentes cenários de navegação social implementados, bem como o planejamento e a execução das gravações com participantes humanos. O capítulo também aborda a pipeline de processamento dos dados, desde a coleta bruta até a geração dos produtos finais organizados e anotados. Por fim, detalha-se a implementação do modelo de forças sociais no simulador UAIbot, bem como o processo de otimização dos seus parâmetros com base em trajetórias reais, com o objetivo de ajustar o modelo ao comportamento observado durante os experimentos.

3.1 Configuração dos experimentos

Esta seção apresenta a configuração dos experimentos realizados para a construção do conjunto de dados *NavWareSet*. São descritos o ambiente físico utilizado nas coletas, a composição e o posicionamento da estação de registro de ground-truth (GRS), e os dois robôs empregados nos cenários de interação humano-robô. As escolhas de hardware, posicionamento dos sensores e estratégias de navegação foram cuidadosamente definidas para permitir a captura de comportamentos sociais realistas, tanto em situações de navegação cooperativa quanto em situações de navegação não social. As subseções a seguir detalham os componentes do ambiente e dos experimentos, estabelecendo a base necessária para o processamento e análise dos dados coletados.

3.1.1 Ambiente de Coleta de Dados

Os dados foram coletados em um espaço amplo, com dimensões aproximadas de 3,9 m × 10,7 m. O ambiente também é utilizado como garagem para os veículos experimentais da Université de Technologie de Belfort-Montbéliard. O ambiente possui uma abertura lateral que

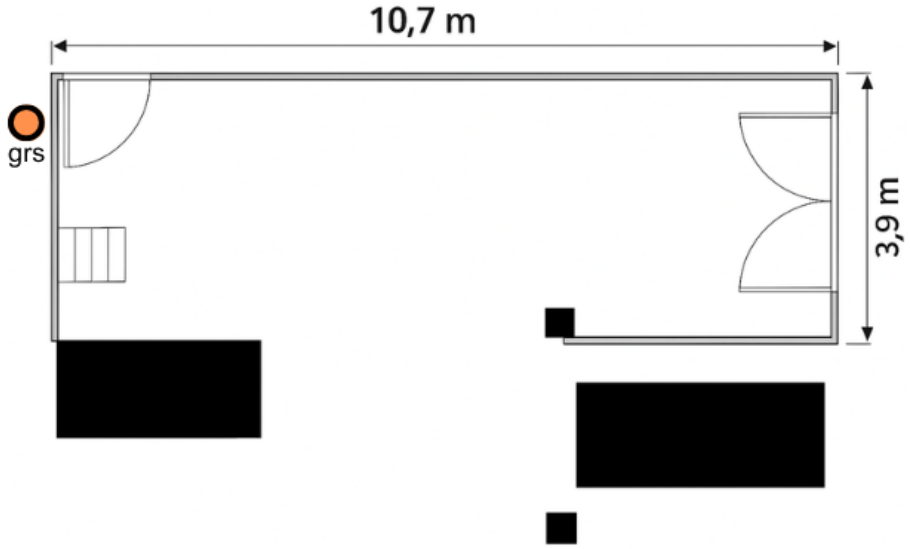

Figura 3.1: Mapa do local de captura de dados

permite a realização de trajetórias tanto em linha reta quanto contornando obstáculos, o que o torna apropriado para representar diferentes situações de navegação social.

A área foi equipada com uma Estação de Registro de Ground-Truth (GRS, do inglês *Ground-Truth Recording Station*), composta por uma câmera de vídeo e um sensor LiDAR 3D, responsáveis por capturar com precisão os movimentos dos pedestres durante os experimentos.

Para cada cena registrada, foram definidos pontos de destino distribuídos pelo ambiente de modo a simular diferentes cenários de navegação social, como cruzamentos, interações frontais e desvios laterais. Esses alvos foram posicionados estrategicamente no mapa para estimular comportamentos naturais de locomoção e interação entre os participantes.

A Figura 3.1 apresenta a planta do ambiente de coleta com os principais elementos destacados. Ao todo, participaram 17 voluntários adultos nos experimentos. Todos foram previamente informados sobre os objetivos da pesquisa e assinaram um termo de consentimento. As sessões de coleta foram realizadas em ambiente interno, sob supervisão, garantindo o bem-estar e a segurança dos participantes.

3.1.2 *Ground-Truth Recording Station*

A Estação de Registro de Ground-Truth (GRS, do inglês Ground-Truth Recording Station) é o sistema responsável pela aquisição precisa dos dados de movimentação durante os experimentos. Ela é composta por dois sensores principais: uma câmera Intel® RealSense™ Depth Camera D455 e um LiDAR 3D Robosense RS-LiDAR-16. Ambos os sensores estão montados em uma estrutura personalizada impressa em 3D, projetada para garantir estabilidade e alinhamento adequados. Essa estrutura é fixada sobre um tripé, permitindo fácil transporte e instalação em diferentes locais do ambiente.

Figura 3.2: Ground-Truth Recording Station

Para reduzir o risco de oclusões parciais causadas pelos próprios participantes ou por elementos do ambiente, a GRS foi posicionada a uma altura de 2,17 metros em relação ao solo. Além disso, visando maximizar a resolução e a densidade dos dados capturados pelo LiDAR na região de interesse, a estação foi inclinada em 15° em direção ao chão.

A Figura 3.2 apresenta a GRS utilizada nos experimentos, com destaque para a montagem dos sensores e a orientação do conjunto em relação à área de coleta de dados.

3.1.3 Robôs utilizados nos experimentos

Os experimentos realizados para a construção do dataset contaram com dois robôs principais: o Clearpath Jackal e o Toyota Human Support Robot (HSR). Cada um foi escolhido por suas características complementares, possibilitando a simulação de diferentes perfis de interação robô-humano em cenários de navegação social. O Jackal apresenta cinemática diferencial e o HSR é holonômico.

O Clearpath Jackal é um robô compacto, robusto e com tração diferencial, projetado para operar em ambientes internos e externos. Ele está equipado com um sensor LiDAR 3D e uma câmera RGB-D, sendo amplamente utilizado em pesquisas relacionadas à navegação autônoma, mapeamento e percepção do ambiente.

Já o Toyota HSR é um robô assistivo mais complexo, voltado para a interação segura e inteligente com seres humanos. Ele combina uma base móvel, um braço manipulador e uma variedade de sensores que o tornam adequado para tarefas domésticas e colaborativas em ambientes compartilhados com pessoas.

Durante os experimentos, ambos os robôs foram teleoperados por um mesmo operador humano. Para cada cenário registrado, a navegação foi realizada duas vezes: primeiro de forma

Figura 3.3: Robôs utilizados nos experimentos

socialmente consciente — respeitando normas implícitas de convívio e evitando interferência nos caminhos dos pedestres — e depois de forma não consciente, ignorando essas normas. Essa abordagem permitiu a coleta de dados contrastantes, essenciais para a posterior identificação dos parâmetros do modelo de forças sociais. A Figura 3.3 ilustra os dois robôs utilizados nos experimentos.

3.2 Descrição dos experimentos

Os experimentos foram projetados com o objetivo de registrar interações entre humanos e robôs em cenários típicos de navegação social. Os cenários adotados foram originalmente propostos por Francis et al. (2023) no artigo *Principles and Guidelines for Evaluating Social Robot Navigation Algorithms*. Além de facilitar a coleta de dados em situações realistas, esses cenários servem como base padronizada para a avaliação do desempenho de algoritmos de navegação social.

A Figura 3.1 ilustra os principais cenários implementados no dataset, incluindo situações como aproximação frontal, cruzamentos perpendiculares, contornos de esquinas cegas e obstrução por pedestres.

Os cenários utilizados nos experimentos foram projetados para representar situações típicas de navegação social encontradas em ambientes compartilhados entre robôs e humanos. Cada cenário visa capturar um tipo específico de interação, como aproximações frontais, cruzamentos perpendiculares, contornos de esquinas com visibilidade restrita, bloqueios intencionais, navegação em grupo ou entrega de objetos. Essa diversidade permite explorar tanto comportamentos socialmente adequados quanto inadequados por parte do robô, possibilitando a coleta de dados contrastantes e representativos. A adoção desses cenários foi inspirada nas diretrizes pro-

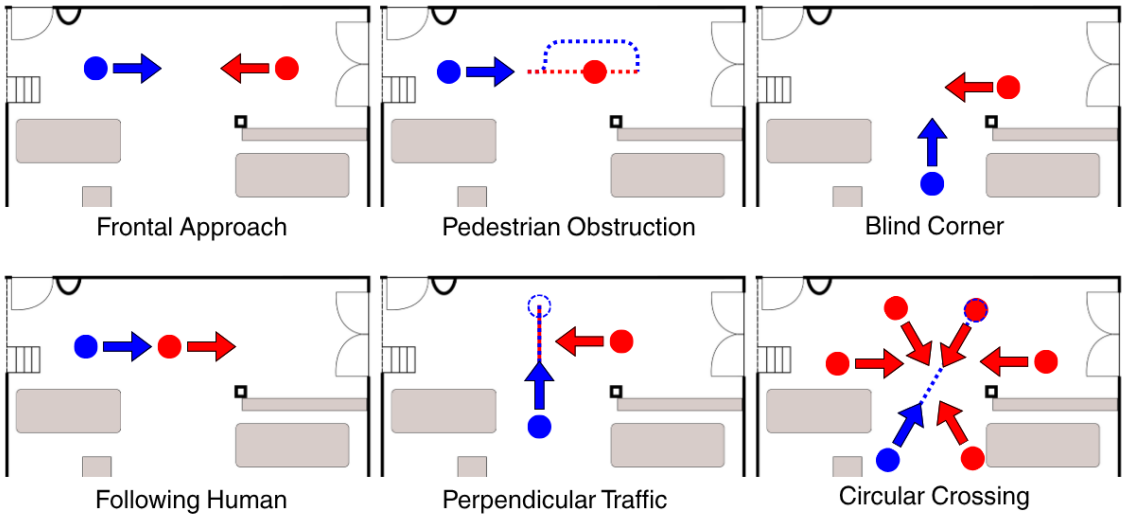

Figura 3.4: Cenários de navegação social propostos por Francis et al. (2023)

postas por Francis et al. (2023), e sua implementação foi cuidadosamente adaptada ao ambiente físico e aos recursos disponíveis neste trabalho. Ao variar fatores como a direção dos movimentos, a quantidade de pedestres, e o nível de colaboração entre agentes, os cenários tornam-se especialmente úteis para análise de respostas sociais, treinamento de modelos e avaliação de algoritmos de navegação. Uma descrição criteriosa dos cenários pode ser vista na tabela 3.1.

Cada cenário foi cuidadosamente configurado com metas de navegação específicas, promovendo diferentes padrões de interação entre humanos e robôs. Ao todo, 17 participantes foram envolvidos, numerados de 1 a 17. Eles foram organizados em dois grupos de cinco pessoas e cinco duplas, que executaram todos os cenários, exceto o de *object handover* (entrega de objeto). O robô pegou e entregou uma série de objetos com marcadores ”AR”, como pode ser visto na Figura 3.5.

Figura 3.5: Objetos utilizados no cenário *object handover*

Tabela 3.1: Cenários de navegação social utilizados no NavWareSet.

