INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




Desenvolvimento de um assistente robótico para localização e recuperação de objetos em ambientes domésticos

JOÃO PEDRO VIEIRA COELHO MOREIRA
outubro de 2025

P.PORTO

Desenvolvimento de um assistente robótico para localização e recuperação de objetos em ambientes domésticos

João Pedro Vieira Coelho Moreira

Aluno nº: 1222741

Dissertação para obtenção do Grau de

Mestre em Engenharia de Inteligência Artificial

Orientador: Doutor Diogo Emanuel Pereira Martinho, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Júri:

Presidente:

Doutor António Constantino Lopes Martins, Professor Coordenador do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Vogais:

Doutor Agostinho Gil Teixeira Lopes, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Doutor Diogo Emanuel Pereira Martinho, Professor Adjunto do Instituto Superior de Engenharia do Porto do Instituto Politécnico do Porto

Porto, 30 de setembro 2025

Resumo

O envelhecimento progressivo da população coloca desafios crescentes na promoção da autonomia e na manutenção da qualidade de vida dos idosos. A robótica de assistência tem ganho destaque como resposta a estes desafios, oferecendo soluções capazes de apoiar a realização de tarefas quotidianas e de reduzir a dependência de terceiros em ambientes domésticos. Esta dissertação apresenta o desenvolvimento de um assistente robótico para localização e recuperação de objetos em casa, concebido para apoiar os utilizadores na execução de atividades simples, mas relevantes para a vida diária.

O robô foi projetado para operar em espaços domésticos estruturados, integrando algoritmos de navegação, visão computacional baseada em *deep learning* e mecanismos de interação homem-robô. A solução garante adaptação a pequenas variações no posicionamento dos objetos e na configuração do espaço, assegurando flexibilidade e utilidade prática.

A solução desenvolvida baseou-se numa leitura compreensiva da literatura e do estado da arte, organizada e conduzida segundo a metodologia PRISMA, o que assegurou uma revisão sistemática e a seleção fundamentada das tecnologias mais adequadas. O sistema foi concebido e testado exclusivamente em ambientes reais, permitindo avaliar o seu desempenho em condições práticas e representativas de um contexto de utilização doméstica. Os resultados obtidos evidenciam o potencial da solução para apoiar a autonomia dos utilizadores e reduzir a dependência de terceiros, mas enquadram-se ainda no âmbito de um protótipo experimental. O sistema demonstrou que é possível implementar um assistente robótico funcional em ambiente real mesmo com recursos de hardware limitados, desde que sustentado por uma arquitetura de processamento distribuído e por técnicas adequadas de visão computacional, navegação e manipulação. Esta prova de conceito confirmou a viabilidade da abordagem e estabeleceu uma base sólida para trabalhos futuros, nos quais poderão ser exploradas melhorias ao nível da perceção, da robustez da navegação e da adaptação a cenários domésticos mais complexos.

Palavras-Chave: Assistência robótica, Visão computacional, YOLO, Processamento distribuído, Localização de objetos, Manipulação robótica, Navegação visual.

Abstract

The progressive aging of the population poses increasing challenges in promoting autonomy and maintaining the quality of life of the elderly. Assistive robotics has gained prominence as a response to these challenges, offering solutions capable of supporting the performance of daily tasks and reducing dependence on third parties in domestic environments. This dissertation presents the development of a robotic assistant for object localization and retrieval at home, designed to support users in carrying out simple but meaningful activities for daily living.

The robot was designed to operate in structured domestic spaces, integrating navigation algorithms, computer vision based on deep learning, and human-robot interaction mechanisms. The solution ensures adaptation to small variations in the positioning of objects and in the configuration of the environment, thus providing flexibility and practical usefulness.

The developed solution was based on a comprehensive review of the current literature and the state of the art, organized and conducted according to the PRISMA methodology, which ensured a systematic review and a well-founded selection of suitable technologies. The system was designed and tested exclusively in real environments, allowing the evaluation of its performance under practical and representative domestic conditions. The results highlight the potential of the solution to support user autonomy and reduce dependence on others, although it still falls within the scope of an experimental prototype. The system demonstrated that it is possible to implement a functional robotic assistant in a real environment even with limited hardware resources, provided it is supported by a distributed processing architecture and appropriate techniques for computer vision, navigation, and manipulation. This proof of concept confirmed the feasibility of the approach and established a solid foundation for future work, in which improvements can be explored in perception, navigation robustness, and adaptation to more complex domestic scenarios.

Keywords: Robotic assistance, Computer vision, YOLO, Distributed processing, Object localization, Robotic manipulation, Visual navigation.

Agradecimentos

Agradeço ao Professor Diogo Martinho pela orientação e apoio ao longo deste trabalho.

À minha família, obrigado pela força e confiança.

Aos meus amigos, agradeço pela presença e amizade.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Snippets

Lista de Siglas e Acrónimos

BIC	Bayesian Information Criterion
CNN	*Convolutional Neural Network*
DOF	*Degrees of Freedom*
DQN	*Deep Q-Network*
DRL	*Deep Reinforcement Learning*
EM	*Expectation–Maximization*
GECAD	*Group on intelligent Engineering and Computing for Advanced innovation and Development*
GMM	*Gaussian Mixture Models*
HRL	*Hierarchical Reinforcement Learning*
IA	Inteligência Artificial
ISEP	Instituto Superior de Engenharia do Porto
LIDAR	*Light Detection and Ranging*
PRISMA	*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*
Q	Valor esperado em Q-learning
RGPD	Regulamento Geral sobre a Proteção de Dados
RNN	*Recurrent Neural Network*
ROS	Robot Operating System
SLAM	*Simultaneous Localization and Mapping*
YOLO	*You Only Look Once*

1 Introdução

O envelhecimento populacional tem-se afirmado como um dos grandes desafios sociais da atualidade, colocando em causa a sustentabilidade dos sistemas de apoio e exigindo novas formas de promover a autonomia e a qualidade de vida. A criação de soluções tecnológicas que respondam a estas necessidades tornou-se, por isso, uma prioridade, sobretudo no domínio da robótica assistida, que alia capacidades de perceção, interação e manipulação em ambientes domésticos. É neste enquadramento que se insere o presente trabalho, centrado no desenvolvimento de um assistente robótico para a localização e recuperação de objetos.

Ao longo deste capítulo são apresentados o contexto, o problema em análise, os objetivos definidos, as metodologias de investigação adotadas, bem como as considerações de proteção de dados, segurança e ética, e a estrutura do documento.

1.1 Contexto

O envelhecimento da população é uma tendência clara nos países mais desenvolvidos, incluindo Portugal, onde o aumento da esperança média de vida (em 2025 era de 81,49 anos) e a redução das taxas de natalidade, como a taxa de fecundidade da União Europeia (UE) (em 2023 era de 1,38 filhos por mulher), refletem mudanças demográficas significativas [1], [2]. Em termos europeus, cerca de 21,6% da população tem 65 anos ou mais [3] [4], [5]. Este fenómeno, impulsionado por avanços médicos e transformações socioeconómicas, como o adiamento da maternidade e a maior participação das mulheres no mercado de trabalho, resulta numa sociedade progressivamente mais envelhecida [6]. Embora algumas destas tendências tragam um lado positivo, refletido numa maior longevidade e esperança média de vida, o envelhecimento progressivo da população traz desafios consideráveis, especialmente no que diz respeito à promoção da autonomia e qualidade de vida das pessoas idosas. Estas, que representam uma das maiores faixas etárias da população [5], são também as mais afetadas pela perda gradual de capacidades cognitivas e motoras, desafios que são frequentemente agravados pelo isolamento social. Esta perda manifesta-se, muitas vezes, na dificuldade em realizar tarefas simples do dia a dia, como localizar ou recuperar objetos no ambiente doméstico. Condições como artrites, demência ou Alzheimer são exemplos concretos de causas destas dificuldades, evidenciando a importância de abordar as limitações das pessoas idosas de forma integrada, tendo em conta as suas necessidades no contexto da vida diária [7].

As limitações de mobilidade, presentes em condições como artrites, sequelas de AVC (Acidente vascular cerebral) ou doenças neuromusculares, dificultam o deslocamento necessário para

procurar objetos, enquanto as dificuldades cognitivas, que variam desde esquecimentos associados ao envelhecimento até patologias como a demência ou Alzheimer, estão frequentemente associadas à perda de objetos no ambiente doméstico [8], [9]. Ao mesmo tempo, é importante reconhecer estas fragilidades e procurar formas de estimular as capacidades físicas e cognitivas, em vez de as negligenciar. Por exemplo, ao interagir com a pessoa idosa, o objetivo não deve ser exclusivamente realizar a tarefa por ela, mas sim envolvê-la no processo, promovendo atividades que ajudem a combater a inatividade e contribuam para a manutenção ou até para a recuperação de parte das suas capacidades. Nesse sentido, é importante que o robô seja capaz de aprender padrões de comportamento, pois até ações simples, como entregar as chaves de casa no horário habitual, podem funcionar como incentivo para que o utilizador realize outras atividades, como sair para um passeio, reforçando a sua autonomia e bem-estar.

Estas dificuldades tornam-se ainda mais evidentes em ambientes domésticos, especialmente no caso de pessoas que vivem sozinhas e enfrentam não apenas a falta de apoio físico, mas também o isolamento social. Este isolamento agrava a ausência de estímulos que incentivem a atividade física e cognitiva, reforçando a importância de soluções tecnológicas que não só assistam na realização de tarefas do quotidiano, mas também promovam a interação e incentivem a atividade [10], [11]. Neste contexto, a utilização de sistemas que combinem assistência com estímulos positivos pode ser uma ferramenta essencial para mitigar os efeitos do isolamento e para melhorar a qualidade de vida de pessoas idosas [12].

1.2 Problema

O envelhecimento da população e as dificuldades mencionadas na secção 1.1 evidenciam a necessidade de soluções inovadoras que respondam a estas exigências. Neste contexto, a robótica apresenta-se como uma área particularmente promissora, oferecendo soluções tangíveis e práticas capazes de combinar assistência física com estímulos cognitivos [13], [14]. Uma das principais vantagens da robótica é a sua natureza física e intuitiva, que facilita significativamente a interação, especialmente para pessoas idosas, frequentemente menos familiarizadas com novas tecnologias [15]. Os avanços na robótica assistida permitem criar dispositivos inteligentes que interagem com os utilizadores, adaptam-se às suas necessidades e ambientes, ajudando nas tarefas diárias e promovendo a autonomia [13], [14].

Neste sentido, o problema principal consiste em desenvolver um assistente robótico que responda aos desafios, operando em ambientes domésticos controlados e adaptando-se às necessidades diárias dos utilizadores, particularmente daqueles que enfrentam limitações de mobilidade, dificuldades cognitivas ou vivem em situações de isolamento. O desafio será de criar um sistema robótico que não apenas localize, recolha e entregue objetos de forma eficiente e segura, mas que também atue como um estímulo positivo, integrando os utilizadores no processo e incentivando a sua participação em pequenas atividades que combatam a inatividade. A interação com o robô deverá, assim, funcionar como um meio de promover comportamentos ativos, estimulando o envolvimento cognitivo e físico em vez de um papel

totalmente passivo. Para além de beneficiar diretamente os utilizadores, essa solução tem ainda o potencial de aliviar significativamente a carga sobre cuidadores ou profissionais de apoio, oferecendo uma abordagem prática aos desafios associados ao envelhecimento e à perda de autonomia.

O robô foi concebido para reconhecer objetos específicos através de visão computacional e deslocar-se autonomamente no espaço doméstico controlado, planeando trajetos curtos e otimizados para recolher esses objetos. Quando recebe uma ordem, o robô inicia a procura dos objetos especificados, e ao identificar qualquer objeto válido no campo de visão ajusta a sua posição, utiliza o braço mecânico para o recolher e procede à entrega num ponto definido previamente. O sistema foi igualmente desenhado para garantir uma interação simples e acessível, com alguma capacidade de adaptação a mudanças no ambiente ou no comportamento do utilizador. A longo prazo, espera-se que o robô possa reconhecer padrões de utilização e antecipar necessidades, executando tarefas de forma proativa e contribuindo para a autonomia do utilizador.

1.3 Objetivos

Este trabalho tem como principal objetivo o desenvolvimento de um sistema robótico funcional para a localização, recolha e transporte de objetos em ambientes domésticos. Para isso, será utilizado o Transbot SE, um robô com capacidade de movimento retilíneo (frente e trás) e rotação sobre o seu eixo vertical (eixo Z), baseado no sistema operativo ROS (Robot Operating System), equipado com um braço robótico de 3 graus de liberdade e uma câmara integrada.

O sistema irá recorrer a técnicas de visão computacional para a deteção e localização de objetos, explorando exclusivamente a imagem obtida pela câmara montada no robô. Com base na posição dos objetos na imagem, o robô deverá ser capaz de inferir a direção e a proximidade dos mesmos, permitindo-lhe posicionar-se adequadamente para realizar a recolha.

A integração do braço robótico permitirá a manipulação de pequenos objetos previamente definidos, assegurando a sua recolha e transporte. A manipulação será realizada com base na posição estimada dos objetos na imagem, garantindo uma atuação coordenada entre o sistema de navegação e o braço robótico.

Para além da componente funcional, o trabalho procurará validar a hipótese de que o robô é capaz de aprender e antecipar padrões de comportamento do utilizador, personalizando as suas interações de forma progressiva. Através do registo de dados sobre locais habituais de armazenamento de objetos e horários de maior procura, será avaliado se o sistema consegue prever, com um grau de fiabilidade aceitável, onde e quando os objetos poderão ser encontrados. Esta capacidade de adaptação visa aumentar a eficiência da procura e recolha de objetos, tornando a interação com o robô mais natural e alinhada com os hábitos do utilizador.

Por fim, o desempenho do sistema será avaliado em ambientes reais, com disposição física variável e condições não simuladas. A avaliação terá como objetivo analisar a eficácia das

capacidades de perceção, deslocamento e manipulação do robô, bem como a sua robustez na execução de tarefas simples de forma autónoma.

1.4 Metodologia de Investigação

A estratégia de pesquisa utilizada neste trabalho baseou-se na metodologia PRISMA (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*), reconhecida pela sua abordagem estruturada e rigorosa na realização de revisões sistemáticas. O PRISMA organiza o processo de seleção de artigos em quatro fases principais: Identificação, Seleção, Elegibilidade e Inclusão, assegurando que os critérios definidos sejam aplicados de forma sistemática, transparente e reproduzível. A pesquisa foi realizada em bases de dados científicas como Web of Science, IEEE Xplore, PubMed, Semantic Schoolar e Wiley, utilizando strings de procura desenvolvidas com base na pergunta principal e nas subperguntas de investigação.

Para organizar e gerir as referências, foi utilizada a ferramenta Mendeley, que permitiu uma gestão eficiente de artigos científicos ao longo de todas as etapas do processo. Esta plataforma facilitou o armazenamento e a rastreabilidade das fontes selecionadas, assegurando que todas as informações relevantes estivessem devidamente organizadas para o desenvolvimento do estado da arte e a fundamentação teórica do trabalho.

1.5 Proteção de Dados, Segurança e Ética

Esta secção aborda as considerações relacionadas com a proteção de dados, segurança e ética associadas ao desenvolvimento e implementação do assistente robótico, assegurando a conformidade com as diretrizes institucionais e regulamentares aplicáveis.

1.5.1 Respeito pela Integridade Académica

Foram rigorosamente cumpridas as diretrizes estabelecidas pelo P.PORTO, incluindo:

- **Artigo 6, ponto 2.8:** Garantiu-se a originalidade do trabalho através da citação rigorosa de ideias, textos ou conteúdos de terceiros.
- **Ponto 2.9:** Não foram apresentados trabalhos previamente publicados como originais sem o devido reconhecimento.
- **Ponto 2.11:** Os resultados refletem de forma fiel as análises realizadas, evitando falsificações ou manipulações de dados.

Adicionalmente, a pesquisa seguiu as práticas recomendadas no **Artigo 10 do Código de Boas Práticas do P.PORTO**, como a documentação meticulosa do processo e a citação adequada das fontes utilizadas.

1.5.2 Considerações Éticas na Interação com Idosos

A interação entre o assistente robótico e os idosos levanta importantes considerações éticas, garantindo que a tecnologia respeita a dignidade e autonomia dos utilizadores. O robô não deve substituir os cuidadores humanos, mas sim atuar como um apoio adicional, promovendo a independência e qualidade de vida dos idosos.

Neste contexto, o robô é projetado para envolver ativamente os utilizadores nas suas atividades diárias, estimulando-os tanto fisicamente como cognitivamente. A assistência proporcionada não deve resultar na passividade dos utilizadores, mas sim incentivá-los a participar ativamente nas tarefas, promovendo a manutenção das suas capacidades funcionais e psicológicas.

1.5.3 Conformidade com o RGPD

O desenvolvimento do assistente robótico deverá estar em conformidade com o RGPD (Regulamento Geral sobre a Proteção de Dados), assegurando a proteção da privacidade dos utilizadores. Não será efetuada a gravação de vídeo para garantir a privacidade dos utilizadores e minimizar riscos relacionados com a exposição de dados sensíveis. Consequentemente, serão implementados mecanismos de segurança que devem assegurar a eliminação de quaisquer dados temporários, assim que possível, garantindo a proteção da privacidade dos utilizadores e a conformidade com as normas de segurança de dados.

Os únicos dados que deverão ser recolhidos, a longo prazo, incluem as preferências do utilizador (como locais habituais de armazenamento de objetos) e hábitos de uso (como horários mais frequentes de procura de itens). A recolha destes dados será transparente e proporcional à finalidade pretendida. Todos os dados serão armazenados de forma local no servidor do sistema, com acesso restrito ao próprio utilizador, que poderá consultá-los ou eliminá-los a qualquer momento.

1.5.4 Conformidade com o *AI act*

O desenvolvimento do assistente robótico deverá também estar em conformidade com o *AI Act*, o regulamento da União Europeia que estabelece regras para o desenvolvimento, implementação e utilização de sistemas de inteligência artificial, garantindo que o robô opera de forma ética, segura e transparente. O *AI Act* classifica os sistemas de IA em quatro níveis de risco; risco inaceitável, risco elevado, risco limitado e risco mínimo ou, alternativamente, nenhum risco [16], [17]. Dependendo do nível, aplicam-se diferentes obrigações legais. Segundo as diretrizes da União Europeia, o robô doméstico em desenvolvimento seria

provavelmente classificado como um sistema de inteligência artificial de alto risco. A sua capacidade de interação direta com utilizadores e o potencial de afetar a segurança física justificam esta análise. Essa classificação implica a adoção de obrigações rigorosas, tais como a garantia de transparência, a necessidade de supervisão humana e a elaboração de documentação técnica detalhada.

A conformidade com o regulamento também implicará a adoção de mecanismos de segurança e monitorização contínua do desempenho e risco do sistema, garantindo que qualquer comportamento inesperado possa ser identificado e corrigido. Dessa forma, a implementação do assistente robótico não apenas respeitará as diretrizes do *AI Act*, mas também reforçará a confiança dos utilizadores no seu funcionamento, promovendo uma adoção segura e responsável da tecnologia.

1.5.5 Impacto social

A introdução de assistentes robóticos em ambientes domésticos apresenta desafios e oportunidades em termos sociais e de sustentabilidade. Do ponto de vista social, é crucial garantir que estas tecnologias complementem as interações humanas, promovendo a inclusão digital e um envelhecimento ativo. A implementação deve ser cuidadosamente planeada para evitar o isolamento social dos idosos, encorajando a sua participação nas atividades do dia a dia e fortalecendo os laços com familiares e cuidadores. É fundamental reforçar que o robô não deve ser utilizado como um substituto para os cuidadores humanos, mas sim como um recurso complementar para aliviar a carga de trabalho e permitir que estes possam focar-se em tarefas mais complexas e de maior valor humano.

1.5.6 Gestão de conflitos e Interesses

A gestão de conflitos e interesses durante a realização deste trabalho foi conduzida de forma transparente e objetiva. Foram promovidas discussões abertas e regulares com os orientadores do projeto, garantindo que todas as decisões fossem tomadas com base em critérios técnicos e éticos.

O desenvolvimento do projeto seguiu uma abordagem imparcial, procurando analisar os dados de forma objetiva e assegurar que quaisquer decisões fossem fundamentadas em evidência científica, respeitando os padrões académicos e profissionais estabelecidos.

1.6 Estrutura do documento

Este documento está organizado da seguinte forma:

- **Introdução**: Apresenta o contexto do estudo, o problema identificado, os objetivos do trabalho a proteção de dados, segurança e ética e a metodologia de investigação adotada.

- **Estado da Arte**: Faz uma pequena introdução aos conceitos teóricos base e define a pergunta principal e as subperguntas de investigação, detalha o processo de pesquisa através da metodologia PRISMA, realiza uma revisão sistemática da literatura e conclui com a resposta às subperguntas, fornecendo a base teórica e tecnológica necessária para o desenvolvimento do trabalho.

- **Métodos e ferramentas**: Descreve a arquitetura geral do sistema, a divisão de processamento entre o robô e o servidor e os protocolos de comunicação utilizados. Apresenta ainda as tecnologias de navegação, visão computacional e modelos preditivos aplicados, assim como as características do robô físico Transbot SE e a forma como os diferentes componentes foram integrados.

- **Experimentação**: Detalha o plano de testes concebido para avaliar o sistema desenvolvido. Define os objetos utilizados, o espaço físico dos ensaios e as condições experimentais. Apresenta os diferentes testes planeados, incluindo a avaliação da deteção de objetos, da função de preensão, do desvio após trajetórias completas e do modelo preditivo, bem como as ameaças à validade e limitações.

- **Resultados e Discussão**: Apresenta os resultados obtidos em cada um dos testes definidos no plano experimental, organizados de acordo com os objetivos estabelecidos. Inclui métricas de desempenho do modelo YOLOv5, taxas de sucesso de preensão, análise do desvio acumulado após trajetórias de procura e avaliação dos módulos preditivos. Discute ainda de forma crítica os resultados, relacionando-os com as limitações observadas e sugerindo melhorias possíveis.

- **Conclusão**: Sintetiza os resultados alcançados, salientando as principais contribuições do trabalho. Destaca os pontos fortes e as limitações identificadas e apresenta propostas para passos futuros, nomeadamente melhorias ao nível dos sensores, algoritmos e robustez experimental.

2 Estado da arte

Esta secção aborda o estado da arte, explorando os avanços científicos e tecnológicos relevantes para o desenvolvimento do assistente robótico. O objetivo é analisar abordagens e metodologias recorrentes na literatura, destacando tendências e desafios no domínio da robótica assistida. Embora algumas destas tecnologias não tenham sido diretamente aplicadas no sistema desenvolvido, a sua análise permite compreender o enquadramento geral da área.

Será também realizada uma revisão sistemática baseada na metodologia PRISMA, com o objetivo de identificar e discutir os trabalhos mais relevantes para a deteção e recolha de objetos em ambientes interiores estruturados, considerando especialmente soluções viáveis para robôs com recursos computacionais limitados.

2.1 Conceitos Base

Esta secção apresenta os principais fundamentos teóricos que sustentam ou contextualizam o desenvolvimento de assistentes robóticos. São abordados tópicos essenciais como Inteligência Artificial, navegação autónoma e visão computacional.

2.1.1 Inteligência Artificial

A IA é uma área multidisciplinar que visa desenvolver sistemas capazes de realizar tarefas que tradicionalmente requerem inteligência humana, como perceção, tomada de decisão e aprendizagem [18]. As abordagens mais comuns incluem algoritmos baseados em lógica simbólica, técnicas estatísticas e modelos de *machine learning*. A IA tem demonstrado um crescimento exponencial, impulsionado por avanços na capacidade computacional, grandes volumes de dados e novos algoritmos de *deep learning [19], [20]*.

A aplicação de IA abrange diversas áreas, como assistência médica, condução autónoma e, mais recentemente, assistência robótica para apoio a idosos [21]. No contexto da robótica, a IA permite dotar os robôs de capacidades avançadas, como perceção do ambiente e tomada de decisão autónoma [21].

Inteligência Artificial na Robótica

A robótica assistida por IA tem-se tornado uma das soluções mais promissoras para a autonomia dos idosos e pessoas com necessidades especiais [21]. A capacidade de integrar sensores, processar informação em tempo real e adaptar-se ao ambiente torna possível construir robôs funcionais em contextos domésticos.

No âmbito dos assistentes robóticos, a IA é aplicada em três domínios principais: navegação autónoma, reconhecimento de objetos e interação homem-robô. Essas capacidades são

possibilitadas por técnicas como *Simultaneous Localization and Mapping* (*SLAM*), redes neuronais convolucionais (*CNNs*) e *reinforcement learning*, garantindo que os robôs possam operar com precisão e eficiência em ambientes domésticos. O *reinforcement learning* permite que os robôs aprendam a partir da interação com o ambiente, otimizando suas ações com base em recompensas [22], [23].

Nota-se, no entanto, que apesar da relevância destas abordagens na literatura, o sistema aqui desenvolvido pode não recorrer diretamente a todas as tecnologias descritas. A sua inclusão nesta revisão visa, ainda assim, oferecer uma visão abrangente das possibilidades existentes e enquadrar o projeto no contexto mais alargado da robótica assistida.

2.1.2 Navegação Robótica

2.1.2.1 Simultaneous Localization and Mapping (SLAM)

O problema do SLAM consiste na necessidade de um robô construir um mapa do ambiente ao mesmo tempo que determina a sua própria localização dentro desse mapa. Esta capacidade é essencial para a navegação autónoma, especialmente em ambientes desconhecidos ou dinâmicos, onde a informação prévia do espaço não está disponível ou pode sofrer alterações ao longo do tempo [24].

A formulação matemática do SLAM (1) pode ser representada através da distribuição de probabilidade conjunta sobre a trajetória do robô e o mapa, dada uma sequência de observações e comandos de controlo:

$$\boldsymbol{p}(\boldsymbol{x_t}, \boldsymbol{m} \mid \boldsymbol{z_{1:t}}, \boldsymbol{u_{1:t}}) \quad (1)$$

- $\boldsymbol{x_t}$ representa o estado do robô no instante, incluindo sua posição, orientação e, possivelmente, velocidade.
- $\boldsymbol{m}$ é o mapa do ambiente em construção, podendo ser uma nuvem de pontos ou uma malha tridimensional.
- $\boldsymbol{z_{1:t}}$ são as observações captadas pelos sensores do robô até o instante, como imagens de câmara ou leituras de sensores *LIDAR*.
- $\boldsymbol{u_{1:t}}$ são os comandos de controlo aplicados ao robô ao longo do tempo, como deslocamentos e rotações.

O termo (1) representa a incerteza sobre a localização do robô e a configuração do ambiente, refletindo o impacto das medições sensoriais e da evolução dinâmica do sistema. Resolver o problema do SLAM envolve encontrar uma estimativa ótima para maximizar essa probabilidade, tendo em conta os erros de medição e a incerteza associada aos modelos de movimento e perceção.

2.1.2.2 Kalman filters

Os *Kalman filters* são um algoritmo recursivo utilizado para estimar o estado de um sistema dinâmico a partir de medições ruidosas e incertezas em modelos de estado. Os filtros podem ser aplicados em várias áreas, incluindo navegação robótica, rastreamento de objetos e sistemas de controlo [25]. Baseiam-se na combinação entre um modelo de previsão e atualizações a partir de medições ruidosas [26].