Nome	**Descrição**	**Layout**	**Objetivo**	**Tarefa Robô**
Frontal Approach	Robô e pedestre se aproximam frontalmente	Espaço transitável	Interação com pedestres	Navegar A→B
Pedestrian Obstruction	Um pedestre bloqueia o caminho do robô	Espaço transitável	Interação com pedestres	Navegar A→B
Blind Corner	Robô e pedestre se encontram em um canto cego	Esquina	Interação com pedestres	Navegar A→B
Perpendicular Traffic	Pedestre cruza perpendicularmente o robô	Interseção	Interação com pedestres	Navegar A→B
Following Human	Robô segue um humano	Espaço de caminhada	Navegação conjunta	Seguir humano
Circular Crossing	Robô e humanos cruzam no centro do espaço	Espaço transitável	Navegação em multidão	Navegar A→B
Object Handover	Robô entrega ou recebe um objeto	Espaço transitável	Navegação interativa	Entregar/receber objeto

Os experimentos foram conduzidos utilizando dois robôs: o Toyota HSR e o Clearpath Jackal. Na maioria dos cenários, os robôs foram teleoperados duas vezes: uma de forma socialmente consciente, respeitando regras de convivência como manutenção de distância interpessoal e desvio suave de trajetórias; e outra de forma não social, utilizando uma navegação direta entre pontos, desconsiderando a presença humana.

A Tabela 3.2 apresenta uma descrição detalhada dos cenários implementados no dataset. Os grupos G1 e G2 representam subconjuntos distintos de participantes (numerados de 1 a 11), que atuaram em todas as cenas padrão. Já as duplas identificadas como “Par 1” a “Par 5” foram formadas especificamente para o cenário de entrega de objeto.

Tabela 3.2: Cenas gravadas no dataset.

Nº	**Nome**	**Cenário**	**Robô**	**Soc.**	**Grupo**	**Participantes**
1	Scene 1	Frontal Approach	HSR	Sim	G1	1–5
2	Scene 2	Pedestrian Obstruction	HSR	Sim	G1	1–5
3	Scene 3	Blind Corner	HSR	Sim	G1	1–5
4	Scene 4	Following Human	HSR	Sim	G1	1–5
5	Scene 5	Perpendicular Traffic	HSR	Sim	G1	1–5
6	Scene 6	Circular Crossing	HSR	Sim	G1	1–5
7	Scene 8	Frontal Approach	HSR	Não	G1	1–5
8	Scene 9	Pedestrian Obstruction	HSR	Não	G1	1–5
9	Scene 10	Blind Corner	HSR	Não	G1	1–5

Continuação na próxima página

Nº	Nome	Cenário	Robô	Soc.	Grupo	Participantes
10	Scene 12	Perpendicular Traffic	HSR	Não	G1	1–5
11	Scene 13	Circular Crossing	HSR	Não	G1	1–5
12	Scene 15	Frontal Approach	Jackal	Sim	G1	1,2,3,5,6
13	Scene 16	Pedestrian Obstruction	Jackal	Sim	G1	1,2,3,5,6
14	Scene 17	Blind Corner	Jackal	Sim	G1	1,2,3,5,6
15	Scene 18	Following Human	Jackal	Sim	G1	1,2,3,5,6
16	Scene 19	Perpendicular Traffic	Jackal	Sim	G1	1,2,3,5,6
17	Scene 20	Circular Crossing	Jackal	Sim	G1	1,2,3,5,6
18	Scene 21	Frontal Approach	Jackal	Não	G1	1,2,3,5,6
19	Scene 22	Pedestrian Obstruction	Jackal	Não	G1	1,2,3,5,6
20	Scene 23	Blind Corner	Jackal	Não	G1	1,2,3,5,6
21	Scene 25	Perpendicular Traffic	Jackal	Não	G1	1,2,3,5,6
22	Scene 26	Circular Crossing	Jackal	Não	G1	1,2,3,5,6
23	Scene 27	Frontal Approach	HSR	Sim	G2	7–11
24	Scene 28	Pedestrian Obstruction	HSR	Sim	G2	7–11
25	Scene 29	Blind Corner	HSR	Sim	G2	7–11
26	Scene 30	Following Human	HSR	Sim	G2	7–11
27	Scene 31	Perpendicular Traffic	HSR	Sim	G2	7–11
28	Scene 32	Circular Crossing	HSR	Sim	G2	7–11
29	Scene 34	Frontal Approach	HSR	Não	G2	7–11
30	Scene 35	Pedestrian Obstruction	HSR	Não	G2	7–11
31	Scene 36	Blind Corner	HSR	Não	G2	7–11
32	Scene 38	Perpendicular Traffic	HSR	Não	G2	7–11
33	Scene 39	Circular Crossing	HSR	Não	G2	7–11
34	Scene 41	Frontal Approach	Jackal	Sim	G2	7–11
35	Scene 42	Pedestrian Obstruction	Jackal	Sim	G2	7–11
36	Scene 43	Blind Corner	Jackal	Sim	G2	7–11
37	Scene 44	Following Human	Jackal	Sim	G2	7–11
38	Scene 45	Perpendicular Traffic	Jackal	Sim	G2	7–11
39	Scene 46	Circular Crossing	Jackal	Sim	G2	7,8,9,10,12
40	Scene 47	Frontal Approach	Jackal	Não	G2	7–11
41	Scene 49	Blind Corner	Jackal	Não	G2	7–11
42	Scene 51	Perpendicular Traffic	Jackal	Não	G2	7–11
43	Scene 52	Circular Crossing	Jackal	Não	G2	7,8,9,10,12
44	Scene 53	Object Handover	HSR	Sim	Par 1	2, 8
45	Scene 54	Object Handover	HSR	Sim	Par 2	1, 9

Continuação na próxima página

Nº	Nome	Cenário	Robô	Soc.	Grupo	Participantes
46	Scene 55	Object Handover	HSR	Sim	Par 3	13,14
47	Scene 56	Object Handover	HSR	Sim	Par 4	4, 15
48	Scene 57	Object Handover	HSR	Sim	Par 5	16,17

Fim da Tabela 3.2

3.2.1 Condições sociais, teleoperação e combinações de cenários

Embora a Tabela 3.2 liste as cenas gravadas, ela não explicita como as versões *conformista* e *não conformista* foram conduzidas nem quais combinações de cenário, robô e condição social estão disponíveis. Na navegação **conformista**, o operador teleopera o robô observando os participantes e mantendo uma distância interpessoal mínima (aproximadamente 1,2 m) sempre que possível. Em aproximações frontais (*frontal approach*), o robô reduz a velocidade quando se aproxima do pedestre e realiza um desvio suave para a esquerda ou direita conforme definido no roteiro experimental, retomando seu rumo inicial após a passagem. Em cruzamentos perpendiculares (*Perpendicular Traffic*) e contornos de esquina (*blind corner*), o operador antecipa a interação e escolhe o lado de desvio que maximize o conforto visual do pedestre (por exemplo, passando à direita no sistema de circulação local). Já na navegação **não conformista**, o teleoperador controla o robô para seguir a trajetória reta mais curta entre origem e destino sem considerar a presença ou conforto dos participantes. Não há redução de velocidade nem desvio lateral intencional; cabe às pessoas evitarem o robô. Essa distinção de comportamento permite contrastar trajetórias geradas com e sem respeito a normas sociais.

Por razões práticas, nem todos os cenários possuem versões não conformistas. Em particular, os cenários *following human* (seguindo humano) e *object handover* (entrega de objeto) exigem cooperação explícita do pedestre; nesse caso, a definição de uma versão "não conformista" – na qual o robô simplesmente ignora a pessoa e segue em frente – não se aplica. A Figura 3.7 apresenta um diagrama resumindo as combinações de cenário, robô e condição social presentes no *NavWareSet*. Cada célula colorida indica que a respectiva combinação está disponível no dataset; células em branco indicam que a combinação não foi gravada.

Além disso, os grupos G1 e G2 listados na Tabela 3.2 representam dois subconjuntos de participantes que participaram de todas as cenas padrão (excepto entrega de objeto). O grupo G1 é composto pelos participantes 1 a 6, enquanto o grupo G2 abrange os participantes 7 a 12. As "duplas" Par 1 a Par 5 foram formadas especificamente para o cenário de entrega de objeto, em que dois participantes interagem com o robô em tarefas cooperativas.

3.2.2 Desafios na anotação automática e opção por anotação manual

Inicialmente, foram avaliadas técnicas automáticas de detecção de pedestres em nuvens de pontos para extrair as trajetórias humanas. Métodos baseados em agrupamento de pon-

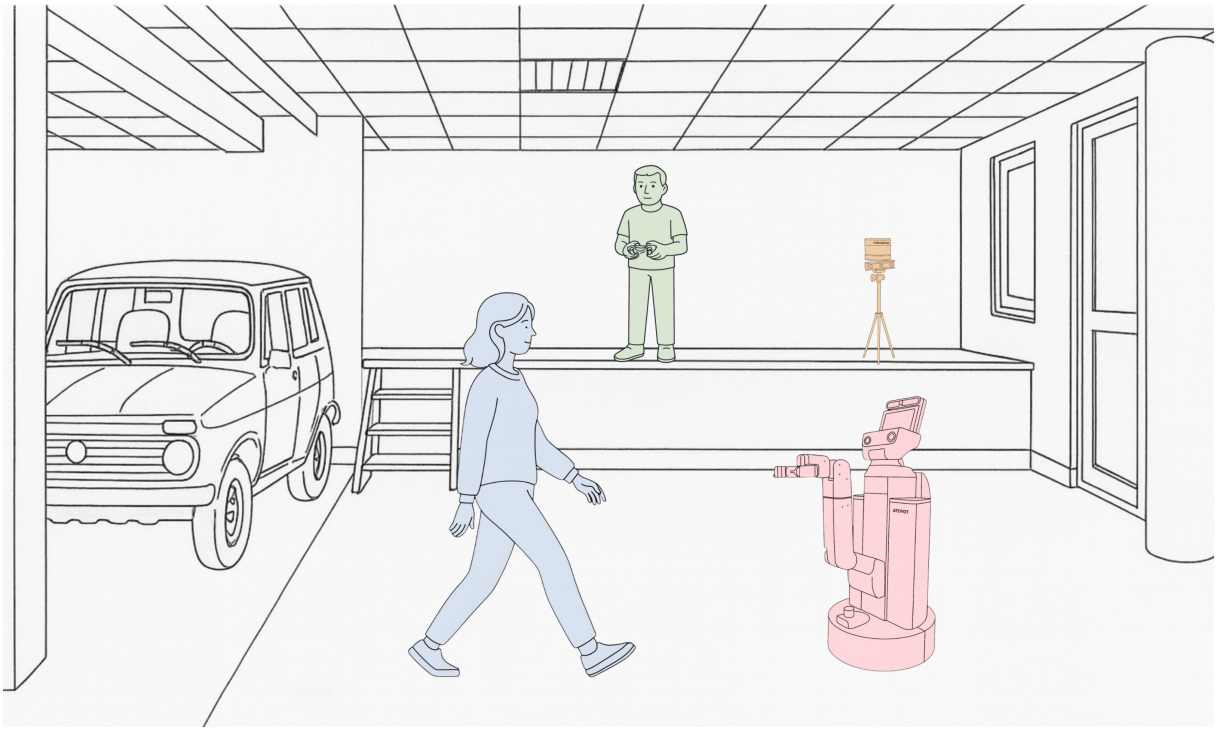

Figura 3.6: Imagem ilustrativa do processo de teleoperação

tos (clustering) e segmentação de séries temporais apresentaram desempenho instável devido a oclusões frequentes, reflexos no piso e baixa resolução do sensor em longas distâncias. Abordagens baseadas em aprendizado profundo, treinadas em datasets como L-CAS e JRDB, também falharam em detectar corretamente os participantes quando estes ficavam parcialmente ocultos pelo robô ou se aproximavam das extremidades do campo de visão. Dada a necessidade de precisão centimétrica para a calibração de modelos, optou-se pela anotação manual das trajetórias. Essa anotação foi realizada quadro a quadro sobre as nuvens de pontos utilizando a ferramenta CVAT, resultando em coordenadas XY sincronizadas com as poses dos robôs. Embora mais demorado, o processo manual assegurou a qualidade do *ground truth*.