Os *Kalman filters* assumem que um sistema dinâmico pode ser modelado por uma equação de estado linear (2) e uma equação de observação:

$$\boldsymbol{x_k} = \boldsymbol{A}\, \boldsymbol{x}_{\{\boldsymbol{k-1}\}} + \boldsymbol{B}\, \boldsymbol{u_k} + \boldsymbol{w_k} \quad (2)$$

- $\boldsymbol{x_k}$ representa o estado do sistema no instante $\boldsymbol{k}$.
- $\boldsymbol{A}$ é a matriz de transição de estado, que descreve como o estado evolui no tempo.
- $\boldsymbol{u_k}$ é o vetor de entrada de controlo, com a matriz $\boldsymbol{B}$ representando sua influência no sistema.
- $\boldsymbol{w_k}$ é o ruído do processo, assumido como uma variável aleatória gaussiana com média zero e covariância $\boldsymbol{Q}$:

A equação (3), indica que o ruído do processo $\boldsymbol{w_k}$ segue uma distribuição Gaussiana com média zero e covariância $\boldsymbol{Q}$, representando a incerteza do modelo de estado, onde $\boldsymbol{Q}$ define a variabilidade e correlação dos erros, influenciando a confiança na previsão do *Kalman filter [25], [26]*.

$$\boldsymbol{w_k} \sim \boldsymbol{\mathcal{N}}(\boldsymbol{0}, \boldsymbol{Q}) \quad (3)$$

Os *Kalman filters* operam recursivamente em dois passos fundamentais:

1. Previsão (*Prediction Step*)
    - Atualiza a estimativa do estado com base no modelo dinâmico.
    - Propaga a incerteza associada à estimativa.
2. Atualização (*Correction Step*)
    - Ajusta a estimativa com base nas medições disponíveis.
    - Reduz a incerteza ao incorporar novas informações.

O primeiro passo, de previsão, é definido pelas equações (4) e (5):

$$\hat{\boldsymbol{x}}_{\mathrm{k}}^{-} = \boldsymbol{A}\, \hat{\boldsymbol{x}}_{\mathrm{k-1}} + \boldsymbol{B}\, \boldsymbol{u}_{\mathrm{k}} \quad (4)$$

$$\boldsymbol{P}_{\mathrm{k}}^{-} = \boldsymbol{A}\boldsymbol{P}_{\mathrm{k}}^{-1}\boldsymbol{A}^{\mathrm{T}} + \boldsymbol{Q} \quad (5)$$

- $\hat{\boldsymbol{x}}_{\mathrm{k}}^{-}$ é a estimativa a priori do estado antes da incorporação da medição.
- $\boldsymbol{P}_{\mathrm{k}}^{-}$ é a matriz de covariância de erro a priori, que quantifica a incerteza da previsão.

Quanto ao segundo passo, da atualização:

1. Cálculo do ganho de *Kalman* (6)*:*

$$\boldsymbol{K}_{\mathrm{k}} = \boldsymbol{P}_{\mathrm{k}}^{-} \boldsymbol{H}^{\mathrm{T}} (\boldsymbol{H}\, \boldsymbol{P}_{\mathrm{k}}^{-} \boldsymbol{H}^{\mathrm{T}} + \boldsymbol{R})^{-1} \quad (6)$$

- $\boldsymbol{P}_{\mathrm{k}}^{-}$ é a covariância do estado antes da atualização (representa a incerteza da previsão).
- $\boldsymbol{H}$ é a matriz de observação, que converte o estado para o espaço das medições.
- $\boldsymbol{R}$ é a covariância do ruído da medição, indicando o nível de incerteza associado às observações do sensor.
- $(\boldsymbol{H}\, \boldsymbol{P}_{\mathrm{k}}^{-} \boldsymbol{H}^{\mathrm{T}} + \boldsymbol{R})$ representa a incerteza combinada da previsão e da medição.

O Ganho de *Kalman* $\boldsymbol{K}_{\mathrm{k}}$ determina o peso que a nova medição terá na correção da estimativa do estado. Se a medição for muito confiável (baixa covariância R), o ganho será maior, e a estimativa será mais influenciada pela medição. Se a medição for pouco confiável (alto R), o modelo de previsão terá maior peso [26].

2. Correção do estado:

Depois de calcular o ganho $\boldsymbol{K}_{\mathrm{k}}$, a estimativa do estado x_{k} é corrigida utilizando a medição z_{k}, conforme a equação (7):

$$\hat{\boldsymbol{x}}_{\mathrm{k}} = \hat{\boldsymbol{x}}_{\mathrm{k}}^{-} + \boldsymbol{K}_{\mathrm{k}} (\boldsymbol{z}_{\mathrm{k}} - \boldsymbol{H}\, \hat{\boldsymbol{x}}_{\mathrm{k}}^{-}) \quad (7)$$

- $\hat{\boldsymbol{x}}_{\mathrm{k}}^{-}$ é a estimativa *a priori* antes da atualização com a medição.
- $\boldsymbol{z}_{\mathrm{k}}$ é a nova medição recebida.
- $\boldsymbol{H}\, \hat{\boldsymbol{x}}_{\mathrm{k}}^{-}$ é a previsão do modelo de como a medição deveria ser.
- $(\boldsymbol{z}_{\mathrm{k}} - \boldsymbol{H}\, \hat{\boldsymbol{x}}_{\mathrm{k}}^{-})$ é chamado de inovação, ou resíduo da medição, que mede a discrepância entre a previsão e a medição real.
- $\boldsymbol{K}_{\mathrm{k}} (\boldsymbol{z}_{\mathrm{k}} - \boldsymbol{H}\, \hat{\boldsymbol{x}}_{\mathrm{k}}^{-})$ ajusta o estado para reduzir essa discrepância.

3. Atualização da Covariância de Erro é dada pela equação (8):

$$\boldsymbol{P}_{\mathrm{k}} = (\boldsymbol{I} - \boldsymbol{K}_{\mathrm{k}}\, \boldsymbol{H})\, \boldsymbol{P}_{\mathrm{k}}^{-} \quad (8)$$

Se $\boldsymbol{K}_{\mathrm{k}}$ for alto, o novo $\boldsymbol{P}_{\mathrm{k}}$ será significativamente menor, pois a medição reduziu a incerteza. Se $\boldsymbol{K}_{\mathrm{k}}$ for baixo, a incerteza permanecerá próxima da anterior.

*2.1.2.3 Algoritmo A**

O algoritmo A* é um dos métodos mais utilizados para *path planning* em robótica e navegação autónoma. Trata-se de um algoritmo de busca heurística que combina os princípios da busca em largura (*Dijkstra*) com uma heurística que orienta a exploração do espaço de busca, tornando-o mais eficiente [27].

O objetivo do algoritmo A* é encontrar o caminho mais curto de um nó inicial $\boldsymbol{X}_{\mathbf{start}}$ até um nó final $\boldsymbol{X}_{\mathbf{end}}$, minimizando a função de custo (9):

$$\boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{g}(\boldsymbol{X}) + \boldsymbol{h}(\boldsymbol{X}) \quad (9)$$

onde:

- $\boldsymbol{g}(\boldsymbol{X})$ representa o custo do caminho desde o nó inicial até X.
- **h(X)** é uma função heurística que estima o custo do nó X até o destino.

O algoritmo mantém duas listas: a *OPEN list*, que contém os nós candidatos a serem explorados, ordenados pelo custo $f(X)$, e a *CLOSED list*, que armazena os nós já expandidos. A heurística $h(X)$ deve ser admissível, ou seja, nunca deve superestimar o custo real até o destino, garantindo assim a obtenção de um caminho ótimo.

O algoritmo segue os seguintes passos:

1. **Inicialização**: O nó inicial $\boldsymbol{X}_{\mathbf{start}}$ é adicionado à *OPEN list* com $g(X_{\text{start}}) = 0$ e $f(X_{\text{start}}) = h(X_{\text{start}})$.

2. **Expansão do nó atual:** O nó X_n com o menor $f(X)$ é retirado da *OPEN list* e colocado na *CLOSED list*. Se $X_n = \boldsymbol{X}_{\mathbf{end}}$, o caminho foi encontrado.

3. **Atualização dos nós vizinhos:** Para cada vizinho X_{i} de X_n, calcula-se o custo atualizado, conforme a equação (10):

$$\boldsymbol{g}(\boldsymbol{X_i}) = \boldsymbol{g}(\boldsymbol{X_n}) + \boldsymbol{d}(\boldsymbol{X_n}, \boldsymbol{X_i}) \quad (10)$$

   onde $d(X_n, X_i)$ é a distância entre os nós. Se X_i não está na *OPEN list* ou tem um custo menor do que o anteriormente calculado, o nó é atualizado e seu predecessor é definido como X_n. O nó X_i é então adicionado à *OPEN list*.

4. **Repetição:** O processo continua até que $\boldsymbol{X}_{\mathbf{end}}$ seja encontrado ou a *OPEN list* esteja vazia (sem solução possível).

A Figura 1 ilustra o funcionamento do algoritmo A*, mostrando a forma como o processo de busca explora o espaço até encontrar o caminho ótimo entre os nós inicial e final. Os nós pesquisados são representados em diferentes cores conforme o seu estado nas listas OPEN e CLOSED, enquanto a linha amarela marca o trajeto final gerado. A imagem permite visualizar o comportamento heurístico do algoritmo, que expande os nós de forma orientada pelo custo total estimado, equilibrando a distância percorrida e a projeção até ao destino.

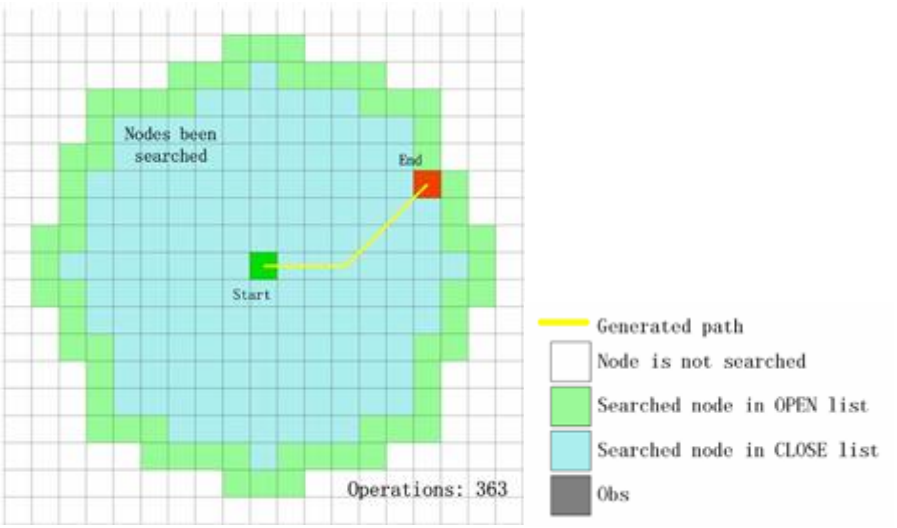

Figura 1 – Demonstração da execução do algoritmo A* [27].

2.1.2.4 Gaussian Mixture Models (GMM)

Os *Gaussian Mixture Models* (GMM) são modelos probabilísticos que representam a distribuição de dados como uma combinação (mista) de várias distribuições normais multivariadas. São úteis para descobrir padrões latentes e agrupar observações quando se assume que os dados provêm de múltiplas "sub-populações" (componentes) com média e covariância próprias.

Seja $x \in \mathbb{R}^d$ um vetor de características. Um GMM com K componentes é definido pela equação (11):

$$p(x) = \sum_{k=1}^{K} \pi_k \; \mathcal{N}(x \mid \mu_k, \Sigma_k) \qquad (11)$$

onde:

- $\pi_k \geq 0 \; e \; \Sigma_k \, \pi_k = 1$ são os pesos (importância) de cada componente;
- $\mu_k \in \mathbb{R}^d$ é a média do componente k;
- $\Sigma_k \in \mathbb{R}^{dxd}$ é a matriz de covariâncias do componente k.

A aprendizagem dos parâmetros $\{\pi_k, \mu_k, \Sigma_k\}$ maximiza a *log-likelihood* dos dados $\{x_n\}_{\{n=1\}}^{N}$. Na prática, utiliza-se o algoritmo *Expectation–Maximization* (EM).

As responsabilidades são determinadas pela equação (12):

$$\gamma_{nk} = \frac{\left(\pi_k \cdot \mathcal{N}(x_n | \mu_k, \Sigma_k)\right)}{\left(\Sigma_j \pi_j \cdot \mathcal{N}(x_n | \mu_j, \Sigma_j)\right)} \quad (12)$$

interpreta a probabilidade de a amostra x_n pertencer ao componente k.

M-step (atualização de parâmetros), equação (13):

$$N_k = \sum_{n=1}^{N} \gamma_{nk}\,,\ \mu_k = \frac{1}{N_k}\sum_{n} \gamma_{nk} x_n\,,\ \Sigma_k = \frac{1}{N_k}\sum_{n} \gamma_{nk}(x_n - \mu_k)(x_n - \mu_k)^{\top} + \lambda I,\ \pi_k = \frac{N_k}{N} \quad (13)$$

Em cima inclui-se uma regularização λI (pequena) para estabilidade numérica quando há poucos dados ou clusters muito compactos.

2.1.2.5 Predictive planning

Predictive Learning consiste na capacidade de um sistema antecipar estados futuros com base em padrões históricos e observações em tempo real. Esta abordagem é essencial para a navegação autónoma em ambientes dinâmicos, permitindo ao robô prever a movimentação de obstáculos móveis, como humanos, e ajustar a sua trajetória para evitar colisões. A capacidade de previsão melhora a eficiência do planeamento de trajetórias e a interação segura em espaços partilhados [28].

A formulação matemática do *predictive Learning* pode ser expressa através da modelação da distribuição de probabilidade sobre os estados futuros do ambiente e a trajetória do robô, dada uma sequência de observações e comandos de controlo (14):

$$\boldsymbol{p}(\boldsymbol{x_{t+1}} \mid \boldsymbol{x_t}, \boldsymbol{u_t}, \boldsymbol{z_t}) \quad (14)$$

onde:

- x_t representa o estado do robô no instante t, incluindo posição, orientação e velocidade.
- u_t são os comandos de controlo aplicados ao robô ao longo do tempo, como deslocamentos e rotações.
- z_t são as observações captadas pelos sensores do robô, como leituras de *LIDAR* ou imagens de câmara.

A estimativa da distribuição (14) reflete a incerteza sobre a evolução do sistema, considerando as medições sensoriais e os modelos de transição de estados. A escolha do modelo de previsão a utilizar depende das condições do ambiente e das capacidades computacionais do robô.