3.3 Pipeline de Dados

O processo de construção do dataset foi estruturado em uma sequência bem definida de etapas, que vão desde o planejamento dos experimentos até a disponibilização pública dos dados anotados. Cada etapa foi cuidadosamente organizada com o objetivo de garantir a qualidade, consistência e reprodutibilidade do conjunto de dados final. Um fluxograma com cada fase da pipeline pode ser visto na Figura 3.8

3.3.1 Planejamento Experimental

O processo tem início com o planejamento detalhado dos cenários de navegação social em um ambiente interno controlado. Essa fase envolve a definição dos tipos de interações a se-

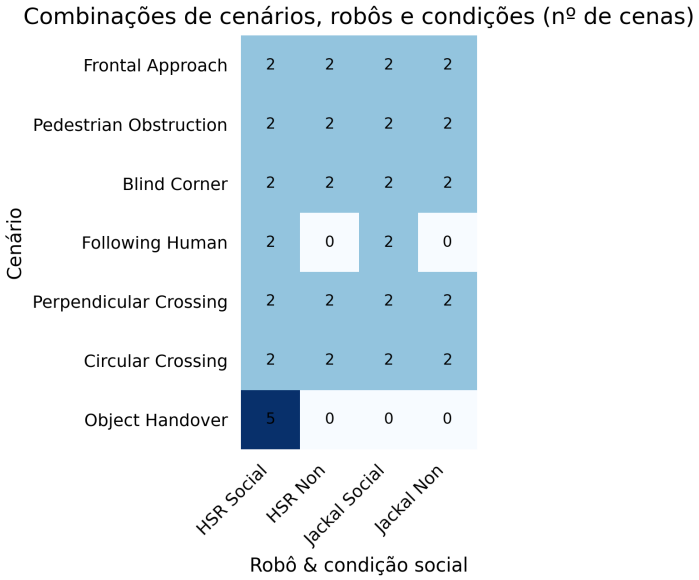

Figura 3.7: Diagrama de combinações de cenários, robôs e condição social do NavWareSet. Cada linha corresponde a um cenário (nomes traduzidos entre parênteses); cada coluna corresponde a uma combinação de robô (HSR ou Jackal) e condição social (S – social/conformista; N – não social/não conformista). Células coloridas indicam que a cena foi executada e coletada; células em branco indicam que a combinação não está presente.

rem registradas, com base em situações recorrentes na literatura, como cruzamentos, obstruções, seguimento e entregas de objetos. São definidos os papéis desempenhados por cada agente — robôs e participantes humanos —, especificando-se se a interação será socialmente adequada (ex.: respeitando distâncias interpessoais e evitando movimentos bruscos) ou inadequada (ex.: ignorando normas sociais). Também são escolhidos os pontos de origem e destino dos trajetos, a configuração espacial do ambiente e os grupos de participantes atribuídos a cada cena. Esse planejamento busca garantir diversidade, repetibilidade e controle sobre as variáveis experimentais.

3.3.2 Coleta de Dados

Na fase de coleta, os robôs **Toyota HSR** e **Clearpath Jackal** são operados em tempo real por um operador humano treinado, seguindo os roteiros definidos no planejamento. A **Ground-Truth Recording Station (GRS)** é posicionada estrategicamente para obter uma visão ampla da cena, com sensores montados sobre um tripé estável a 2,17 m de altura e inclinados em 15° em direção ao chão. Os robôs registram dados internos de seus sensores embarcados:

- LiDAR 3D (percepção de profundidade e obstáculos);
- Câmeras RGB-D (visão colorida com profundidade);
- Câmeras estéreo (visão binocular);
- IMU (orientação e aceleração angular);
- Odometria e comandos de velocidade.

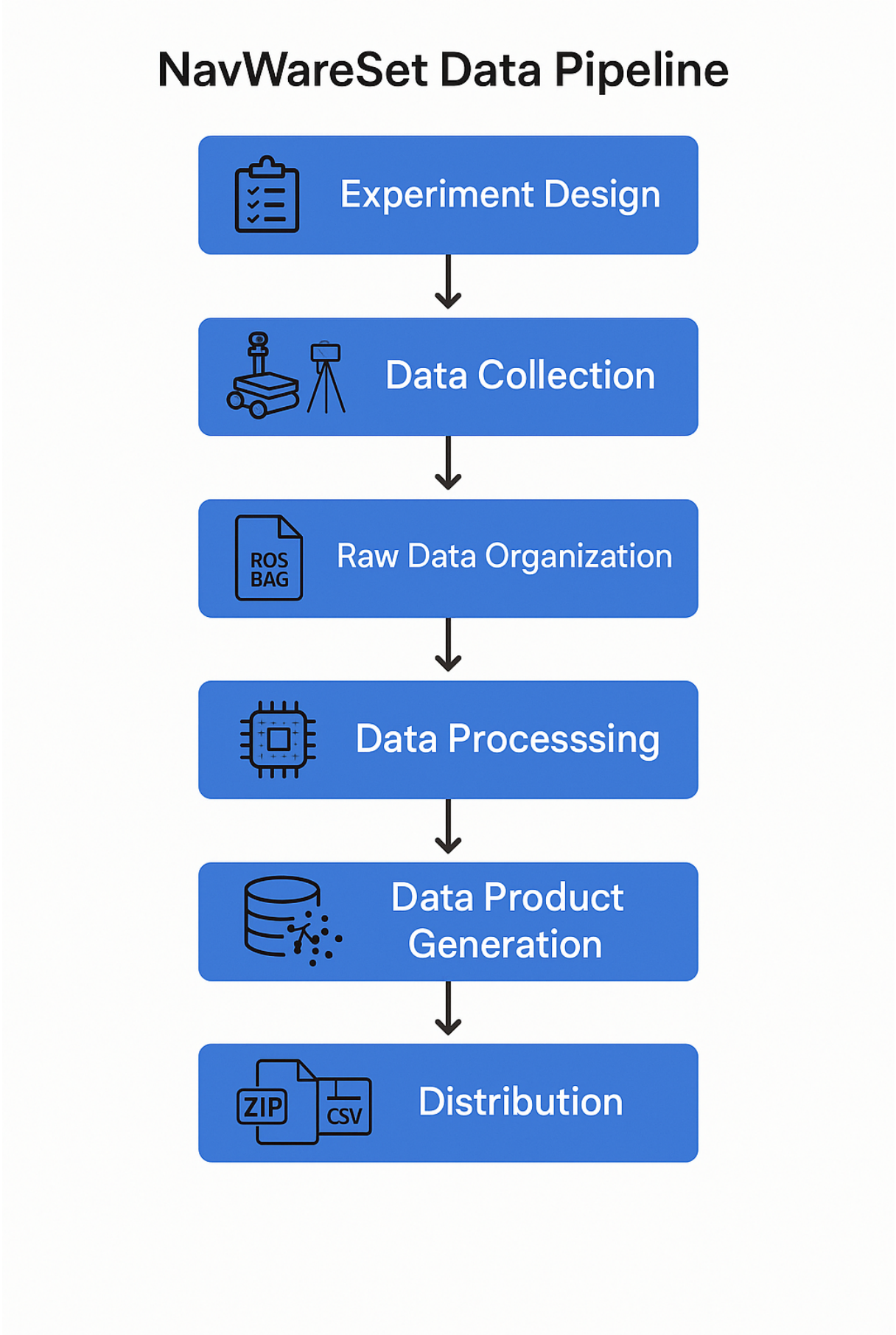

Figura 3.8: Pipeline de Dados

Paralelamente, a GRS registra vídeo em alta resolução e nuvem de pontos densas do ambiente externo. A sincronização temporal entre os dados dos robôs e da GRS é garantida por marcações de início/fim manual e verificação posterior via timestamps, permitindo correlação precisa entre as fontes.

3.3.3 Organização dos Dados Brutos

Após a coleta, os dados são organizados em dois arquivos principais no formato `ROS bag` para cada cena:

- Um arquivo com os dados registrados pelo robô, contendo os sensores embarcados;
- Um arquivo com os dados da GRS, contendo LiDAR e vídeo externos.

Os arquivos são nomeados de forma padronizada, contendo identificadores de cena, robô, condição social e grupo de participantes. Essa estrutura modular facilita buscas, automação do processamento e verificação cruzada dos dados, além de manter independência entre as fontes sensoriais.

3.3.4 Processamento dos Dados

A etapa de processamento envolve tanto métodos automáticos quanto manuais. Inicialmente, as poses dos robôs ao longo do tempo são estimadas por algoritmos de SLAM (*Simultaneous Localization and Mapping*), com base na fusão de odometria, IMU e LiDAR. Em seguida, os dados da GRS são usados para gerar nuvens de pontos espaciais, que servem como referência para anotação.

A anotação da posição dos participantes humanos é realizada de forma manual, quadro a quadro, utilizando a ferramenta **CVAT** aplicada sobre as nuvens de pontos 3D. As anotações incluem as coordenadas XY dos pedestres ao longo do tempo, e são exportadas no formato **Supervisely**, que permite posterior conversão para arquivos CSV e integração com ferramentas de análise. Todo o processo é validado visualmente para garantir precisão e coerência.

3.3.5 Geração dos Produtos de Dados

Com os dados processados e anotados, são gerados os seguintes produtos por cena:

- **Nuvens de pontos 3D anotadas**, alinhadas às poses dos robôs;
- **Arquivos** `.csv` com trajetórias temporais dos robôs e dos pedestres;
- **Mapas de ocupação** com obstáculos estáticos representados como matrizes binárias;
- **Arquivos auxiliares de calibração** entre sensores, incluindo transformações extrínsecas;
- **Metadados** com descrição dos participantes, robô utilizado, tipo de cenário, condição social e tempos de gravação.

Esses arquivos tornam o dataset adequado tanto para análise qualitativa quanto para uso quantitativo em algoritmos supervisionados ou simulações.

3.3.6 Distribuição

O conjunto de dados completo é disponibilizado publicamente em três plataformas:

1. O **site oficial** do projeto[1] fornece uma visão geral, vídeos ilustrativos e documentação de referência;
2. O **repositório no GitHub**[2] contém scripts de leitura, visualização e conversão de dados;
3. A plataforma institucional **dat@UBFC**[3] oferece acesso aos arquivos completos, com opção de download em lote ou por cena individual.

Todos os materiais são licenciados de forma aberta para promover reprodutibilidade, transparência e reutilização em pesquisas futuras.

3.4 Implementação do SFM no UAIbot

O modelo de forças sociais foi implementado no simulador UAIbot com o objetivo de permitir a simulação de interações realistas entre pedestres e obstáculos. A arquitetura adotada combina componentes visuais, cinemáticos e de simulação física para modelar agentes móveis com comportamentos orientados por forças.

3.4.1 Estrutura do Agente

Cada pedestre (Figura 3.9) foi modelado como um objeto da classe `Pedestrian`, que herda propriedades gráficas e geométricas da classe `Cylinder`. Essa classe encapsula não apenas a aparência visual do agente no ambiente 3D, mas também todos os parâmetros necessários para calcular as forças sociais.

Os principais atributos incluem:

- Posição atual ($\vec{r}_a$) e objetivo ($\vec{g}$);
- Velocidade atual ($\vec{v}_a$) e velocidade desejada (v_a^0);
- Tempo de relaxação (τ_a) e massa (m_a);

3.4.2 Forças Modeladas

A implementação segue a formulação clássica do Social Force Model, somando forças de atração e repulsão:

- **Força de atração ao objetivo**: calculada pelo método `fag()`, aplica uma aceleração que direciona o pedestre de sua posição atual rumo à posição desejada com velocidade v_a^0.
- **Força de repulsão**: o método `fdaq()` computa a força de repulsão com base na distância entre o agente e o obstáculo (ou outro pedestre), usando uma função exponencial decres-

[1] `https://anr-navware.github.io/navwareset/`
[2] `https://github.com/anr-navware`
[3] `https://search-data.ubfc.fr/FR-13002091000019-2025-05-22`

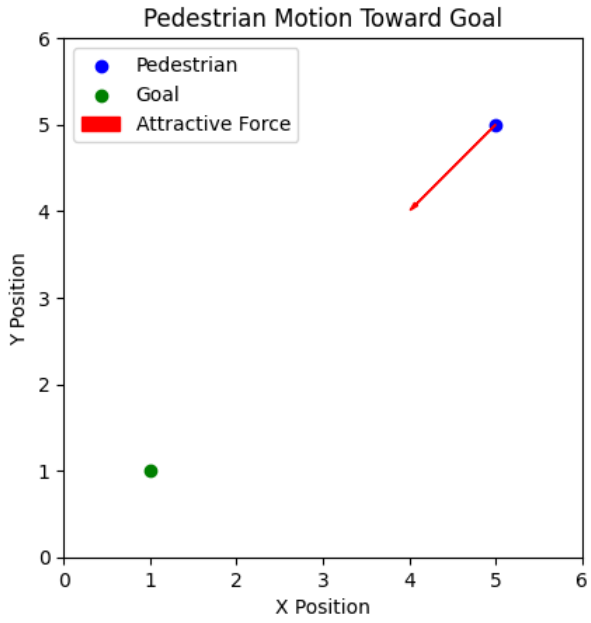

Figura 3.9: Estrutura do Agente.

cente.

- **Ponderação direcional**: a função `w()` aplica um fator de anisotropia às forças repulsivas, penalizando interações que ocorrem fora do campo de visão frontal do pedestre.

A força total resultante aplicada ao agente é obtida pela soma vetorial de todas as componentes, considerando a influência simultânea de múltiplos obstáculos e pedestres.

3.4.3 Modelagem dos Obstáculos

Dois tipos de obstáculos foram implementados:

- **Colunas** (Figura 3.10) (`ObstacleColumn`): objetos cilíndricos estáticos com raio e altura definidos, herdando da classe `Cylinder`.
- **Paredes finas** (`ObstacleThinWall`): objetos do tipo `Box` definidos por dois pontos extremos e altura. O cálculo de distância é feito geometricamente com base na projeção ortogonal de um ponto em um segmento.

Cada obstáculo implementa o método `d()`, responsável por calcular o vetor mínimo entre sua superfície e a borda do pedestre, o que é essencial para o cálculo da força de repulsão.

3.4.4 Cálculo da Distância e Animação

Todos os objetos (pedestres e obstáculos) compartilham um mecanismo comum para o cálculo da distância vetorial mínima entre seus volumes e o pedestre. Esse cálculo é feito geometricamente e atualizado a cada iteração de simulação. O resultado das forças é utilizado para atualizar a velocidade e posição dos agentes ao longo do tempo.

A animação e visualização das simulações são feitas no UAIbot, com as trajetórias dos

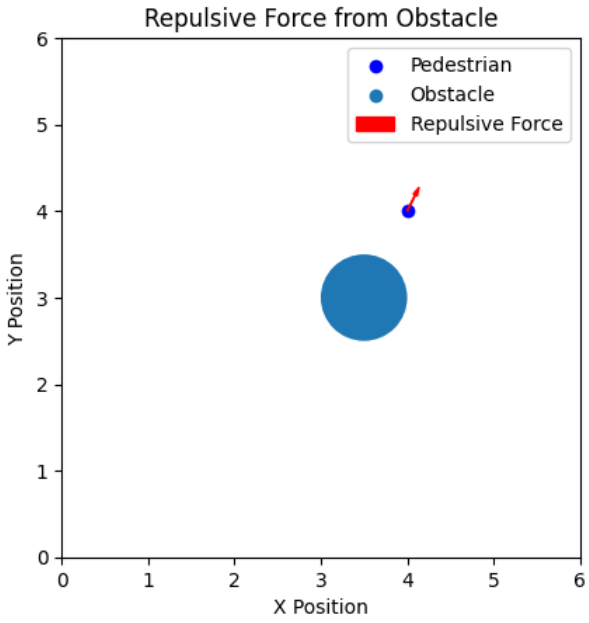

Figura 3.10: Exemplo de obstaculo repelindo um pedestre

agentes sendo atualizadas de acordo com a integração das equações de movimento geradas pelas forças aplicadas.

3.4.5 Parâmetros e Ajustabilidade

Todos os parâmetros do modelo podem ser ajustados individualmente por agente, o que permite simular perfis de comportamento diferenciados, como agentes mais cautelosos, apressados ou indiferentes à presença de obstáculos. Essa flexibilidade foi explorada durante os experimentos de identificação de parâmetros descritos nos capítulos seguintes.

3.5 Otimização dos parâmetros do modelo

Com o objetivo de ajustar o modelo de forças sociais (Social Force Model — SFM) aos dados reais coletados durante os experimentos, foi implementado um processo de otimização numérica utilizando simulações no simulador UAIbot. Essa etapa visou encontrar os valores dos parâmetros que minimizam a diferença entre as trajetórias previstas pelo modelo e as trajetórias reais observadas nos dados.

3.5.1 Definição do Problema de Otimização

A identificação dos parâmetros foi formulada como um problema de minimização de função de custo. A função-objetivo computa a soma dos erros ao longo do tempo entre as posições previstas pelo modelo e as posições reais dos pedestres registradas nos arquivos `.csv`. O objetivo é minimizar essa soma total de distâncias para um conjunto de cenas.

Os parâmetros otimizados foram:

- τ_a — tempo de relaxação do pedestre;
- v_a^0 — velocidade desejada;
- a_i — amplitude da força repulsiva com obstáculos;
- b_i — fator de decaimento da força repulsiva com obstáculos;
- λ_a — parâmetro de anisotropia direcional.

3.5.2 Procedimento de Simulação

Para cada arquivo de trilha (`track_scene*.csv`), os dados da posição inicial, posição final e trajetória do robô foram utilizados para inicializar uma simulação no UAIbot com um agente do tipo `Pedestrian` e um obstáculo do tipo `ObstacleColumn`, representando o robô.

Durante a simulação:

- A força de atração ao objetivo (`fag`) e a força de repulsão com o robô (`fdaq`) foram computadas em cada passo de tempo;
- A posição e a velocidade do pedestre foram atualizadas com base na integração explícita (método de Euler);
- A trajetória prevista foi comparada com a trajetória real, computando-se o erro euclidiano ponto a ponto;
- O somatório desses erros para todas as amostras compôs o valor de custo utilizado na otimização.

3.5.3 Estratégia de Otimização

A identificação dos parâmetros foi realizada por meio da minimização de uma função de custo baseada na distância entre as posições simuladas e as posições reais dos pedestres. A função objetivo utilizada na otimização pode ser expressa como:

$$\{\tau_a, v_a^0, a_i, b_i, \lambda_a\} = \arg\min_{\tau_a, v_a^0, a_i, b_i, \lambda_a} \left\{ \sum_{n=1}^{N} \sum_{t} \left\| \vec{x}_{\text{sim}}^{(n)}(t) - \vec{x}_{\text{real}}^{(n)}(t) \right\| \right\} \tag{3.1}$$

onde:

- N representa o número de trajetórias analisadas;
- t é o tempo discreto durante a simulação;
- $\vec{x}_{\text{sim}}^{(n)}(t)$ é a posição simulada do pedestre no instante t para a cena n;
- $\vec{x}_{\text{real}}^{(n)}(t)$ é a posição real (extraída do dataset) para o mesmo instante e cena.

O objetivo da otimização é encontrar o conjunto de parâmetros que minimiza a soma total dos erros de predição ao longo do tempo e entre diferentes cenas. O valor da função objetivo corresponde ao erro total acumulado e foi utilizado como métrica principal para avaliar a qualidade do ajuste do modelo.

A otimização foi resolvida com o algoritmo `L-BFGS-B`, que é eficiente para problemas

de média escala com restrições nos parâmetros. Os limites de busca foram definidos conforme apresentado na Tabela 3.3, assegurando que os valores otimizados permanecessem dentro de faixas fisicamente plausíveis.

O algoritmo `L-BFGS-B` (*Limited-memory Broyden–Fletcher–Goldfarb–Shanno with Box constraints*) foi utilizado por sua eficiência na otimização de funções suaves com um número moderado de parâmetros. Ele permite impor limites superiores e inferiores aos parâmetros, garantindo que os valores resultantes estejam em faixas fisicamente plausíveis — definidas com base na literatura Helbing & Molnár (1995); Zanlungo et al. (2011) e em observações empíricas. Como a função objetivo pode conter múltiplos mínimos locais, múltiplas execuções foram realizadas com diferentes condições iniciais para aumentar a robustez da solução.

Tabela 3.3: Limites utilizados na otimização dos parâmetros do modelo SFM.

Parâmetro	**Mínimo**	**Máximo**
τ_a	0.1	2.0
v_a^0	0.5	2.5
a_i	0.1	10.0
b_i	0.1	10.0
λ_a	0.01	2.0

A função de custo final retorna o somatório total das diferenças entre as trajetórias reais e previstas, considerando todas as cenas analisadas. O resultado da otimização inclui os valores dos parâmetros ajustados globalmente (ou seja, compartilhados entre todas as cenas), além do valor mínimo encontrado para a função-objetivo.

Capítulo 4

Resultados

4.1 Introdução

Este capítulo apresenta os principais resultados obtidos ao longo do desenvolvimento deste trabalho. Inicialmente, é descrito em detalhes o conjunto de dados **NavWareSet**, que foi desenvolvido como parte fundamental da pesquisa, incluindo sua estrutura, formato e as visualizações geradas a partir das anotações. Em seguida, são apresentados os resultados relacionados à implementação do modelo de forças sociais (Social Force Model — SFM) no simulador UAIbot, destacando as funcionalidades implementadas e exemplos práticos de simulação. Por fim, são discutidos os resultados da etapa de otimização dos parâmetros do modelo, com base em dados reais extraídos do NavWareSet, além de uma comparação com valores de referência encontrados na literatura. O conjunto desses resultados demonstra a consistência da abordagem adotada e sua capacidade de simular interações realistas entre robôs e pedestres em cenários de navegação social.