2.1.2.6 Visual Odometry

O *Visual odometry* é o processo de estimar o movimento de um robô ou câmara através da análise de uma sequência de imagens captadas do ambiente. Esta abordagem é fundamental para a navegação autónoma em ambientes onde outras formas de odometria, como as baseadas em rodas, são insuficientes devido a deslizamento ou terrenos irregulares [29].

A formulação matemática do *Visual odometry* pode ser expressa pela estimativa da trajetória do robô com base em observações visuais, sendo definida pela distribuição de probabilidade apresentada na equação (15):

$$\boldsymbol{p(x_t \mid x_t - 1, z_t)} \quad (15)$$

onde:

- x_t representa o estado do robô no instante t, incluindo posição, orientação e velocidade.
- $x_t - 1$ é o estado do robô no instante anterior.
- z_t são as observações visuais captadas pela câmara, utilizadas para inferir mudanças na posição do robô.

A escolha do método de *Visual odometry* depende do tipo de sensores disponíveis e das condições do ambiente. Abordagens baseadas em redes neuronais profundas, como *CNNs*, podem ser utilizadas para melhorar a robustez da correspondência de características visuais [29].

2.1.2.7 Deep Reinforcement Learning (DRL)

O *Deep Reinforcement Learning* (*DRL*) combina técnicas de *reinforcement learning* com redes neuronais profundas para permitir que um agente aprenda a tomar decisões em ambientes complexos e de alta dimensionalidade [30]. Esta abordagem tem sido amplamente utilizada para navegação autónoma, manipulação robótica e jogos.

A formulação matemática do *DRL* (16) baseia-se na maximização da recompensa acumulada a longo prazo através da política ótima π^* onde o problema é modelado como um *Markov Decision Process* (*MDP*):

$$\boldsymbol{V^*(s) = \max_a (R(s,a) + \gamma \sum_{s'} T(s'|s,a) V^*(s'))} \quad (16)$$

onde:

- $V^*(s)$ é a função de valor ótima associada ao estado s.
- $R(s,a)$ é a recompensa obtida ao executar a ação a no estado s.
- $T(s'|s,a)$ representa a probabilidade de transição entre estados.
- γ é o fator de desconto que pondera recompensas futuras.

A política ótima pode ser definida por (17):

$$\boldsymbol{\pi^*(s) = \arg\max_{a} Q^*(s,a)} \quad (17)$$

Onde $Q^*(s,a)$ é a função de valor da ação que orienta a escolha das melhores decisões.

O *DRL* permite que agentes aprendam diretamente a partir de observações brutas, utilizando redes neuronais para aproximar $Q^*(s,a)$ e lidar com espaços de estados contínuos e de alta dimensão [30].

2.1.2.8 Deep Q-Network (DQN)

O *Deep Q-Network* (*DQN*) é um algoritmo de *DRL* que combina *Q-Learning* com redes neuronais profundas para aproximar a função de valor $Q(s,a)$. Ele foi concebido para lidar com problemas de alta dimensionalidade sem necessidade de um modelo explícito do ambiente [30].

A função (18) é aprendida iterativamente através da atualização:

$$\boldsymbol{Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)} \quad (18)$$

onde:

- r é a recompensa recebida após executar a ação a no estado s.
- s' é o estado resultante após a ação a.
- α é a taxa de aprendizagem.
- γ é o fator de desconto.

Para estabilizar a aprendizagem, o *DQN* introduz duas técnicas principais:

1. **Replay de experiência**: O agente armazena experiências passadas (s,a,r,s') numa memória e reutiliza-as para minimizar correlações nas atualizações.
2. **Rede alvo separada**: Uma segunda rede $Q_{target}(s,a)$ é utilizada para calcular o valor alvo, reduzindo instabilidades durante o treino.

2.1.3 Visão Computacional para Identificação e Localização de Objetos

2.1.3.1 Convolutional Neural Networks (CNNs)

As CNNs são amplamente utilizadas na visão computacional para identificar e localizar objetos em ambientes domésticos. Estas redes neuronais destacam-se pela sua capacidade de extrair automaticamente características relevantes das imagens, permitindo a deteção precisa de objetos sem necessidade de engenharia manual de características [31].

A operação principal de uma *CNN* é a convolução, representada matematicamente pela equação (19):

$$\boldsymbol{f(x,y) = \sum_i \sum_j I(x-i, y-j) \cdot K(i,j)} \quad (19)$$

onde:

- $I(x,y)$ representa a imagem de entrada.
- $K(i,j)$ é o kernel de convolução, que aprende padrões visuais como bordas e texturas.
- $f(x,y)$ é o mapa de características extraído da imagem.

Além da deteção de objetos, as *CNNs* são amplamente utilizadas na segmentação semântica (*semantic segmentation*), onde cada píxel da imagem é classificado de acordo com a sua categoria. Esta técnica é essencial para permitir que o robô compreenda não apenas a presença de objetos, mas também a sua relação espacial com o ambiente, facilitando a navegação e a manipulação de objetos em espaços dinâmicos [31].

A Figura 2 ilustra a estrutura de uma CNN, destacando as principais camadas envolvidas no processamento de dados. A secção (a) mostra um fluxo típico de uma *CNN*, onde os dados de entrada passam por múltiplas camadas de convolução, responsáveis pela extração de características, seguidas por camadas de *pooling*, que reduzem a dimensionalidade dos mapas de características, e, finalmente, camadas totalmente conectadas (*fully connected*), que realizam a classificação final. As secções (b), (c) e (d) representam diferentes arquiteturas de redes neuronais, com (b) ilustrando uma rede tradicional, (c) enfatizando a hierarquia de conexões numa rede profunda e (d) mostrando um modelo que combina operações de redução (*down-sampling*) e aumento de resolução (*up-sampling*), possivelmente representando uma arquitetura voltada para segmentação de imagens [32].

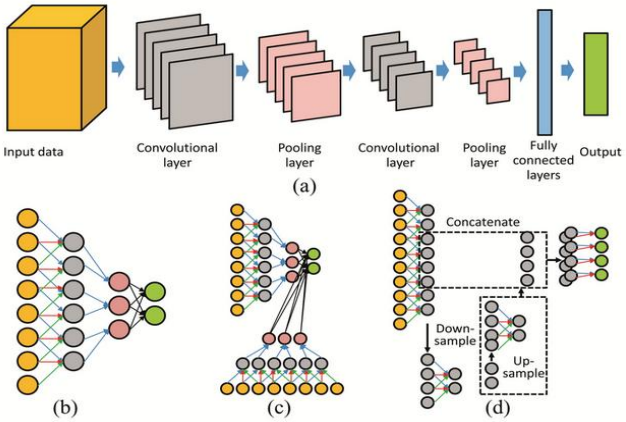

Figura 2 – Visão geral da arquitetura de uma CNN [32]

2.1.3.2 Recurrent Neural Networks (RNNs)

As *Recurrent Neural Networks* (*RNNs*) são redes neuronais especializadas no processamento de dados sequenciais, sendo especialmente úteis para análise de séries temporais, reconhecimento de padrões dinâmicos e previsão de estados futuros. No contexto da visão computacional aplicada a robôs assistivos, as *RNNs* podem ser utilizadas para modelar o comportamento e a movimentação de objetos móveis dentro do ambiente doméstico, permitindo ao robô antecipar mudanças e adaptar suas ações de forma mais eficaz [33].

Diferente das *CNNs*, que processam imagens estáticas, as *RNNs* incorporam uma componente de memória, permitindo que o modelo retenha informações sobre estados anteriores [33]. A evolução do estado oculto de uma *RNN* pode ser expressa matematicamente como (20):

$$\boldsymbol{h_t = \sigma(W_h h_{t-1} + W_x x_t + b)} \quad (20)$$

onde:

- h_t representa o estado oculto no instante t, armazenando informações acumuladas ao longo do tempo.
- x_t é a entrada no instante t, como uma imagem ou um conjunto de características extraídas por uma *CNN*.
- W_h e W_x são matrizes de pesos treináveis que determinam a influência de estados passados e da entrada atual.
- b é o viés da rede.
- σ é uma função de ativação.

As *RNNs* podem ser combinadas com *CNNs* para criar arquiteturas híbridas que permitem ao robô não apenas identificar objetos, mas também prever padrões de movimentação, melhorando a sua capacidade de interação com o ambiente e os utilizadores [33].

2.1.3.3 You Only Look Once (YOLO)

O *You Only Look Once* (*YOLO*) é um dos algoritmos mais eficientes para deteção de objetos em tempo real. Ao contrário de abordagens tradicionais que dividem a tarefa de deteção em múltiplas fases, como a extração de regiões de interesse e posterior classificação, o *YOLO* realiza a deteção em uma única passagem pela rede neural, o que o torna extremamente rápido e adequado para aplicações onde a baixa latência é essencial [34].

A base do *YOLO* é uma CNN que processa uma imagem de entrada e a divide em uma grelha de $S\ x\ S$ células, onde cada célula é responsável por prever um conjunto de *bounding boxes* e as respetivas classes associadas. A saída do modelo pode ser representada matematicamente como (21):

$$\boldsymbol{Y} = \{\left(\boldsymbol{b_x}, \boldsymbol{b_y}, \boldsymbol{b_w}, \boldsymbol{b_h}, \boldsymbol{c}, \boldsymbol{p_1}, \boldsymbol{p_2}, \dots, \boldsymbol{p_C}\right)\} \quad (21)$$

onde:

- (b_x, b_y) representam as coordenadas do centro da caixa delimitadora.
- (b_w, b_h) representam a largura e altura da caixa.
- c indica a confiança da previsão da caixa.
- $p_1, p_2, \dots, p_C$ representam as probabilidades de cada uma das C classes.

Desde a sua introdução, várias versões do *YOLO* foram desenvolvidas, incluindo *YOLOv2*, *YOLOv3*, *YOLOv4* e *YOLOv5*, cada uma trazendo melhorias em termos de precisão e eficiência computacional [34]. Neste trabalho, a versão específica do YOLO a utilizar é apresentada e justificada na secção 3.3.2.1, tendo em conta as necessidades computacionais e a complexidade do ambiente.

A Figura 3 ilustra o funcionamento do algoritmo *YOLO* na deteção de objetos. Inicialmente, a imagem de entrada é dividida numa grelha $S\ x\ S$, onde cada célula é responsável por prever *bounding boxes* e a confiança associada à deteção. Em seguida, um mapa de probabilidades de classe é gerado, indicando a presença de diferentes objetos em áreas específicas da imagem. Após a aplicação de filtros, apenas as caixas com maior confiança são mantidas, resultando nas deteções finais, onde cada objeto identificado é rodeado por uma *bounding box* correspondente.

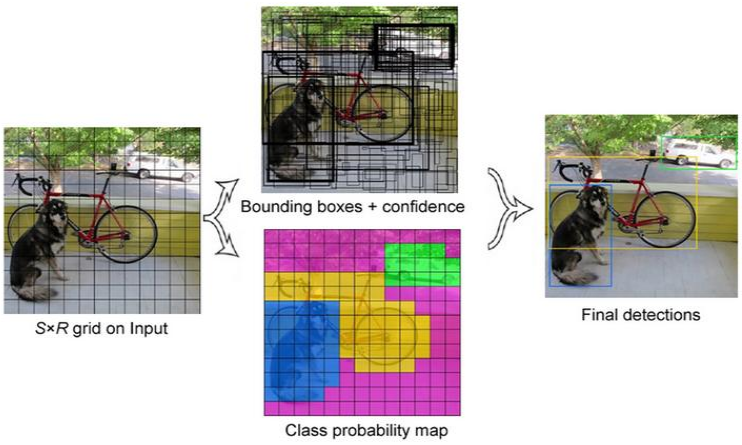

Figura 3 – Deteção do modelo YOLO [35]

2.2 Questões de investigação

A investigação foi inicialmente guiada por uma pergunta principal, complementada por subperguntas que permitiram explorar os diversos aspetos técnicos e práticos relevantes para o desenvolvimento do assistente robótico. A pergunta principal é:

- Como desenvolver um assistente robótico baseado em métodos de aprendizagem para localização e recuperação de objetos em ambientes domésticos, promovendo a autonomia e a qualidade de vida dos utilizadores?

Para responder a esta questão, foram definidas várias subperguntas:

- RQ-1 Que algoritmos de navegação permitem ao robô mapear o espaço e calcular caminhos seguros e otimizados?
  - *Justificação*: Avaliar os métodos de navegação que possibilitam ao robô compreender o ambiente doméstico, mapear trajetos e deslocar-se de forma eficiente e segura até ao objeto desejado.
- RQ-2 Quais tecnologias de visão computacional são mais adequadas para identificar e localizar objetos de forma eficiente em ambientes domésticos?
  - *Justificação*: Identificar as soluções tecnológicas existentes e mais eficazes para o reconhecimento e localização de objetos, utilizando algoritmos de visão computacional.
- RQ-3 Como é que as técnicas de aprendizagem automática podem ser aplicadas para melhorar continuamente o desempenho do robô?
  - *Justificação*: Estudar a aplicação de métodos de *reinforcement learning* e outras técnicas de aprendizagem automática, garantindo que o robô melhora o seu desempenho com base nas experiências acumuladas e no feedback do utilizador.

- RQ-4 Como podem ser aplicadas estratégias de interação homem-robô de forma a garantir uma comunicação intuitiva e eficiente com o assistente robótico em ambientes domésticos?
    - *Justificação*: Avaliar métodos que permitam ao robô interpretar comandos e responder de forma eficaz, assegurando uma interação simples e funcional com o utilizador

Importa salientar que, embora todas as subperguntas tenham sido consideradas na fase de planeamento e revisão da literatura, a implementação prática não contemplou todas as soluções inicialmente previstas. Em particular, as técnicas de *reinforcement learning* associadas à RQ-3 foram analisadas e enquadradas teoricamente, mas não integradas na versão final do sistema devido a limitações técnicas e de recursos, sendo esta decisão detalhada na secção 3.3.4.