4.2 NavWareSet

O conjunto de dados **NavWareSet** foi desenvolvido com o objetivo de fornecer uma base rica e realista para o estudo da navegação social de robôs. Ele documenta interações entre robôs e pedestres em ambientes internos controlados, explorando tanto comportamentos socialmente adequados quanto não adequados. O processo de coleta, anotação e organização dos dados resultou em um repositório robusto e de acesso público, com múltiplas modalidades de informação sincronizada.

4.2.1 Resumo Geral dos Dados

O NavWareSet abrange sete diferentes cenários de navegação social, envolvendo interações individuais e em grupo. As gravações foram realizadas utilizando dois robôs distintos — **Toyota HSR** e **Clearpath Jackal** — e uma estação fixa de registro (GRS). Os principais resultados

quantitativos são:

- Mais de **192 minutos** de dados brutos de interação;
- Mais de **172 minutos** de trajetórias humanas anotadas manualmente;
- Mais de **1000 trilhas individuais de pedestres** anotadas;
- Mais de **600 trilhas de robôs** com odometria e comandos de movimento;
- Registros com **sensores embarcados e externos**, incluindo LIDAR 3D, RGB-D, vídeo, odometria e mapas de ocupação;
- Registro de comportamento social e não social para cada cenário;
- Arquivos separados e organizados por cena, com metadados e arquivos auxiliares.

4.2.2 Formato e Organização dos Arquivos

Cada cena do **NavWareSet** é armazenada e disponibilizada de forma modular, visando facilitar o acesso, reutilização e reprodutibilidade por parte da comunidade científica. A estrutura de dados foi projetada para contemplar as diferentes modalidades sensoriais capturadas, os dados anotados manualmente, e as informações derivadas como mapas e poses estimadas. A seguir, descreve-se em detalhes os quatro principais pacotes de arquivos fornecidos para cada cena.

Arquivo `x_robot.zip`

Este arquivo contém a **ROS bag** com os dados sensoriais brutos registrados diretamente a partir do robô durante a execução da cena número *x*. Uma representação visual do conteúdo das *bags* feita pelo *Rviz* pode ser vista na duas imagens superiores da Figura 4.1. Os tópicos disponíveis variam conforme o robô utilizado:

- **Toyota HSR:**
  - Imagens RGB: `/hsrb/head_rgbd_sensor/rgb/image_rect_color`
  - Imagens estéreo: `/hsrb/head_l_stereo_camera/image_raw`, `/hsrb/head_r_stereo_camera`
  - Nuvem de pontos: `/hsrb/head_rgbd_sensor/depth_registered/rectified_points`
  - Imagens de profundidade: `/hsrb/head_rgbd_sensor/depth_registered/image_rect_raw`
  - Velocidades: `/hsrb/command_velocity`
  - Varredura a laser: `/hsrb/base_scan`
  - Mapas estáticos e dinâmicos: `/static_obstacle_ros_map`, `/dynamic_obstacle_map`
  - Informações de calibração e transformações: `/tf`, `/tf_static`
- **Clearpath Jackal:**
  - Imagens RGB: `/camera/color/image_raw`
  - Nuvem de pontos: `/rslidar_points`
  - Poses estimadas (AMCL): `/amcl_pose`
  - Comandos de velocidade: `/cmd_vel`
  - Mapa local: `/map`

  - Informações de calibração: `/camera/color/camera_info`
  - Transformações: `/tf`

Esses dados são essenciais para o replay da cena no ambiente ROS e para a extração de trajetórias, imagens e mapas a partir da perspectiva embarcada.

Arquivo `x_grs.zip`

Este pacote contém a **ROS bag** com os dados registrados pela Ground-Truth Recording Station (GRS), posicionada estaticamente em uma extremidade do ambiente. Ele fornece uma visão externa e independente da cena, o que é fundamental para validação e anotação posterior.

Os tópicos disponíveis incluem:

- Vídeo RGB da cena: `/camera/color/image_raw`
- Nuvem de pontos 3D do ambiente: `/rslidar_points`
- Informações de calibração da câmera: `/camera/color/camera_info`

Os dados da GRS permitem obter uma reconstrução precisa do ambiente, facilitando a anotação de trajetórias humanas com base na nuvem de pontos sincronizada com o tempo.

Arquivo `x_annotated.zip`

Este arquivo compreende os dados anotados manualmente, derivados a partir da nuvem de pontos gerada pela GRS, com a posição dos pedestres em cada quadro. A estrutura do diretório segue a seguinte organização:

```
x_annotated/
|-- scene_x/
|   |-- pointcloud/
|   |   |-- <timestamp>.pcd
|   |-- ann/
|   |   |-- <timestamp>.json
|-- x_robot_pose.csv
|-- x_grs_to_bot_offset.json
|-- x_occupancy_xy_points.json
|-- meta.json
`-- key_id_map.json
```

Uma explicação detalhada de cada arquivo pode ser encontrada a seguir:

- As nuvens de pontos (`.pcd`) estão organizadas por timestamp, representando a geometria do ambiente em cada instante;
- Os arquivos de anotação (`.json`) seguem o formato **Supervisely**, contendo as posições identificadas dos pedestres em cada quadro e seus respectivos cuboides;

- O arquivo `x_robot_pose.csv` inclui a pose estimada do robô (posição e orientação) sincronizada com os frames anotados;
- O arquivo `x_grs_to_bot_offset.json` define o offset espacial entre os sensores da GRS e do robô, essencial para alinhamento das informações gravadas de diferentes plataformas;
- O arquivo `x_occupancy_xy_points.json` define os pontos que compõem o mapa de ocupação estático da cena;
- O arquivo `meta.json` armazena metadados relativos ao formato Supervisely;
- O `key_id_map.json` estabelece a correspondência entre os IDs usados nas anotações.

Arquivo `x_poses.zip`

Este diretório contém os dados consolidados de posição dos pedestres e do robô ao longo do tempo, extraídos dos dados anotados e sincronizados. A estrutura é mais enxuta e segue o seguinte padrão:

```
x_poses/
|-- x_occupancy_xy_points.json
|-- x_robot_and_participants.csv
```

O arquivo `x_robot_and_participants.csv` organiza as posições no seguinte formato:

```
timestamp, robot_x, robot_y, robot_yaw_rad, x1, y1, x2, y2, ...
```

Cada linha do arquivo representa um instante de tempo, indicando a pose do robô (posição *x*, posição *y* e orientação em radianos), seguida pelas posições dos pedestres presentes no ambiente naquele instante.

A Tabela 4.1 apresenta um exemplo real de conteúdo extraído de uma cena do dataset:

Tabela 4.1: Exemplo de entradas no arquivo de posições dos participantes e poses do robô

.

timestamp	**r_x**	**r_y**	**yaw**	**x1**	**y1**	**x2**	**y2**	**x3**	**y3**	**x4**	**y4**	**x5**	**y5**
1730220611292	2.7051	-0.8938	-0.0046	1.44	0.55	0.68	-1.16	10.65	0.18	11.09	-1.56	10.92	-2.65
1730220611392	2.7051	-0.8938	-0.0046	1.43	0.55	0.68	-1.16	10.65	0.18	11.09	-1.56	10.92	-2.65
1730220611492	2.7051	-0.8938	-0.0046	1.43	0.55	0.68	-1.16	10.65	0.18	11.09	-1.56	10.92	-2.65

Nesta tabela:

- As colunas `r_x`, `r_y` e `yaw` representam a posição e orientação do robô.
- Os pares x*i*, y*i* correspondem às coordenadas (x, y) dos pedestres no instante de tempo indicado.

- O campo `timestamp` segue o padrão de tempo UNIX em nanossegundos, alinhado com os dados dos arquivos ROS bag e das anotações.

Cenas do tipo “Object Handover” (cenas 53 a 57) não incluem posições de participantes, uma vez que os humanos permanecem estáticos e apenas o robô se movimenta entre os alvos de entrega.

4.2.3 Visualizações dos Dados

A Figura 4.1 apresenta exemplos visuais do conteúdo do NavWareSet. Os dados demonstram boa qualidade espacial e temporal, com fidelidade nas anotações e sincronização precisa entre as modalidades sensoriais. O processo de anotação com CVAT foi realizado manualmente, garantindo a precisão das trajetórias humanas. As informações adicionais, como mapas de obstáculos, offsets de calibração e identificadores de participantes, são incluídas para facilitar o uso por terceiros. Scripts para a visualização dos dados, como mostrado na figura 4.1 podem ser acessados no repositório do projeto[1].

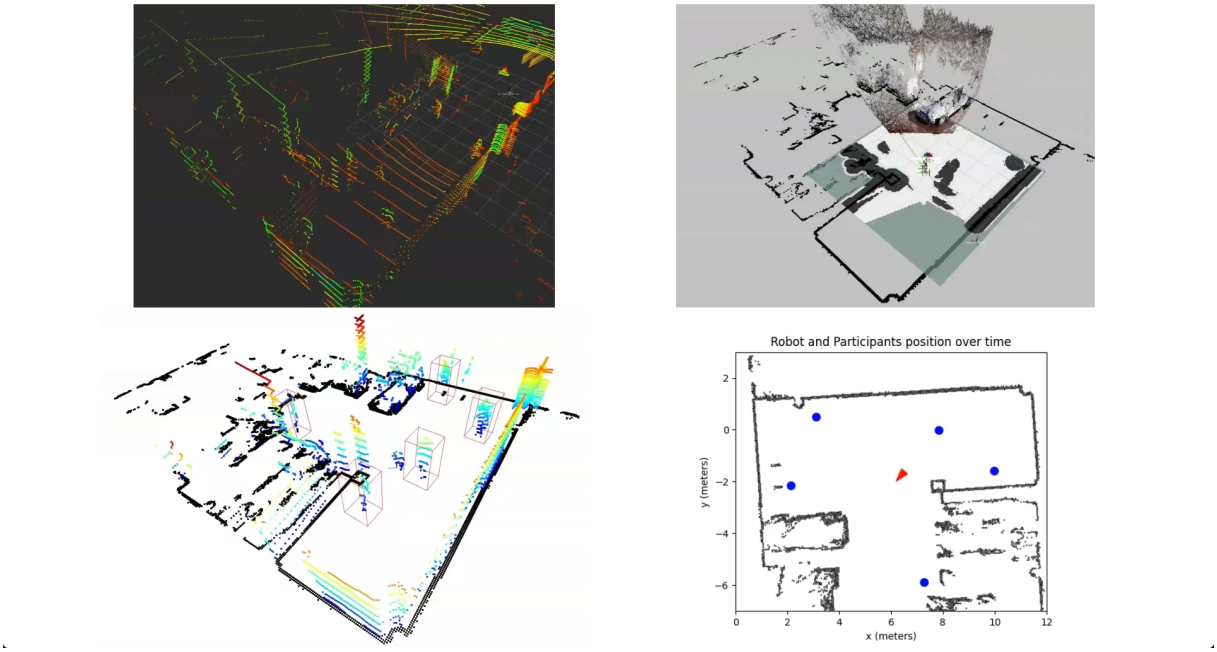

Figura 4.1: Exemplos de visualizações extraídas do NavWareSet. Superior esquerdo: nuvem de pontos LIDAR do ambiente; Superior direito: dados sensoriais captados pelo robô; Inferior esquerdo: nuvem de pontos sincronizada com as posições dos pedestres e pose do robô; Inferior direito: trajetórias extraídas do robô e dos participantes sobre o mapa do ambiente.