2.3 String de procura

A string de procura foi organizada com base nas subperguntas definidas na secção 2.2, com cada *scope* a representar um tema específico diretamente relacionado com uma das subperguntas de investigação. A construção da string recorreu ao uso de operadores booleanos, onde o operador OR foi utilizado dentro de cada *scope* para combinar palavras-chave e sinónimos relacionados, enquanto o operador AND foi aplicado entre os *scopes* para assegurar a combinação lógica dos diferentes temas, garantindo que os resultados abordem todos os tópicos em simultâneo, conforme apresentado na Tabela 1.

Scope	String
Navegação	("indoor navigation" OR "path planning" OR "navigation system" OR "motion planning" OR "obstacle avoidance" OR "autonomous navigation") AND
Reconhecimento de Objetos	("Object recognition" OR "object localization" OR "computer vision" OR "object detection" OR "deep learning for vision" OR "scene understanding") AND
Reinforcement learning	("reinforcement learning" OR "adaptive learning" OR "deep reinforcement learning" OR "reward-based learning") AND
Robótica e Interação	("robot" OR "robotics" OR "robot control" OR "manipulation systems" OR "robot arm" OR "human-robot interaction" OR "interaction strategies")

Tabela 1 – String de procura para bases de dados

2.4 Critérios de inclusão e exclusão

Nesta secção, são apresentados os critérios de inclusão e exclusão aplicados na terceira fase do processo do PRISMA (*Eligibility*). Estes critérios asseguram que apenas estudos relevantes, alinhados com os objetivos da investigação, sejam considerados, garantindo a qualidade e a pertinência dos resultados obtidos. A Tabela 2 sintetiza os critérios de inclusão, enquanto a Tabela 3 apresenta os critérios de exclusão.

Tabela 2 – Critérios de inclusão

IC1	O artigo deve abordar o desenvolvimento de robôs, focados em auxiliar pessoas na realização de tarefas específicas.
IC2	O estudo deve incluir tecnologias para navegação autónoma ou manipulação espacial em ambientes tridimensionais, como mapeamento, planeamento de percursos ou interação com objetos.
IC3	O artigo deve explorar algoritmos de visão computacional, focados na navegação do robô, ou com possibilidade de aplicação à deteção e identificação de objetos.
IC4	O trabalho deve apresentar aplicações ou técnicas de *reinforcement learning* ou outras abordagens de aprendizagem automática.

Tabela 3 - Critérios de exclusão

EC1	O artigo não está disponível em inglês ou português.
EC2	O estudo não inclui robôs que estabelecem qualquer tipo de interação com humanos ou utilizadores, incluindo interação mediada por interfaces.
EC3	O documento é uma revisão, editorial, *opinion piece* ou qualquer publicação que não seja *peer-reviewed*.
EC4	O artigo não inclui informação obtida em ambientes “reais” e não simulados ou casos onde os dados utilizados tenham sido validados com cenários reais.
EC5	O artigo foca-se em soluções destinadas a ambientes com interações intensivas entre múltiplos agentes ou cenários de alta densidade e complexidade, como trânsito automóvel ou espaços públicos densamente povoados.
EC6	O artigo deve ser publicado entre 2019 e 2025 numa base de dados científica reconhecida, como *IEEE Xplore*, *Web of Science* ou similares.

2.5 Seleção de Estudos

Nesta secção, é apresentado o processo de seleção de artigos, conduzido com base na metodologia PRISMA. A aplicação desta abordagem assegura que todos os critérios definidos são aplicados de forma sistemática e reproduzível, permitindo uma triagem transparente e estruturada das fontes relevantes. O fluxograma do PRISMA na Figura 4 ilustra as etapas seguidas.

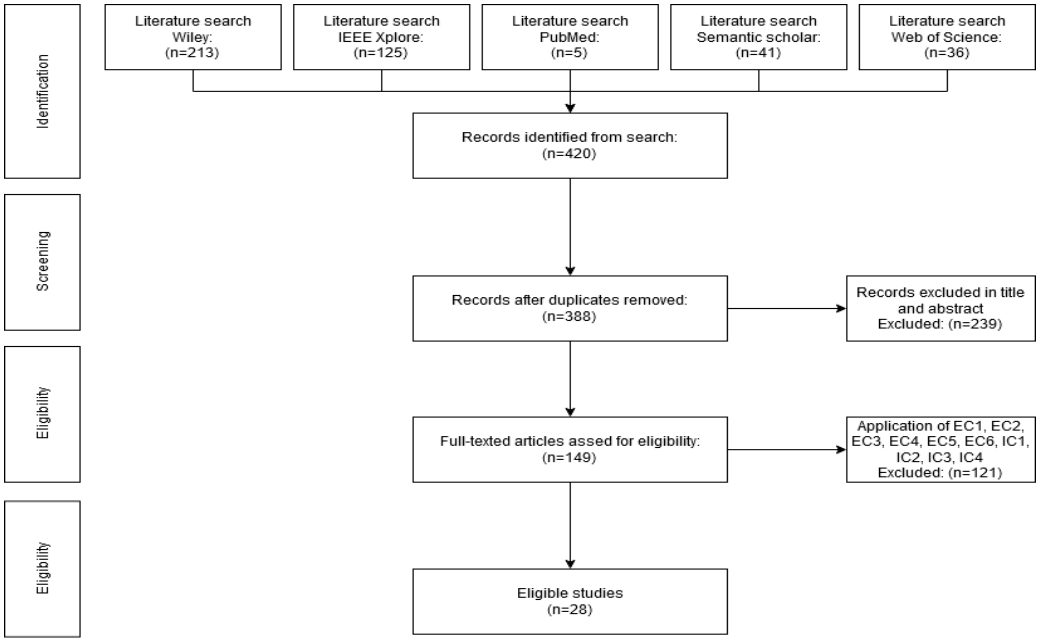

Figura 4 – Fluxograma do PRISMA

2.6 Resultados PRISMA

A secção que se segue introduz o processo de revisão sistemática realizado com base na metodologia PRISMA. Durante a fase de identificação (*Identification*), foi aplicada nas bases de dados uma restrição temporal, considerando apenas artigos publicados entre 2019 e 2025. Esta limitação inicial, que reflete diretamente o critério de exclusão EC6, assegurou que os resultados obtidos estivessem atualizados, embora o critério continue a ser aplicado nas fases seguintes para evitar exceções.

Na fase de triagem (*Screening*), foram excluídos artigos que abordavam robôs inseridos em contextos não relacionados com a temática principal do trabalho, como robôs aquáticos, drones aéreos ou sistemas de condução autónoma. Também foram removidos estudos que exploravam interações intensivas entre múltiplos robôs, dado que não se alinham com o foco deste projeto em assistentes robóticos individuais.

Finalmente, durante a etapa de elegibilidade (*Eligibility*), foram avaliados os artigos que passaram pela triagem, garantindo que cumpriam estritamente todos os critérios de inclusão e

não continham nenhum dos critérios de exclusão. Como resultado, obteve-se um conjunto final de 28 artigos selecionados, que podem ser consultados no Apêndice A – Artigos resultantes do PRISMA. Este número, embora modesto, foi considerado suficiente para construir uma base sólida para o estado da arte, respondendo às subperguntas de investigação e, consequentemente, à pergunta principal.

2.7 Análise de Dados

2.7.1 RQ-1 Que algoritmos de navegação permitem ao robô mapear o espaço e calcular caminhos seguros e otimizados?

DRL destaca-se como uma abordagem central, permitindo treinar robôs para navegar autonomamente, integrando sensores visuais para mapear ambientes e identificar obstáculos de forma eficiente [36], [37], [38]. Modelos hierárquicos, como o DQN, são usados para dividir tarefas complexas em subtarefas mais geríveis, otimizando o desempenho em cenários internos [39], [40].

Em adição, estratégias de *predictive planning*, que se baseiam na previsão da intenção humana, são adotadas para ajustar trajetórias em tempo real, garantindo segurança e eficiência durante a navegação [39], [41]. *Kalman filters* podem desempenhar um papel crucial, integrando dados sensoriais e previsões em tempo real para fornecer estimativas contínuas do estado do robô, mesmo na presença de ruído nos sensores. Esta técnica é amplamente usada para combinar leituras de LIDAR, câmaras e outros sensores, assegurando que os cálculos de posição e trajetória sejam otimizados [42].

Para calcular caminhos seguros, algoritmos de pesquisa em grafos, como o *A** e suas variantes, são amplamente aplicados para gerar trajetórias otimizadas a partir de mapas criados com sensores de profundidade ou LIDAR [43], [44].

Modelos de SLAM são frequentemente utilizados, pois permitem que o robô mapeie o ambiente e se localize simultaneamente. Esta técnica, frequentemente combinada com *reinforcement learning*, assegura que o robô possa operar em ambientes dinâmicos com layouts variáveis [37], [42]. Complementando o SLAM, a *visual odometry* utiliza sequências de imagens para calcular o movimento do robô, fornecendo uma estimativa contínua da posição e orientação, essencial para tarefas de precisão em tempo real [45].

Por fim, *deep neural networks* integram dados complexos, como imagens e nuvens de pontos do LIDAR, para auxiliar o robô na tomada de decisões durante a navegação. CNNs são usadas para interpretar dados visuais, enquanto RNNs analisam sequências temporais, permitindo previsões mais precisas e uma maior adaptação do robô ao ambiente [43], [46].

2.7.2 RQ-2 Quais tecnologias de visão computacional são mais adequadas para identificar e localizar objetos de forma eficiente em ambientes domésticos?

O uso de CNNs tem se destacado pela sua capacidade de reconhecer objetos com alta precisão em ambientes dinâmicos, particularmente em sistemas que utilizam YOLO para deteção rápida e eficiente de múltiplos alvos [47], [48]. A integração de técnicas como *visual odometry* para cálculo de movimentos e estimativas posicionais com base em sequências de imagens é amplamente aplicada em robôs que necessitam operar em espaços restritos [49], [50].

Modelos que combinam visão computacional com *reinforcement learning* têm se mostrado eficazes para a navegação visual orientada por objetivos em cenários internos. Esses sistemas utilizam módulos de perceção visual para identificar alvos específicos e ajustam suas trajetórias dinamicamente para melhorar a eficiência da navegação [41], [51].

Outro destaque é o uso de algoritmos que modelam o comportamento humano de busca a partir de dados de visão egocêntrica. Essas técnicas possibilitam que os robôs simulem estratégias humanas de busca por objetos, melhorando a sua eficiência em ambientes domésticos dinâmicos [49]. A combinação de *multi-view stereo* e sistemas de câmaras permite criar mapas detalhados e estimar profundidade, o que é essencial para tarefas precisas de localização de objetos [44].

2.7.3 RQ-3 Como é que as técnicas de aprendizagem automática podem ser aplicadas para melhorar continuamente o desempenho do robô?

As técnicas de aprendizagem automática devem ter um papel crucial na melhoria contínua do desempenho de robôs, especialmente no contexto de ambientes domésticos. Métodos como DRL são amplamente utilizados para ajustar as estratégias do robô com base no feedback do ambiente, permitindo-lhe adaptar-se a condições dinâmicas e melhorar a sua eficiência em tempo real [52], [53], [54]. A incorporação de *Imitation Learning*, opcionalmente, poderá também ser uma vertente a explorar, esta que possibilita que o robô aprenda diretamente a partir de demonstrações humanas, reduzindo significativamente o tempo necessário para alcançar um desempenho otimizado em tarefas complexas [55]. *Frameworks* interativos que integram *visão computacional* e *linguagem natural* poderão ser utilizados para melhorar a interface entre o robô e o utilizador, promovendo interações mais intuitivas e permitindo ajustes contínuos com base no feedback do utilizador [56].

Modelos de planeamento de movimento em tempo real, integrando sensores visuais e algoritmos preditivos, contribuem significativamente para otimizar as trajetórias e reduzir erros durante as operações do robô. Estes métodos utilizam o feedback do ambiente para ajustar as ações do robô, garantindo tanto a segurança quanto a eficiência [57], [58]. Em paralelo, abordagens baseadas em *Transfer Learning* deverão permitir que o robô utilize conhecimento previamente adquirido para acelerar o processo de adaptação a novas tarefas e ambientes. Esta técnica potencialmente possibilita a transferência de conhecimento entre robôs mesmo que

pertençam a diferentes utilizadores, reduzindo significativamente o tempo e os recursos necessários para o treino [59].

2.7.4 RQ-4 Como podem ser aplicadas estratégias de interação homem-robô de forma a garantir uma comunicação intuitiva e eficiente com o assistente robótico em ambientes domésticos?

Frameworks interativos que integram *visão computacional* e *linguagem natural* destacam-se como estratégias eficazes para promover uma comunicação intuitiva entre o robô e o utilizador [56]. Abordagens baseadas em navegação visual egocêntrica e modelagem comportamental poderão ser utilizadas para adaptar as ações do robô ao contexto do utilizador, assegurando interações mais naturais [49], [60].

Estratégias de aprendizagem por imitação, como já mencionadas na secção 2.7.3, poderão permitir que o robô adapte a sua interação ao observar padrões humanos, melhorando a fluidez da comunicação e reduzir o tempo de adaptação a diferentes utilizadores [61]. Sistemas que combinam reconfiguração em tempo real e previsão de intenções humanas deverão ajustar automaticamente as ações do robô para se alinharem às necessidades específicas de cada ambiente doméstico [60].

Técnicas que transferirão conhecimento entre robôs, especialmente em cenários onde múltiplos utilizadores estiverem envolvidos, deverão contribuir para uniformizar padrões de interação e reduzir a necessidade de treino extensivo [62], [63]. Estas estratégias, ao combinarem adaptação dinâmica e tecnologias avançadas, asseguram que a interação homem-robô se torne cada vez mais intuitiva.

2.7.5 Conclusão geral

A investigação permitiu identificar abordagens avançadas para a navegação autónoma, visão computacional, aprendizagem automática e interação homem-robô no desenvolvimento do assistente robótico. DRL, aliado a DQN, demonstrou ser uma opção possível e eficaz na navegação autónoma, permitindo adaptação contínua a ambientes dinâmicos [36], [37], [38]. SLAM e *Kalman filters* devem garantir um mapeamento preciso, enquanto algoritmos como A* otimizam trajetórias e evitam obstáculos [39], [40], [42]. A *visual odometry* revelou-se essencial para o posicionamento contínuo [45], e abordagens de *adaptive motion planning* deverão ajustar trajetórias em tempo real [41]. Redes neuronais profundas, como CNNs e RNNs, poderão melhorar a interpretação do ambiente e a tomada de decisão [43], [46].