[1] https://github.com/anr-navware/NavWareSet-Tutorials

4.3 Resultados da Implementação do Social Force Model no UAIbot

A implementação do modelo de forças sociais (Social Force Model — SFM) no simulador UAIbot possibilitou a simulação interativa de agentes pedestres em ambientes com obstáculos estáticos ou dinâmicos. Essa seção apresenta os resultados funcionais obtidos com a implementação, destacando cada método desenvolvido e fornecendo exemplos de uso para facilitar sua replicação e reutilização.

Antes de utilizar o simulador UAIbot, é necessário instalá-lo via `pip`. O pacote `uaibot` está disponível no PyPI e pode ser instalado com o seguinte comando:

```
pip install uaibot==1.2.4
```

Certifique-se de estar utilizando uma versão atualizada do Python (≥ 3.7) e, se necessário, recomenda-se a criação de um ambiente virtual para evitar conflitos com outras bibliotecas. O código-fonte completo, bem como tutoriais e exemplos adicionais, está disponível no repositório oficial do projeto[2].

4.3.1 Criação de Pedestres e Obstáculos

A classe `Pedestrian` foi construída como uma subclasse da primitiva `Cylinder`, incorporando atributos físicos e sociais de agentes móveis. Da mesma forma, obstáculos foram representados por duas classes distintas: `ObstacleColumn` e `ObstacleThinWall`, herdando respectivamente de `Cylinder` e `Box`.

```
from uaibot import Pedestrian, ObstacleColumn
import numpy as np

# Inicializa um pedestre com posição inicial e objetivo
ra = np.array([[0], [0]])
goal = np.array([[5], [5]])
ped = Pedestrian(ra=ra, g=goal)

# Inicializa um obstáculo fixo representando uma coluna
ro = np.array([[2], [2]])
obs = ObstacleColumn(ro=ro)
```

[2] https://github.com/uaibot/uaibot

4.3.2 Cálculo da Força de Atração ao Objetivo

A força de atração que direciona o pedestre ao seu objetivo é calculada pela função `fag()`, com base na diferença entre a velocidade desejada e a atual.

```
f_ag = ped.fag()
```

Essa força é definida por:

$$\vec{f}_{ag} = m_a \cdot \frac{v_a^0 \cdot \vec{e}_a^0 - \vec{v}_a}{\tau_a}$$

onde $\vec{e}_a^0$ é o vetor unitário apontando para o objetivo.

4.3.3 Cálculo da Força de Repulsão com Obstáculos

A função `fdaq(obstacle)` calcula a força de repulsão do pedestre em relação a um obstáculo, levando em conta a distância e o tipo do obstáculo.

```
f_rep = ped.fdaq(obs)
```

A força é modulada pela distância e se decai exponencialmente conforme a fórmula:

$$\vec{f}_{aq} = a \cdot e^{-d/b} \cdot \hat{d}$$

onde os parâmetros a e b variam conforme o tipo do obstáculo (pedestre ou fixo).

Na Figura 4.2 podemos ver o desenho bidimensional da trajetória de um pedestre desviando de um obstáculo do tipo coluna. A trajetória foi calculada usando as funções disponíveis no UAIbot.

4.3.4 Função de Anisotropia Direcional

Para incorporar a sensibilidade direcional do agente, a função `w(obstacle)` aplica um fator multiplicativo à força de repulsão com base na direção relativa do obstáculo.

```
weight = ped.w(obs)
f_total = ped.fdaq(obs) * weight
```

Se o obstáculo estiver fora do campo de visão do pedestre, sua influência é reduzida proporcionalmente. Podemos ver na Figura 4.3 o quanto o fator anisotrópico diminui os efeitos de repulsão dependendo da posição do obstaculo em relação ao pedestre.

4.3.5 Atualização de Velocidade e Posição

O comportamento dinâmico do pedestre é obtido através de uma integração explícita das forças resultantes:

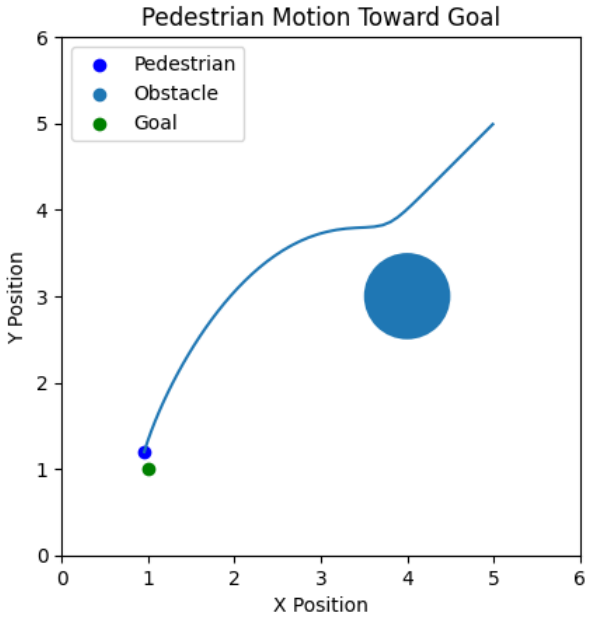

Figura 4.2: Trajetória gerada por SFM mostrando desvio de obstáculo circular simulado com UAIbot.

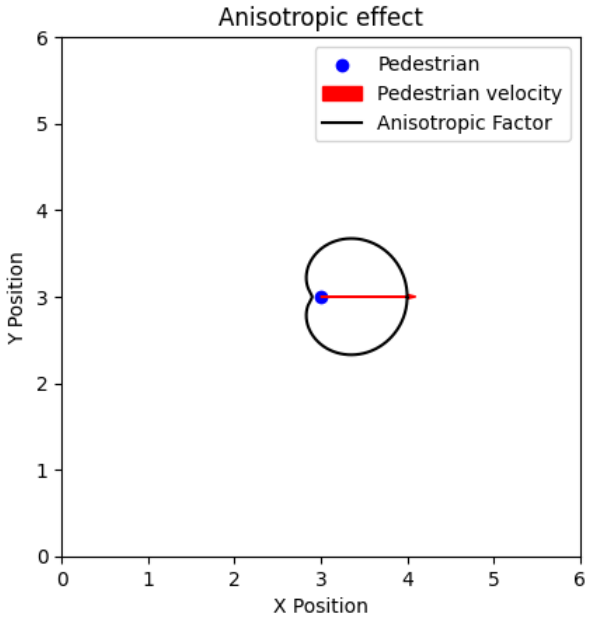

Figura 4.3: Anisotropia

```
dt = 0.1
for step in range(100):
    force_total = ped.fag() + ped.fdaq(obs) * ped.w(obs)
    ped.va = ped.va + force_total * dt
    ped.ra = ped.ra + ped.va * dt
```

Esse loop pode ser estendido para múltiplos pedestres e obstáculos, e permite a visualização da cena dentro do ambiente gráfico do UAIbot.

4.3.6 Cálculo Vetorial de Distância

Todos os obstáculos implementam o método `d()`, responsável por calcular o vetor mínimo entre a superfície do obstáculo e a borda do agente. Isso garante um comportamento realista de colisão evitada.

```
d_vec = obs.d(ped.ra, ped.radius)
```

Esse vetor é essencial para definir a direção da força de repulsão.

4.3.7 Paredes Finas com Distância Geométrica

A classe `ObstacleThinWall` foi especialmente desenvolvida para representar paredes retas com geometria precisa. Ela permite calcular a menor distância entre a parede e o agente, mesmo que a projeção caia fora do segmento. A espessura da parede não é considerada no cálculo, por isso o nome ”parede fina”.

```
start = np.array([[1], [0]])
end = np.array([[1], [5]])
wall = ObstacleThinWall(start=start, end=end)

d_vec = wall.d(ped.ra, ped.radius)
```

Na Figura 4.4 podemos ver o desenho bidimensional da trajetória de um pedestre desviando de um obstáculo do tipo Parede Fina. A trajetória foi calculada usando as funções disponíveis no UAIbot.

4.3.8 Visualização dos Resultados

A trajetória dos agentes pode ser observada no visualizador 3D do UAIbot ou exportada para análise posterior. Isso torna o UAIbot uma ferramenta útil tanto para experimentação quanto para demonstração visual de resultados. As trajetórias citadas até agora nesse capítulo foram feitas em 2D com o *matplotlib*, mas com o UAIbot também é possível salvar animações 3D dos agentes se movendo em meio aos obstáculos, como pode ser visto na simulação apre-

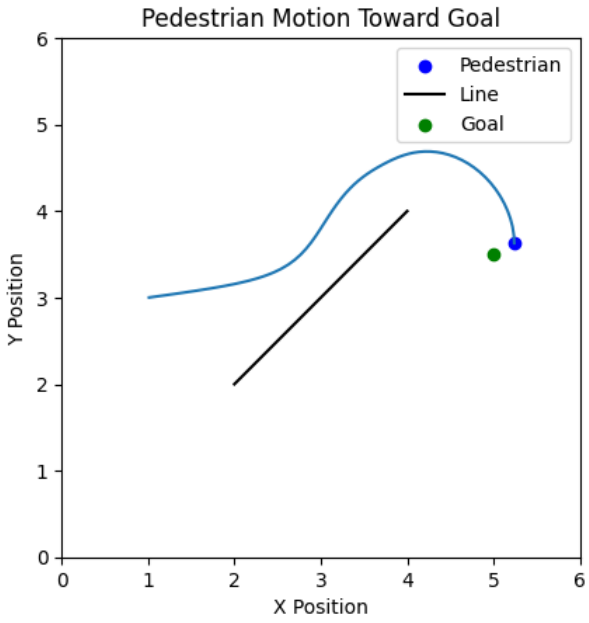

Figura 4.4: Pedestre desviando de uma parede fina

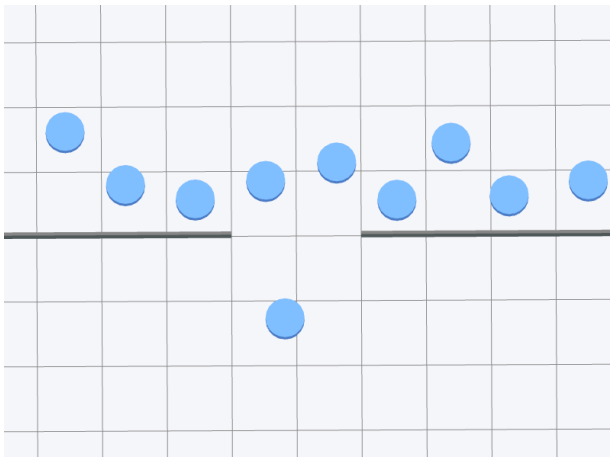

Figura 4.5: Evacuação de uma sala por um grupo de pedestres

sentada na Figura 4.5 e como mostrado na simulação online[3], que simula a evacuação de uma sala por um grupo de pedestres.

Com a implementação das funções acima, o simulador passou a oferecer suporte completo à simulação baseada no Social Force Model, com suporte a múltiplos agentes, obstáculos estáticos e móveis, além de controle fino sobre parâmetros físicos e sociais. Essa base foi utilizada na etapa de identificação de parâmetros descrita no Capítulo 3.

[3]https://uaibot.github.io/assets/simf.html

4.4 Resultados da Otimização dos Parâmetros do Modelo de Forças Sociais

Após a implementação do Social Force Model no simulador UAIbot, foi realizada a etapa de ajuste de parâmetros com base em trajetórias reais extraídas do NavWareSet. A otimização teve como objetivo encontrar valores que minimizassem o erro entre as trajetórias simuladas e as observadas, conforme descrito metodologicamente no Capítulo 3.

4.4.1 Cenas e Dados Utilizados

Foram utilizadas 14 trilhas diferentes de cenas com obstáculos móveis (robôs), extraídas de quatro diferentes cenas (Scene 8, Scene 21, Scene 34 e Scene 47). Cada trilha representa o deslocamento de um participante diante de um robô com comportamento não social. As trilhas foram anotadas manualmente e incluíram dados temporais, poses dos robôs e posições dos pedestres ao longo do tempo. Elas foram escolhidas pois apresentavam um claro desvio da trajetória do pedestre pela presença do robô, ao mesmo tempo que o pedestre se mantinha longe de outros obstáculos que pudessem também desviar sua trajetória.

A otimização foi realizada de forma global, ou seja, os mesmos parâmetros foram ajustados para minimizar o erro somado de todas as trilhas, resultando em um conjunto compartilhado de valores representativos do comportamento médio dos participantes. O código utilizado pode ser visto no apêndice deste trabalho.

4.4.2 Parâmetros Otimizados

Os cinco parâmetros otimizados foram:

- τ — tempo de relaxação;
- v_a^0 — velocidade desejada;
- a_i — amplitude da força repulsiva com obstáculos;
- b_i — fator de decaimento da força repulsiva;
- λ_a — fator de anisotropia direcional.

Os valores finais obtidos pela otimização foram:

Tabela 4.2: Parâmetros otimizados do modelo SFM no UAIbot.

Parâmetro	**Valor Otimizado**
τ	0.2771
v_a^0	1.5578
a_i	2.1541
b_i	5.1526
λ_a	0.4401

A função de custo total resultante da otimização foi de 346.34, representando a soma dos erros euclidianos entre as posições reais e previstas ao longo de todas as trilhas analisadas.

4.4.3 Comparação com a Literatura

Para avaliar a validade dos parâmetros obtidos, realizou-se uma comparação com dois trabalhos relevantes da literatura: Ferrer et al. (2013) e Agrawal et al. (2024). A Tabela 4.3 apresenta os valores publicados nesses trabalhos e os obtidos na presente dissertação. Importante salientar que todos os artigos usados nas comparações usam a mesma versão do SFM implementada aqui neste trabalho (especificação circular). Quanto à forma em que seus parâmetros foram encontrados, isso é pouco discutido em seus respectivos artigos.

Tabela 4.3: Comparação entre parâmetros otimizados e valores da literatura.

Fonte	a_i	b_i	λ_a	τ
UAIbot (este trabalho)	2.15	5.17	0.44	0.28
Ferrer et al. (2013)	2.66	0.79	0.59	0.43
Agrawal et al. (2024)	7.93	0.99	0.40	0.60

4.4.4 Análise dos Resultados

Observa-se que os parâmetros ajustados neste trabalho se encontram dentro de intervalos razoáveis em comparação com a literatura. No entanto, destacam-se algumas diferenças importantes:

- O valor de b_i (decaimento da força) é significativamente maior do que nos trabalhos anteriores, isso pode ser explicado pelo fato das cenas escolhidas terem sido todas feitas com base no cenário *Frontal Approach* onde os pedestres veem o robô com bastante antecedência. Esse fenômeno já foi explorado em outros trabalhos que tentam estender o SFM para levar isso em conta, como Farina et al. (2017).
- O tempo de relaxação τ ficou abaixo dos valores reportados por Ferrer e Agrawal, o que sugere que os participantes ajustaram suas trajetórias de forma mais rápida em resposta ao ambiente dinâmico.
- O fator de anisotropia λ_a foi intermediário entre os trabalhos comparados, apontando para uma sensibilidade direcional moderada à presença do robô.

Essas diferenças reforçam a importância de ajustar os parâmetros do modelo com base em dados reais e específicos ao contexto experimental considerado. O uso do NavWareSet e do UAIbot como ferramentas integradas possibilitou uma simulação ajustada à realidade observada. Na Figura 4.6 podemos ver a comparação entre uma trajetória obtida no NavWareSet e uma calculada com o UAIbot.

A Figura 4.6 apresenta a comparação entre a trajetória real de um pedestre (linha verde) e a trajetória simulada com o modelo de forças sociais (linha azul) implementado no UAIbot,

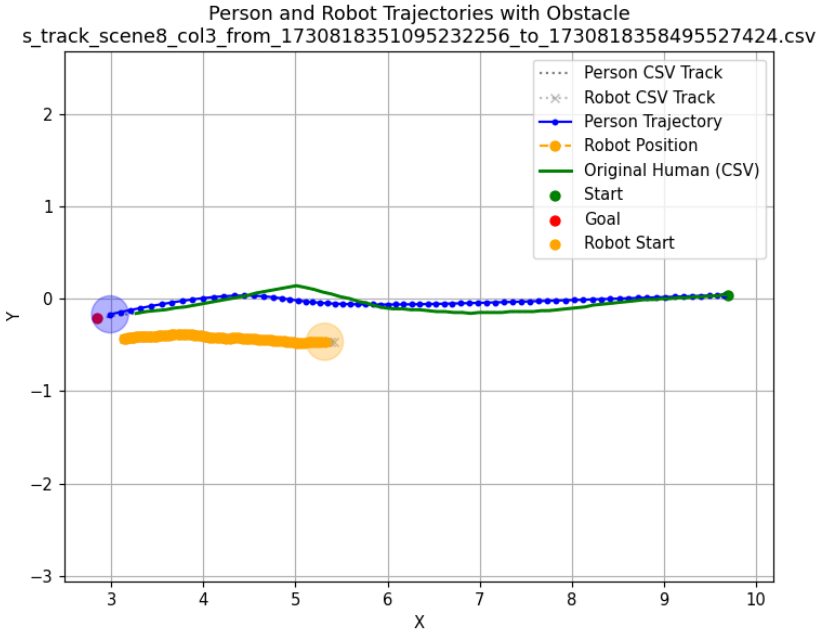

Figura 4.6: Comparação entre uma trajetória obtida no NavWareSet e uma calculada com o UAIbot

para a cena 8 do NavWareSet. O pedestre parte da posição indicada pelo ponto verde e se dirige ao ponto vermelho (seu objetivo final), enquanto o robô parte da posição laranja e percorre a trajetória pontilhada em amarelo. A interação entre o pedestre e o robô, que se aproxima de forma não social, resulta em um desvio suave na trajetória real, visível na curva central da linha verde. O modelo SFM simulado consegue replicar de forma coerente esse comportamento de desvio, apresentando uma trajetória (linha azul) que se aproxima da original tanto na forma quanto no tempo de execução. A diferença entre as trajetórias é resultado das aproximações do modelo, que considera apenas as forças sociais entre o pedestre e o robô. A boa correspondência entre as trajetórias indica que os parâmetros otimizados são adequados para representar o comportamento observado experimentalmente.

Capítulo 5

Conclusão

Esta dissertação apresentou o desenvolvimento do **NavWareSet**, um conjunto de dados inédito voltado para a análise e modelagem de interações sociais entre robôs e seres humanos em ambientes internos. A coleta dos dados foi realizada por meio de experimentos cuidadosamente planejados, abrangendo cenários variados de navegação social e envolvendo múltiplos participantes humanos e dois robôs com perfis distintos: o Toyota HSR e o Clearpath Jackal. Uma característica central do NavWareSet é a existência de versões *conformista* e *não conformista* para quase todos os cenários, permitindo comparar diretamente a influência do respeito às normas sociais sobre as respostas humanas. Em cada sessão, o teleoperador seguiu um roteiro detalhado, decidindo o momento e o lado do desvio em função do conforto do pedestre ou, na versão não conformista, mantendo uma trajetória reta sem considerar as pessoas. Com a utilização de sensores embarcados e de uma estação externa de registro (*Ground-Truth Recording Station*), foi possível capturar dados multimodais sincronizados e de alta qualidade. Por limitações dos algoritmos de detecção automática aplicados às nuvens de pontos 3D, optou-se por anotar manualmente as posições dos pedestres quadro a quadro, garantindo um *ground truth* preciso para calibração de modelos.

O **NavWareSet** foi concebido para apoiar pesquisas em navegação social de robôs, oferecendo cenas realistas, trajetórias humanas anotadas, mapas de ocupação e dados sensoriais sincronizados. Espera-se que o dataset seja utilizado por pesquisadores interessados em treinar algoritmos de navegação autônoma, calibrar modelos de interação social, ou avaliar métricas de convivência humano-robô em ambientes compartilhados. O conjunto de dados já está sendo utilizado por membros do projeto NavWare[1], tanto para fins de análise quanto para testes preliminares de planejadores de movimento baseados em aprendizado.

Além da construção do dataset, foi implementada uma versão funcional e extensível do modelo de forças sociais (Social Force Model — SFM) no simulador **UAIbot**. Esse simulador, desenvolvido em colaboração com o professor Vinicius Mariano e seus alunos, publicado como projeto de código aberto[2], foi expandido para permitir a simulação de pedestres e obstáculos

[1] `https://anr-navware.github.io/`

[2] `https://uaibot.github.io/`

com base em interações sociais. O SFM foi utilizado nesta dissertação como um *estudo de caso* para demonstrar a utilidade do NavWareSet: sua implementação no UAIbot permitiu experimentar diferentes parâmetros e realizar calibrações objetivas, mas não pretende ser a abordagem definitiva para a navegação social. A otimização foi realizada com base em trajetórias reais extraídas do NavWareSet, ajustando os parâmetros para minimizar o erro entre as trajetórias simuladas e observadas. O processo evidenciou a influência dos cenários escolhidos e a sensibilidade aos valores iniciais, reforçando a necessidade de considerar múltiplas sementes e cenas distintas em futuras calibrações.

As principais contribuições deste estudo incluem:

- a criação e disponibilização pública de um conjunto de dados multimodal e anotado, com foco na navegação social de robôs, incluindo versões conformistas e não conformistas de quase todos os cenários, teleoperação documentada e metadados detalhados;
- a demonstração de uma metodologia transparente de coleta, teleoperação e anotação manual, com discussão crítica sobre desafios como o efeito de novidade e a variabilidade das trajetórias;
- a implementação do SFM no UAIbot como estudo de caso, com suporte a obstáculos estáticos e móveis e visualização 3D interativa;
- a aplicação de uma metodologia objetiva para identificação de parâmetros a partir de dados reais e comparação com a literatura;
- a demonstração da utilidade do UAIbot como plataforma leve e programável para experimentação em simulações sociais e calibração de modelos.

Esses resultados fortalecem a base empírica para o estudo da navegação social e oferecem ferramentas úteis para pesquisadores que desejam treinar, avaliar ou comparar algoritmos nesse domínio. O NavWareSet preenche lacunas identificadas na literatura ao disponibilizar dados contrastivos de comportamentos conformistas e não conformistas, associados a teleoperação transparente e anotação precisa. Ao posicionar o SFM como estudo de caso e não como foco principal, abre-se espaço para que outros pesquisadores explorem técnicas baseadas em aprendizado profundo ou modelos híbridos utilizando o mesmo dataset.