Na visão computacional, CNNs e YOLO destacaram-se na deteção rápida e precisa de objetos [47], [48], especialmente enquanto a *semantic segmentation* aprimora a compreensão do ambiente [47]. Modelos inspirados em estratégias humanas de busca, baseados em visão egocêntrica, deverão aumentar a eficiência do robô na procura de objetos [49]. A fusão de *multi-view stereo* e câmaras deverão alcançar uma melhor estimativa de profundidade [44].

Na interação homem-robô, *frameworks* interativas que combinam visão computacional e linguagem natural emergiram como soluções eficazes para tornar a comunicação mais intuitiva [56]. A navegação visual egocêntrica e a modelação comportamental foram identificadas como abordagens úteis para permitir que o robô ajuste as suas ações ao contexto do utilizador, assegurando interações mais naturais [49], [60]. A reconfiguração em tempo real e a previsão da intenção do utilizador mostraram-se estratégias eficazes para ajustar automaticamente as ações do robô às necessidades específicas de cada ambiente doméstico [60]. Finalmente, a transferência de conhecimento entre robôs poderá desempenhar um papel importante na uniformização de padrões de interação e na redução do tempo de treino necessário para novos utilizadores, tornando a adaptação do sistema mais eficiente [62], [63].

3 Métodos e ferramentas

Esta secção descreve os métodos e ferramentas utilizados no desenvolvimento do sistema, abrangendo as características do robô físico, a arquitetura geral, as tecnologias selecionadas e o processo de integração dos componentes.

3.1 Robô físico

O robô escolhido para a implementação do projeto é o Transbot SE, um robô com capacidade de movimento retilíneo (frente e trás) e rotação sobre o seu eixo vertical (eixo Z), equipado com:

- **Câmara com 2 graus de liberdade,** possibilitando o ajuste dinâmico do ângulo de visão para reconhecimento e acompanhamento de objetos;
- **Braço robótico com 3 graus de liberdade,** destinado à manipulação de pequenos objetos com precisão suficiente para tarefas experimentais;
- **Computador de bordo (Raspberry Pi 5),** que assegura o processamento local das tarefas de perceção e controlo;
- **Placa principal de controlo (Main Control Board)**, que assegura a comunicação eficiente entre o Raspberry Pi e os motores das rodas e da câmara.

O sistema baseia-se no ROS Melodic, que disponibiliza as ferramentas e estruturas necessárias à comunicação entre sensores, atuadores e módulos de planeamento. A sua arquitetura modular, baseada em nós, permite uma separação funcional clara e eficiente entre os diferentes componentes do sistema. O ROS facilita também a escalabilidade do projeto e a integração com pacotes de código *open-source* para tarefas como mapeamento, localização e navegação [64].

Embora o ROS suporte várias linguagens de programação, o desenvolvimento deste projeto foi realizado exclusivamente em Python, dado o seu ecossistema robusto para aplicações de robótica e visão artificial, bem como a facilidade de integração com os pacotes utilizados.

A Figura 5 apresenta o Transbot SE utilizado neste trabalho, ilustrando a sua configuração física e a disposição dos principais componentes de hardware descritos, o que permite uma melhor perceção das suas capacidades e limitações no contexto experimental.

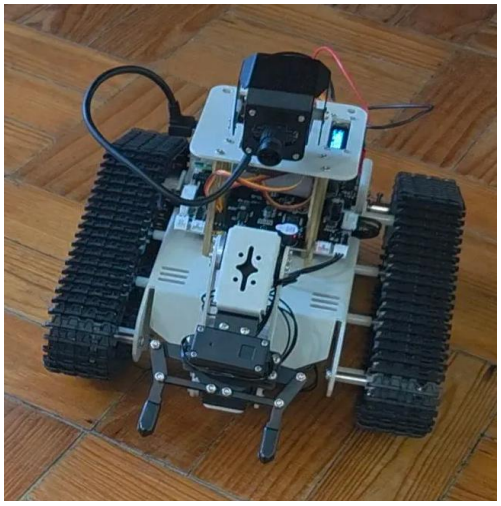

Figura 5 – Foto do Transbot SE

3.2 Arquitetura geral do Sistema

A arquitetura do sistema foi concebida para cumprir os objetivos funcionais de deteção, aproximação, captura e entrega de objetos em ambientes domésticos, utilizando apenas uma câmara como sensor. Durante a fase inicial de testes verificou-se uma limitação crítica no fornecimento de energia ao Raspberry Pi 5, que afeta diretamente o seu desempenho. O robô é alimentado por uma *lithium battery* de 12 V / 4400 mAh, esta que fornece energia a uma *expansion board* responsável por distribuir a alimentação aos vários componentes, incluindo o Raspberry. Quando em operação, especialmente durante tarefas que exigem maior consumo de energia, como a execução simultânea do modelo YOLO e a locomoção, o Raspberry sofre quedas de tensão significativas que levam não apenas à redução de desempenho, mas, em muitos casos, ao bloqueio completo ou desligamento inesperado do sistema. Este comportamento torna inviável a execução local de tarefas de deteção intensivas enquanto o robô está em movimento. Como podemos ver na Figura 6, o alerta de *low voltage* é apresentado sempre que a tensão de alimentação desce abaixo do valor mínimo necessário, evidenciando a incapacidade da bateria de fornecer energia estável em cenários de carga elevada e obrigando à reavaliação da forma como as tarefas de processamento são distribuídas.

Figura 6 – Aviso de *low voltage* do Raspberry Pi 5

Para contornar esta limitação, optou-se por uma separação clara de responsabilidades, o robô executa as tarefas de aquisição de imagem, controlo de movimento e acionamento da garra,

enquanto um servidor remoto (um portátil MSI GF63 Thin 9SC, equipado com processador Intel Core i7 de 9ª geração e GPU NVIDIA GTX 1650) processa operações computacionalmente exigentes, como a deteção de objetos através do modelo YOLO. Esta abordagem evita que o Raspberry entre em falha assegurando maior fiabilidade e continuidade de operação durante a locomoção. A Figura 7 apresenta uma visão de alto nível dos principais atores do sistema, destacando a interação entre o utilizador, o servidor remoto e o robô Transbot SE. O diagrama ilustra de forma simplificada como o utilizador desencadeia um cenário passivo através do servidor, que por sua vez comunica com o robô.

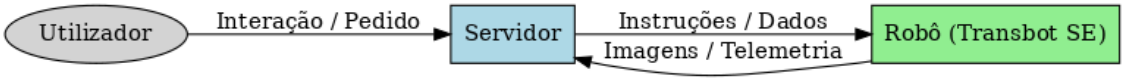

Figura 7 – Diagrama de alto nível dos atores do sistema

Além da deteção e movimentação, a arquitetura integra ainda funcionalidades de apoio à navegação e à procura de objetos. O servidor mantém em tempo real um mapa *top-down* com a posição estimada do robô e a localização dos objetos detetados, regista em ficheiros CSV cada deteção dos objetos (coordenadas, hora e classe) e guarda também os pedidos do utilizador para iniciar procuras, incluindo a hora, data e a indicação do objeto pretendido. Para suporte visual e monitorização, o servidor apresenta também uma *live feed* proveniente da câmara do robô, já processado pelo YOLO, com as deteções sobrepostas na imagem. Toda a comunicação entre robô e servidor é realizada através de um protocolo simples e fiável baseado em TCP, garantindo a integridade e a ordem das mensagens trocadas. A Figura 8 mostra a interface do servidor, à esquerda, o *live feed* com as deteções e à direita, o mapa top-down.

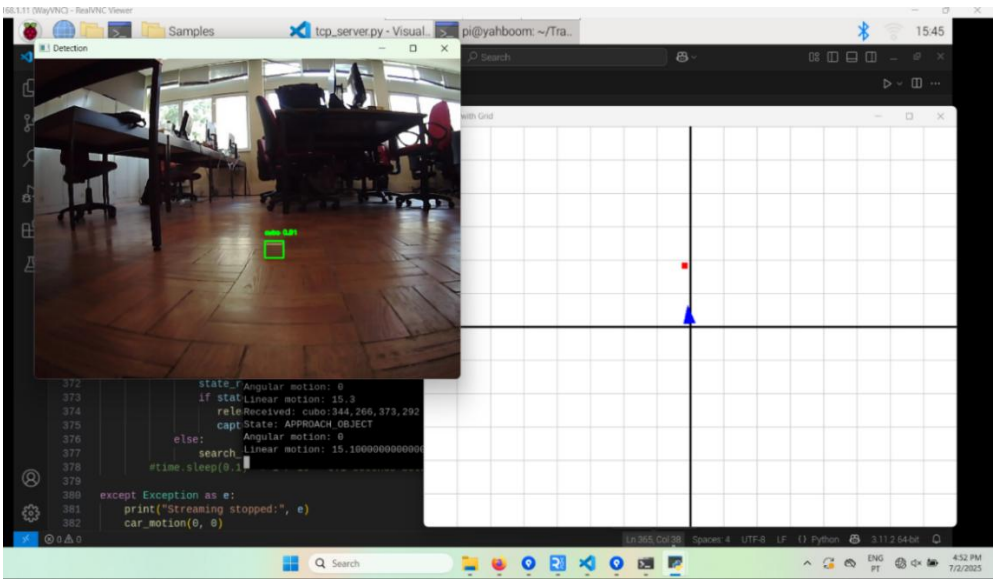

Figura 8 – Visão da interface do servidor

Sempre que é iniciado um ciclo de procura, o servidor envia ao robô um conjunto de três pontos de interesse, calculados por um modelo preditivo a partir do histórico de deteções. Este modelo utiliza dados recolhidos previamente, resultantes do registo de deteções passadas para estimar

as localizações mais prováveis onde cada tipo de objeto poderá ser encontrado. A definição destes pontos tem como intuito forçar o robô a adotar uma estratégia de busca mais direcionada, podendo reduzir deslocações desnecessárias e aumentando a probabilidade de sucesso na recolha. Os detalhes sobre o modelo utilizado e o seu funcionamento são apresentados na secção 3.3.3.

3.2.1 Divisão de processamento

Como mencionado na secção 3.2, dada a limitação de desempenho do Raspberry Pi 5 causada por problemas de subtensão, o sistema foi concebido com uma divisão clara de responsabilidades. O robô executa apenas tarefas críticas de controlo e aquisição de dados, enquanto o processamento intensivo é delegado a um servidor remoto. A comunicação entre ambos é contínua e bidirecional. O robô envia não só *frames* para análise, mas também todas as ações executadas (movimentos, estados da garra, eventos como a captura de um objeto), permitindo ao servidor manter uma visão completa e atualizada do estado do sistema. Foi também considerada a possibilidade de falhas inesperadas na comunicação. Determinadas funcionalidades ainda não foram implementadas, sendo discutidas em maior detalhe na secção 6.2. A Tabela 4 resume as principais funções atribuídas a cada lado do sistema.

Tabela 4 – Principais funções atribuídas a cada lado do sistema

Robô (on-board, Raspberry Pi 5)	**Servidor (processamento pesado)**
Aquisição de imagem da câmara e envio periódico de *frames* ao servidor.	Execução da inferência YOLO para deteção dos objetos (com filtragem por confiança).
Execução dos comandos de movimento (frente/trás, rotação no eixo Z) e controlo da garra (3 DOF).	Normalização e envio das *bounding boxes* detetadas.
Lógica reativa leve baseada nas deteções recebidas: – correção de rumo (esquerda/direita) com base na posição horizontal do alvo; – decisão de avanço/recuo pela posição vertical.	Manutenção e atualização da pose global do robô (X, Y, θ) com base nas ações reportadas e modelo cinemático simples.
Envio de telemetria: – Pose estimada/odometria; – Todos os movimentos executados; – Estado e ações da garra; – Eventos (p.ex., captura de objeto).	Renderização em vista *top-down* da posição do robô e dos objetos detetados.
Gestão da ligação TCP: reconexão e *timeouts*.	Registo estruturado de deteções (CSV: X, Y, timestamp, classe, confiança, *bbox*).
–	Previsão de até três pontos prováveis de localização do alvo, com base no histórico de deteções.

3.2.2 Comunicação e protocolos

A comunicação entre o robô e o servidor é estabelecida através de uma ligação TCP persistente, que combina um protocolo textual para comandos e eventos com um canal binário prefixado por tamanho para a transmissão das imagens da câmara. O Snippet 1 mostra a sequência de inicialização desta ligação no lado do servidor. Em primeiro lugar, cria-se a socket TCP, associa-se ao endereço e porta definidos, *bind*, e o servidor entra em modo de escuta *listen*.

Dentro do ciclo principal, o servidor aguarda a ligação do Raspberry Pi *accept*, bloqueando até que o pedido chegue. Uma vez estabelecida a ligação, são impressos no terminal os detalhes do cliente conectado, e o controlo é passado para a função *handle_connection*, responsável por receber e processar todas as mensagens durante a sessão. Este ciclo repete-se indefinidamente, permitindo reconexões caso a sessão seja terminada.

No contexto do protocolo, a mensagem *search* desempenha o papel de comando de arranque. Quando recebida, desencadeia no robô o início do ciclo de procura de objetos. O envio dessa mensagem, com os três pontos de interesse, pode ocorrer por ação direta do utilizador (cenário passivo) ou automaticamente pelo módulo preditivo (cenário proativo).