Entre as limitações deste trabalho, destacam-se a restrição dos experimentos a ambientes internos e controlados e o número limitado de participantes humanos e cenas disponíveis. Esses fatores podem limitar a generalização dos modelos treinados, já que variáveis externas como iluminação, mobiliário ou diferenças culturais não foram exploradas. O procedimento de teleoperação, embora bem documentado, pode introduzir vieses na escolha do lado de desvio ou na velocidade do robô, e o número reduzido de participantes não captura toda a diversidade demográfica. Por fim, embora o SFM seja amplamente utilizado, ele apresenta limitações conhecidas para representar comportamentos humanos complexos, como hesitação, coordenação grupal ou antecipação de múltiplos agentes simultâneos; sua otimização depende fortemente da escolha das trilhas e das condições iniciais.

Como trabalhos futuros, propõe-se:

- ampliar o NavWareSet com novos cenários (incluindo ambientes externos e layouts arquitetônicos diversos), obstáculos dinâmicos e participantes de diferentes faixas etárias e origens culturais;
- documentar e testar diferentes estratégias de teleoperação, bem como múltiplas sementes iniciais na otimização de modelos como o SFM, avaliando outros cenários além da aproximação frontal;
- comparar o SFM com modelos baseados em aprendizado profundo, como redes neurais generativas de trajetórias, e explorar modelos híbridos que combinem regras físicas e aprendizado de máquina;
- integrar o SFM calibrado com sistemas reais de navegação embarcada, para validação em tempo real com robôs físicos e pesquisa de campo;
- estender a análise crítica para examinar efeitos de aprendizagem a longo prazo, vieses individuais e nuances entre comportamentos conformistas e não conformistas, oferecendo uma base mais rica para a comunidade de navegação social.

Conclui-se, portanto, que o trabalho atinge seus objetivos ao fornecer uma base sólida, reprodutível e extensível para o avanço da navegação social de robôs, combinando coleta realista de dados, modelagem física e validação experimental em um ambiente de simulação aberto e acessível.

Referências Bibliográficas

Agrawal, S., Dengler, N. & Bennewitz, M. (2024). Evaluating robot influence on pedestrian behavior models for crowd simulation and benchmarking.

Agrawal, S., Ostermann-Myrau, N., Dengler, N. & Bennewitz, M. (2026). Peroi: A pedestrian-robot interaction dataset for learning avoidance, neutrality, and attraction behaviors in social navigation.

Brayan, J., Deng, S., Alves Neto, A., Okunevich, I., Krajnik, T., Bremond, F. & Yan, Z. (2025a). NavWareSet: A Dataset of Socially Compliant and Non-Compliant Robot Navigation. working paper or preprint.

Brayan, J., Neto, A. A., Petrovič, P., Freitas, G. M. & Gonçalves, V. M. (2025b). Uaibot: Beginner-friendly web-based simulator for interactive robotics learning and research.

Farina, F., Fontanelli, D., Garulli, A., Giannitrapani, A. & Prattichizzo, D. (2017). Walking ahead: The headed social force model. *PLOS ONE*, 12(1):1–23.

Ferrer, G., Garrell, A. & Sanfeliu, A. (2013). Robot companion: A social-force based approach with human awareness-navigation in crowded environments. pages 1688–1694.

Francis, A., Pérez-D'Arpino, C., Li, C., Xia, F., Alahi, A., Alami, R., Bera, A., Biswas, A., Biswas, J., Chandra, R., Chiang, H.-T. L., Everett, M., Ha, S., Hart, J., How, J. P., Karnan, H., Lee, T.-W. E., Manso, L. J., Mirksy, R., Pirk, S., Singamaneni, P. T., Stone, P., Taylor, A. V., Trautman, P., Tsoi, N., Vázquez, M., Xiao, X., Xu, P., Yokoyama, N., Toshev, A. & Martín-Martín, R. (2023). Principles and guidelines for evaluating social robot navigation algorithms.

Helbing, D., Buzna, L., Johansson, A. & Werner, T. (2005). Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions. *Transportation Science*, 39(1):1–24.

Helbing, D. & Johansson, A. (2013). Pedestrian, crowd, and evacuation dynamics.

Helbing, D. & Molnár, P. (1995). Social force model for pedestrian dynamics. *Phys. Rev. E*, 51:4282–4286.

Johansson, A., Helbing, D. & Shukla, P. (2008). Specification of the social force pedestrian model by evolutionary adjustment to video tracking data.

Karnan, H., Nair, A., Xiao, X., Warnell, G., Pirk, S., Toshev, A., Hart, J., Biswas, J. & Stone, P. (2022). Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation.

Karwowski, J., Szynkiewicz, W. & Niewiadomska-Szynkiewicz, E. (2024). Bridging requirements, planning, and evaluation: A review of social robot navigation. *Sensors*, 24(9).

Korbmacher, R. & Tordeux, A. (2022). Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches.

Kretz, T., Lohmiller, J. & Sukennik, P. (2017). Some indications on how to calibrate the social force model of pedestrian dynamics.

Laufer, J. (2008). Passenger and pedestrian modelling at transport facilities. In *Proceedings of the 2008 Annual AIPTM Conference*. PTV Asia-Pacific.

Lerner, A., Chrysanthou, Y. & Lischinski, D. (2007). Crowds by example. *Computer Graphics Forum*, 26(3):655–664.

Luber, M., Stork, J., Tipaldi, G. D. & Arras, K. (2010). People tracking with human motion predictions from social forces. pages 464–469.

Martin-Martin, R., Rezatofighi, H., Shenoi, A., Patel, M., Gwak, J., Dass, N., Federman, A., Goebel, P. & Savarese, S. (2019). Jrdb: A dataset and benchmark for visual perception for navigation in human environments.

Mavrogiannis, C., Baldini, F., Wang, A., Zhao, D., Trautman, P., Steinfeld, A. & Oh, J. (2021). Core challenges of social robot navigation: A survey.

Mirsky, R., Xiao, X., Hart, J. & Stone, P. (2022). Conflict avoidance in social navigation – a survey.

Pellegrini, S., Ess, A. & Van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. pages 261–268.

Robicquet, A., Sadeghian, A., Alahi, A. & Savarese, S. (2016). Learning social etiquette: Human trajectory understanding in crowded scenes. In *European Conference on Computer Vision*.

Rudenko, A., Kucner, T. P., Swaminathan, C. S., Chadalavada, R. T., Arras, K. O. & Lilienthal, A. J. (2020). ThÖr: Human-robot navigation data collection and accurate motion trajectories dataset. *IEEE Robotics and Automation Letters*, 5(2):676–682.

Saadatnejad, S., Bahari, M., Khorsandi, P., Saneian, M., Moosavi-Dezfooli, S.-M. & Alahi, A. (2022). Are socially-aware trajectory prediction models really socially-aware?

Singamaneni, P. T., Bachiller-Burgos, P., Manso, L. J., Garrell, A., Sanfeliu, A., Spalanzani, A. & Alami, R. (2024). A survey on socially aware robot navigation: Taxonomy and future challenges. *The International Journal of Robotics Research*, 43(10):1533–1572.

Sisbot, E., Marin-Urias, L., Alami, R. & Siméon, T. (2007). A human aware mobile robot motion planner. *Robotics, IEEE Transactions on*, 23:874 – 883.

Yan, Z., Sun, L., Duckett, T. & Bellotto, N. (2018). Multisensor online transfer learning for 3d lidar-based human detection with a mobile robot.

Zanlungo, F., Ikeda, T. & Kanda, T. (2011). Social force model with explicit collision prediction. *Europhysics Letters*, 93(6):68005.

Apêndice A

Código para Otimização dos Parâmetros do Modelo de Forças Sociais

A.1 Código-fonte em Python

```
import pandas as pd
import numpy as np
import glob
from scipy.optimize import minimize

# Import your SFM library here
import uaibot as ub

def residuals(
    dfs,
    ta=0.3,
    va0=1.4,
    a_i=2.5,
    b_i=6,
    lambda_a=0.1,
    ro_radius=0.2,
    dt=0.1,
    t_max=10,
    tolerance=0.3
):
    total_distance = 0.0
    for df in dfs:
        initial_x = df['x'].iloc[0]
        initial_y = df['y'].iloc[0]
```

```
        final_x = df['x'].iloc[-1]
        final_y = df['y'].iloc[-1]
        robot_x = df['robot_x'].iloc[0]
        robot_y = df['robot_y'].iloc[0]

        ro = np.array([robot_x, robot_y])
        ra = np.array([initial_x, initial_y])
        g = np.array([final_x, final_y])

        p = ub.Pedestrian(ra, g, va0=va0, a_i=a_i, b_i=b_i, lambda_a=lambda_a, ta=ta, rad
        obstacle = ub.ObstacleColumn(ro, radius=ro_radius, name="robot", color="red", hei

        positions = []
        t = 0
        k = 1
        while np.linalg.norm(p.ra - p.g) > tolerance and t < t_max:
            fag_result = p.fag()
            fdaq_result = p.fdaq(obstacle) * p.w(obstacle)
            p.va = p.va + (fag_result + fdaq_result) * dt
            p.ra = p.ra + p.va * dt
            try:
                new_robot_x = df['robot_x'].iloc[k]
                new_robot_y = df['robot_y'].iloc[k]
                obstacle.ro = np.array([new_robot_x, new_robot_y])
            except:
                break
            positions.append(p.ra.flatten())
            t += dt
            k += 1

        positions = np.array(positions)
        min_len = min(len(positions), len(df))
        for i in range(min_len):
            csv_pos = (df['x'].iloc[i], df['y'].iloc[i])
            pred_pos = (positions[i, 0], positions[i, 1])
            dist = np.linalg.norm(np.array(csv_pos) - np.array(pred_pos))
            total_distance += dist

    return total_distance

def optimize_parameters(dfs):
```

```
    x0 = [0.8545, 0.6252, 4.3905, 3.1484, 0.0101]
    bounds = [
        (0.1, 2.0),
        (0.5, 2.5),
        (0.1, 10.0),
        (0.1, 10.0),
        (0.01, 2.0)
    ]

    def objective(x):
        ta, va0, a_i, b_i, lambda_a = x
        return residuals(dfs, ta=ta, va0=va0, a_i=a_i, b_i=b_i, lambda_a=lambda_a)

    result = minimize(objective, x0, bounds=bounds, method='L-BFGS-B')
    return result

if __name__ == "__main__":
    csv_files = sorted(glob.glob("s_track_scene*.csv"))
    if not csv_files:
        print("No CSV files found matching 'track_scene*.csv' in the current directory.")
    else:
        dfs = []
        for csv_file in csv_files:
            print(f"Loading: {csv_file}")
            df = pd.read_csv(csv_file)
            if 'timestamp' in df.columns:
                df = df.sort_values('timestamp')
            dfs.append(df)
        print("Optimizing parameters across all tracks...")
        result = optimize_parameters(dfs)
        print("Optimization result (shared across all tracks):")
        print(f"  ta       = {result.x[0]:.4f}")
        print(f"  va0      = {result.x[1]:.4f}")
        print(f"  a_i      = {result.x[2]:.4f}")
        print(f"  b_i      = {result.x[3]:.4f}")
        print(f"  lambda_a = {result.x[4]:.4f}")
        print(f"Sum of all differences between real and predicted values (all tracks): {r
```