Snippet 1 – Ciclo de receção do servidor com ligação persistente

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as
server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen(1)
    print(f"Server listening on {HOST}:{PORT}. Press Ctrl+C
to stop.")

    try:
        while True:
            print("Waiting for Raspberry Pi connection...")
            conn, addr = server_socket.accept()
            print(f"Connected by {addr}")

            # Loop principal de processamento
            handle_connection(conn)

    except KeyboardInterrupt:
        print("\nServer stopped by user.")
```

O protocolo implementado define um conjunto reduzido de mensagens com formato fixo, permitindo distinguir rapidamente comandos, notificações e dados de deteção. Estas mensagens, trocadas em ambos os sentidos durante a sessão TCP, asseguram a coordenação entre o processamento remoto e o controlo local do robô. A Tabela 5 apresenta os tipos de mensagens que o robô envia para o servidor.

Tabela 5 – Mensagens enviadas pelo robô para o servidor

Robô → Servidor		
Nome	**Formato**	**Descrição**
Frame	(binário)	Imagem da câmara enviada periodicamente (JPEG bruto ou codificado em base64). Não é interpretada como mensagem textual.
motion	*"motion:<linear_speed>,<angular_speed>"*	Enviada a cada *frame*, indica as velocidades linear e angular efetivamente aplicadas pelo robô.
get_current	*"get_current"*	Solicita ao servidor a posição atual do robô no espaço (X, Y, θ).
captured	*"captured:<nome_objeto>"*	Notifica o servidor de que o objeto indicado foi agarrado com sucesso.
finished	*"finished"*	Notifica o servidor de que o ciclo de procura foi terminado

A Tabela 6 descreve as mensagens enviadas pelo servidor para o robô, que incluem os resultados da deteção de objetos, informações de posicionamento e o envio dos pontos de procura sugeridos pelo modelo preditivo.

Tabela 6 – Mensagens enviadas pelo servidor para o robô

Servidor → Robô		
Nome	**Formato**	**Descrição**
detections	*"<label>:<x1>,<y1>,<x2>,<y2>"*	Indica o objeto detetado pelo YOLO. Apenas é enviado o mais próximo (centro mais baixo na imagem), mesmo que haja mais de um.
current	*"current:<robot_x>,<robot_y>,<robot_theta>"*	Resposta ao pedido *get_current*, contendo posição (X,Y) e ângulo θ do robô.
search	*"search: (<x1,y1>), (<x2,y2>), (<x3,y3>)"*	Enviada no início do ciclo de procura, indica três pontos de interesse gerados pelo modelo preditivo para orientar a busca.

3.3 Tecnologias a utilizar

Esta secção apresenta as principais tecnologias de software e hardware aplicadas no projeto, descrevendo o seu papel no funcionamento do sistema. Inclui desde os mecanismos de navegação e controlo, visão computacional, localização e estratégias de procura e aborda também decisões de design, como a escolha de algoritmos e justificações para funcionalidades não implementadas nesta iteração.

3.3.1 Navegação e controlo de movimento

O sistema de navegação e controlo de movimento foi desenvolvido de forma a permitir que o robô Transbot SE se desloque de forma controlada, utilizando apenas a informação proveniente da câmara como referência sensorial. Dada a ausência de sensores adicionais, como LIDAR ou ultrassons, e considerando as restrições de processamento impostas pelo computador de bordo, optou-se por uma abordagem simples, mas fiável, baseada em comandos discretos de translação e rotação.

O modelo de movimento disponível no Transbot SE para este projeto contempla apenas dois tipos de ações:

- **Movimento retilíneo** para a frente ou para trás.
- **Rotação sobre o seu eixo vertical (eixo Z)**, no sentido horário ou anti-horário.

A Figura 9 ilustra de forma esquemática estes dois graus de liberdade do Transbot SE, representando os movimentos de translação para a frente e para trás, bem como as rotações no sentido horário e anti-horário em torno do seu eixo vertical (eixo Z).

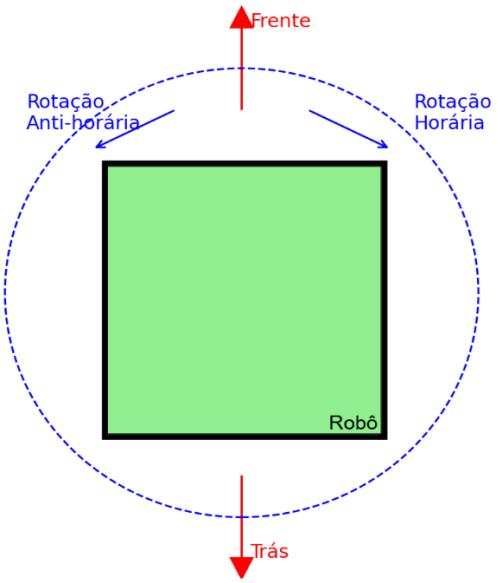

Figura 9 – Movimentos possíveis do robô Transbot SE (translação e rotação)

O controlo destas ações é realizado através de uma função de software executada sempre no próprio robô, que parametriza a velocidade linear e angular do mesmo. A função de controlo principal *car_motion(line, angular)* recebe dois parâmetros:

- line: velocidade linear (positiva para avanço, negativa para recuo).
- angular: velocidade de rotação (positiva para rotação no sentido horário, negativa para anti-horário).

Esta função é chamada de forma iterativa pelo módulo de decisão, que ajusta continuamente os valores de velocidade com base nas coordenadas do objeto detetado na imagem, podendo também ser invocada noutras situações, como no deslocamento para pontos designados fornecidos previamente pelo servidor.

3.3.1.1 Critério de pick-up do objeto

O movimento de aproximação é interrompido quando o objeto se encontra centrado horizontalmente no campo de visão com um desvio inferior a um limiar pré-definido em píxeis e está localizado na região inferior da imagem, indicando curta distância. Estes limiares de centragem e proximidade são configuráveis através de um ficheiro JSON (Snippet 2) no qual se encontram definidos, para cada classe de objeto, os valores de *target_left*, *target_right*, *target_top* e *target_bottom* permitindo afinar e ajustar a posição alvo conforme necessário. Nesse momento é emitido o comando para o braço robótico realizar a preensão utilizando a garra para agarrar o objeto identificado.

3.3.1.2 Estratégia de alinhamento

A navegação até ao objeto baseia-se numa estratégia visual de alinhamento, que consiste num conjunto de regras simples extraídas diretamente da posição do objeto no enquadramento da

câmara, convertendo essa informação em comandos de movimento para orientar e aproximar o robô.

- **Ajuste horizontal**: a posição horizontal do objeto no enquadramento da câmara determina a necessidade de rotação. Caso o centro do objeto esteja desviado para a esquerda ou para a direita da imagem, é emitido um comando de rotação na direção correspondente, com velocidade angular proporcional ao desvio.

- **Ajuste vertical**: a posição vertical do objeto na imagem é utilizada como indicador da proximidade. Objetos situados na zona superior da imagem são interpretados como estando mais afastados, enquanto objetos localizados na zona inferior indicam maior proximidade. O avanço é executado até que o objeto atinja a zona inferior central da imagem.

Do ponto de vista de implementação, os ajustes de alinhamento são realizados através de dois controladores independentes, um para o movimento angular e outro para o linear, ambos definidos como funções proporcionais discretas com limiares mínimos. No caso da rotação, calcula-se a posição horizontal do centro da *bounding box* do objeto (c_x) e compara-se com a zona-alvo definida pelos dois limites *target_left* e *target_right* desse objeto correspondente. Se c_x estiver fora dessa zona, a velocidade angular ω é ajustada segundo a regra mostrada na equação (22):

$$\omega = -k\omega \cdot (c_x - target_{limite}) \quad (22)$$

onde o lado do erro depende da direção do desvio e o ganho $k\omega$ é definido empiricamente (no código, 0.05). Para evitar movimentos demasiado lentos, é imposto um valor mínimo de velocidade angular *min_angular_vel*, que atualmente está configurado como 3 °/s e pode ser alterado no ficheiro de configuração JSON (Snippet 2).

No caso do movimento linear, aplica-se uma lógica semelhante. Calcula-se o centro vertical da bounding box (c_y) e compara-se com os limites *target_top* e *target_bottom* do objeto correspondente. Se c_y estiver acima do limite, o robô avança, se estiver abaixo, recua. A velocidade linear v é ajustada proporcionalmente de acordo com a equação (23):

$$v = -kv \cdot (c_y - target_{limite}) \quad (23)$$

com um ganho kv definido empiricamente (no código, 0.10) e um valor mínimo de velocidade linear *min_linear_vel*, atualmente definido como 5.5 cm/s e igualmente configurável no ficheiro JSON.

No Snippet 2 podemos ver um exemplo da estrutura do ficheiro JSON.

Snippet 2 – Exemplo do ficheiro JSON para os parâmetros de navegação

```
{
  "min_angular_vel": 3,
  "min_linear_vel": 5.5,
  "targets": {
    "cubo": {
      "target_left": 309,
      "target_right": 387,
      "target_top": 413,
      "target_bottom": 452
    },
    "meds": {
      "target_left": 319,
      "target_right": 399,
      "target_top": 420,
      "target_bottom": 463
    }
  }
}
```

3.3.1.3 Tecnologias não adotadas

A revisão do estado da arte identificou várias abordagens sólidas para navegação autónoma, como DRL/DQN, planeamento preditivo, filtros de Kalman, A* em mapas, SLAM, *visual odometry* e redes profundas para tomada de decisão. Nesta implementação, foram excluídas por uma combinação de fatores: limitação computacional do Raspberry Pi 5 (agravada por *undervoltage*), disponibilidade de um único sensor (câmara), ausência de LIDAR/RGB-D, necessidade de baixa latência e a opção deliberada por um protótipo funcional e fiável em ambiente controlado.

O DRL/DQN não foi utilizado pelo custo de treino (tipicamente com GPU e muitas interações em simulação), pela complexidade de definição de recompensas e pela dificuldade de transferir políticas para o robô com os recursos disponíveis. O planeamento preditivo foi considerado desajustado ao cenário de testes, que não exige previsão de trajetórias humanas ou obstáculos móveis, e carece de sensores que suportem modelos de previsão credíveis. *Kalman filters* teriam utilidade real com fusão multi-sensor, mas com apenas visão monocular e sem odometria fiável integrada, a relação custo/benefício não justificou a adoção.

Os algoritmos de planeamento em grafos (A*) pressupõem um mapa consistente do espaço e sem SLAM e sem profundidade, o ganho seria reduzido face ao esquema atual de translações e rotações em zonas desimpedidas. Mais em particular, o Visual SLAM foi efetivamente ponderado, mas descartado nesta fase pelo impacto computacional e pela exigência de extração de *features* estáveis sob variação de iluminação. Em alternativa, o servidor mantém coordenadas globais a partir dos movimentos comandados. Pela mesma razão, a *visual odometry* não foi integrada devido a elevadas exigências de processamento e sensibilidade a *drift* sem sensores auxiliares.

Redes profundas adicionais para navegação (CNN/RNN além do YOLO) foram evitadas para conter latência e complexidade. A decisão de navegação foi mantida heurística, a partir da posição do objeto na imagem, o que cumpre os objetivos experimentais com o hardware disponível. Estas opções não invalidam o interesse das técnicas excluídas, apenas refletem escolhas de engenharia compatíveis com os recursos e o âmbito desta iteração.

3.3.2 Visão computacional

A componente de visão computacional constitui o núcleo de perceção do sistema e foi desenvolvida a partir das tendências identificadas no estado da arte na secção 2.7.2. Entre as várias abordagens possíveis, destacou-se a utilização de redes neuronais convolucionais, em particular o modelo YOLO, devido à sua maturidade, rapidez e adequação a sistemas de tempo real. Como os alvos são apenas duas classes (mostradas em detalhe na secção 4.1.1), deve ser escolhido uma versão do YOLO leve, fiável e com uma pipeline madura de treino e exportação, características que permitem equilibrar a precisão da deteção. Embora técnicas complementares como *visual odometry*, integração de visão com *reinforcement learning* e estratégias multi-câmara apresentem vantagens noutros contextos, a presente implementação optou por uma solução mais simples e focada na deteção direta dos objetos, assegurando a viabilidade prática do protótipo.

3.3.2.1 Seleção da versão do YOLO

O objetivo desta secção é justificar a seleção do YOLOv5 como arquitetura central de deteção de objetos em tempo real neste trabalho. A opção resulta de uma análise comparativa entre algumas versões da família YOLO (v1–v8) e de uma confrontação honesta com os requisitos e constrangimentos do sistema implementado (apenas uma câmara como sensor, necessário ciclo de perceção-ação com baixa latência, servidor remoto a executar a inferência e um computador de bordo limitado).

As iterações iniciais do YOLO (YOLOv1–v3) estabeleceram as bases do paradigma *one-stage [65],* no qual a deteção de objetos é feita num único passo da rede, prevendo simultaneamente *bounding boxes* e classes. O YOLOv4 estabilizou o compromisso precisão/tempo de execução em GPUs convencionais [65]. Em seguida, YOLOv5 consolidou um ecossistema maduro em PyTorch, com *tooling* estável, *export* direto (ONNX, TensorRT), treino/inferência simples e modelos pré-treinados amplamente validados [65], [66]. As versões posteriores (YOLOv7 e YOLOv8) apresentam, em cenários de *benchmark* e hardware topo de gama, ganhos marginais de mean average precision (mAP)/latência por imagem [66], [67]. Esses ganhos, porém, não se traduzem automaticamente em melhor "tempo de ciclo", *end-to-end,* quando o sistema inclui envio de *video frames* por rede e lógica de decisão do lado do robô. Neste projeto, a estabilidade operacional e a integração sem fricção pesam mais do que ganhos teóricos por imagem.

Quanto às variantes de tamanho disponíveis nas diferentes versões do YOLO, a opção recaiu sobre a média por oferecer o melhor equilíbrio entre robustez e eficiência. As versões mais

pequenas reduzem a latência, mas perdem consistência em cenários reais, enquanto as maiores aumentam a precisão à custa de maior carga computacional, sem ganhos práticos num problema restrito a poucas classes. A variante m (quando disponível) costuma representar um compromisso entre desempenho e eficiência, reduzindo latência comparada à variante l e melhorando estabilidade frente à variante s. No caso do YOLOv7, como não há variante m oficial disponível, usou-se a configuração padrão (única) oferecida pela implementação.

A decisão foi guiada por quatro critérios principais que refletem as necessidades do projeto. O primeiro prende-se com a latência útil e previsível. Mais importante do que a velocidade de inferência isolada, medida em milissegundos por imagem, é o tempo de resposta *end-to-end*, que inclui todo o ciclo desde a captura da imagem até à execução de uma ação motora, passando pelo envio da informação ao servidor, a inferência e o retorno do resultado. Um modelo que seja marginalmente mais rápido em teoria, mas sujeito a flutuações devido à carga computacional, não garante um fecho de ciclo mais eficiente. Os testes que resultaram na Figura 10 foram conduzidos em condições reais do sistema desenvolvido.

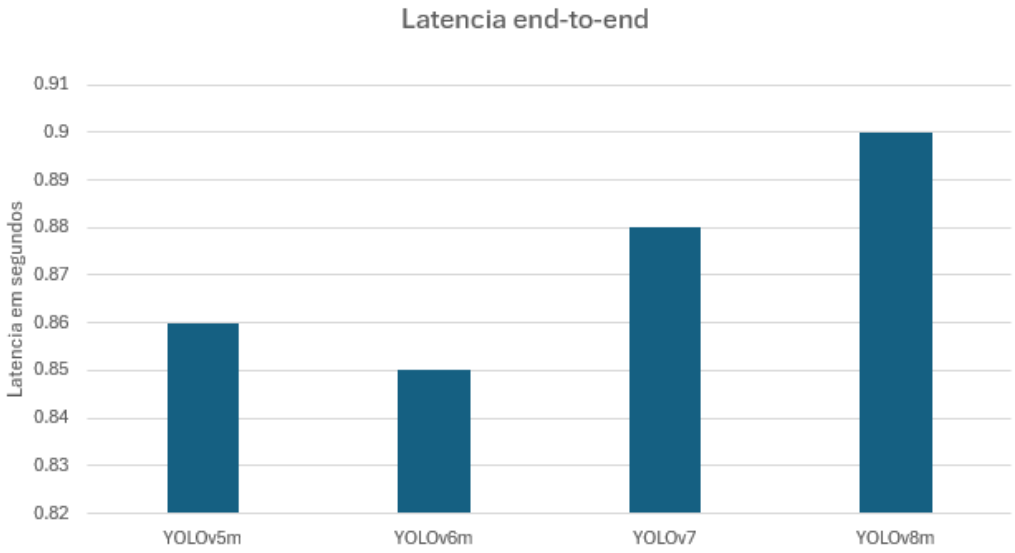

Figura 10 – Testes de latência *end-to-end* de diferentes modelos YOLO

O robô, equipado com uma câmara de 1080p, realizou *downsampling* das capturas para 720×720 píxeis, de forma a equilibrar qualidade visual e eficiência de transmissão. Durante o ensaio, foi utilizado um vídeo previamente gravado, contendo pessoas (classe presente em todos os modelos treinados no Common Objects in Context (COCO)), de forma a garantir que o detetor estivesse constantemente ocupado com inferência de objetos em vez de processar imagens vazias. As imagens foram enviadas via ligação Wi-Fi (TCP) a partir do Raspberry Pi 5 para o computador servidor, um MSI GF63 Thin 9SC com Intel Core i7 de 9ª geração e GPU NVIDIA GTX 1650. No servidor, a inferência foi executada diretamente em PyTorch, recorrendo a modelos YOLOv5m, YOLOv6m, YOLOv7 e YOLOv8m em formato *.pt*, todos previamente treinados no conjunto de dados COCO. Após a deteção, os resultados eram devolvidos ao Raspberry Pi, que acionava de imediato a resposta motora (para efeitos deste teste, a abertura e fecho da garra). A Figura 10 mostra que o YOLOv5m e o YOLOv6m se destacaram por

apresentarem os menores tempos de ciclo, enquanto os modelos YOLOv7 e YOLOv8m revelaram latências ligeiramente superiores. Ainda que o YOLOv5m não seja o modelo de última geração, a sua performance revelou-se sólida e equilibrada no contexto do sistema, comparável a variantes mais recentes e mais exigentes computacionalmente. Importa, contudo, salientar que a maior parte da latência resulta do ciclo completo, incluindo a transmissão pela rede e o acionamento mecânico da garra, pelo que as diferenças entre modelos, embora observáveis, tendem a ser pequenas face ao impacto global do pipeline.

O segundo critério de análise está relacionado com a robustez em cenários com poucas classes. As versões mais recentes da família YOLO foram concebidas para detetar simultaneamente dezenas de classes distintas, refletindo a complexidade do conjunto COCO (80 classes). Essa capacidade extra, embora vantajosa em cenários gerais, perde relevância quando o domínio de deteção é restrito (poucas classes). Nesta situação, a diferença de *mAP* medida em *benchmarks* com múltiplas classes pode não se traduzir em ganhos práticos significativos. Além disso, o ajuste de parâmetros como o limiar de confiança (score mínimo para validar uma deteção) e o Non-Maximum Suppression (NMS) (método que elimina sobreposições entre caixas de deteção) reduz ainda mais as discrepâncias entre modelos. Ao ajustar estes limiares ao contexto do robô, é possível obter deteções estáveis e consistentes.

O terceiro critério prende-se com a maturidade e manutenção do ecossistema. Neste aspeto, o YOLOv5 destaca-se devido à sua documentação extensa e comunidade ativa, sendo possível encontrar pesos pré-treinados, guias de treino e pipelines de exportação estáveis. Isso reduz a probabilidade de enfrentar problemas de integração e poupa tempo em engenharia acessória que não acrescentaria valor direto à investigação.

Por último, foi considerada a compatibilidade com a arquitetura implementada. O sistema foi desenvolvido inteiramente em Python, com a inferência a decorrer remotamente e o robô a executar em tempo real as decisões reativas com base nas deteções. Neste contexto, o YOLOv5 integra-se de forma particularmente fluida, com o suporte oficial em PyTorch, a facilidade de adaptação do código e a compatibilidade direta com exportação para diferentes formatos e servidores.

3.3.2.2 Transfer learning

Para adaptar o modelo às necessidades específicas do sistema desenvolvido, foi aplicada a técnica de *transfer learning* ao YOLOv5m. Em vez de treinar a rede de raiz, recorreu-se a pesos pré-treinados no conjunto COCO, que serviram como ponto de partida. A partir daí, o modelo foi re-treinado com um conjunto reduzido de imagens recolhidas experimentalmente, contendo apenas as classes de interesse. O número total de imagens recolhidas e os detalhes do processo encontram-se descritos com maior rigor na secção 5.1.1. Este processo permitiu ajustar as camadas finais da rede, especializando-a para distinguir com maior precisão os objetos relevantes, sem a necessidade de milhares de imagens rotuladas ou de um tempo de treino excessivo.

3.3.2.3 Tecnologias não adotadas

A revisão do estado da arte destacou várias abordagens complementares à deteção de objetos com YOLO, entre as quais se encontram a *visual odometry*, a integração da perceção visual com *reinforcement learning*, estratégias de procura inspiradas em visão egocêntrica e ainda a utilização de sistemas multi-câmara para estimação de profundidade. Apesar da relevância científica destas soluções, nenhuma foi incluída na implementação desenvolvida nesta dissertação.

No caso da *visual odometry*, apesar de ser um método amplamente utilizado para estimar deslocamentos a partir de sequências de imagens, a sua integração implicaria um fluxo contínuo de processamento visual que não combina com a arquitetura definida. Em teoria, poderia considerar-se a mesma divisão de carga já utilizada para o YOLO e delegar no servidor a execução da *visual odometry*. No entanto, esta solução não seria viável na prática por duas razões principais. Em primeiro lugar, a *visual odometry* exige um envio constante de imagens em sequência com elevada cadência temporal, uma vez que pequenas variações entre *frames* sucessivos são críticas para a estimativa correta do movimento. Esse requisito implicaria um aumento significativo da largura de banda e da latência de comunicação entre o robô e o servidor, o que comprometeria a resposta em tempo real necessária para a navegação. Em segundo lugar, ao contrário da deteção de objetos, em que a informação pode ser processada com alguma tolerância temporal, a *visual odometry* não admite interrupções nem perdas de *frames* sem que a estimativa se degrade de forma rápida. No contexto experimental do projeto, onde a ligação de rede não era dedicada nem imune a perdas, a robustez da solução ficaria seriamente comprometida.

Também a integração entre visão e *reinforcement learning*, frequentemente explorada na literatura para navegação orientada por objetivos, acabou por não ser considerada viável. Esta decisão deveu-se sobretudo ao custo computacional e temporal associado ao treino de políticas de aprendizagem, o que estaria desalinhado com o calendário de desenvolvimento e com os recursos disponíveis. Em alternativa, foi definida uma lógica determinística de procura apoiada num modelo preditivo leve, o que permitiu responder de forma mais pragmática ao objetivo experimental do projeto.

As estratégias de visão egocêntrica, que procuram reproduzir comportamentos humanos de busca de objetos, apresentam igualmente interesse, mas exigem datasets específicos e algoritmos mais complexos do que aqueles que se mostravam exequíveis neste contexto. Da mesma forma, soluções baseadas em múltiplas câmaras ou em técnicas de multi-view stereo revelam um grande potencial para melhorar a estimação de profundidade e a localização precisa de objetos, mas não foram aplicadas porque o Transbot SE está equipado apenas com uma câmara e a adição de novos sensores não se enquadrava no âmbito do trabalho.

3.3.3 Procura baseada em histórico (modelo preditivo)

A procura baseada em histórico tem como objetivo reduzir o tempo de localização de um objeto e minimizar deslocações aleatórias, transformando a informação acumulada em previsões direcionadas. O sistema mantém dois tipos de registos históricos distintos. O primeiro corresponde às deteções realizadas pelo modelo YOLO, onde cada ocorrência é armazenada no servidor com hora, coordenadas e classe do objeto. O segundo é constituído por entradas associadas às interações do utilizador, em que cada pedido manual de procura gera um registo com a hora e o objeto solicitado. No primeiro caso, os dados são simplificados e normalizados: o tempo é discretizado em três faixas; manhã (04:00–11:59), tarde (12:00–19:59) e noite (20:00–03:59) e em ambos os casos cada classe de objeto é representada por um identificador numérico.

Com base nestes dois conjuntos de registos, o processo preditivo está dividido em dois níveis. No primeiro, existe um módulo de agendamento que observa os registos de utilização do robô no cenário passivo. Estes registos formam um padrão de horários em que tipicamente o robô é chamado. Com base nessa informação, o módulo constrói um perfil de probabilidade sobre o ciclo diário e identifica intervalos mais prováveis de necessidade futura. Quando o relógio entra num desses intervalos e o robô está em estado *(IDLE)*, o servidor considera que existe oportunidade para agir proativamente. Assim, o disparo do cenário proativo não acontece em intervalos rígidos, mas sim em janelas temporais que reproduzem os padrões reais de utilização aprendidos a partir dos próprios logs.

No segundo nível, quando um disparo é decidido, o servidor seleciona o modelo correspondente à classe em questão e à faixa temporal corrente. O modelo é um GMM com três componentes, treinado sobre as coordenadas (x, y) das deteções recentes dessa classe na mesma faixa temporal. A escolha de três componentes corresponde diretamente à estratégia de explorar três pontos distintos. As médias das três componentes tornam-se os candidatos de pesquisa e são enviadas ao robô na forma de uma mensagem *search* contendo as coordenadas previstas. A receção desta mensagem por parte do robô enquanto *idling* é interpretada como início do cenário proativo. Nota-se que apesar de o modelo GMM fornecer três pontos relativos a uma classe de objetos, o comportamento do robô durante a procura não é exclusivo e ao deslocar-se para esses pontos qualquer objeto válido detetado em tempo real interrompe a marcha e ativa o ciclo de captura independentemente da classe inicialmente em foco.

Do lado do robô, inicia-se com um varrimento de 360º no ponto inicial $(0,0)$, segue-se deslocação ao primeiro ponto previsto, novo varrimento e, se necessário, repetição para os pontos seguintes. Qualquer deteção durante as deslocações interrompe o percurso e transfere o controlo para a rotina normal de aproximação e captura. Se nenhum ponto resultar, o robô regressa ao ponto inicial $(0,0)$ e o ciclo de procura é encerrado.

3.3.4 Reinforcement Learning

Optou-se por não aplicar reinforcement learning nos comportamentos reativos do robô, onde a operação em tempo real e a previsibilidade são críticas. O sistema é distribuído, com o robô capturando imagens e o servidor a executar a inferência pesada, o que introduz latência e jitter de rede variáveis. Nestas situações de treino, essa variabilidade comprometeria a sincronização de estados, ações e recompensas, dificultando o credit assignment e o reward shaping no horizonte temporal do projeto. A visão limitada do robô, apenas com uma câmara e sem profundidade, restringe ainda a definição de estados suficientemente informativos para uma política estável, tornando o treino em mundo real arriscado, demorado e pouco reprodutível sem simulador fiel ou demonstrações humanas extensas. Por isso, o custo de implementação de RL para comportamentos imediatos supera os benefícios nesta fase.

Em alternativa, privilegiou-se um controlador determinístico simples (alinhamento e aproximação guiados pela posição relativa do objeto na imagem) e uma camada de procura baseada em histórico calculada no servidor como mencionado previamente na secção 3.3.1. A integração de *reinforcement learning* fica, assim, planeada para um possível trabalho futuro em condições mais controladas (treino em simulação com *imitation learning* e *fine-tuning*), idealmente com maior diversidade de sensores.

3.3.5 Simulação e modelação

Inicialmente, estava prevista a utilização de um ambiente de simulação para complementar os testes realizados no robô físico. O Gazebo foi selecionado como plataforma, dada a sua integração nativa com o ROS e a capacidade de fornecer física de alta-fidelidade, simulação de sensores complexos e interação entre múltiplos agentes robóticos. O objetivo seria disponibilizar um espaço controlado para ensaios rápidos e iterativos, facilitando a identificação precoce de falhas e a otimização de algoritmos antes da implementação no robô real. Para esse fim, chegou a ser equacionada a utilização do AWS Robomaker Small House World no Gzweb [68], um ambiente virtual representativo de um cenário doméstico.

No entanto, a utilização prática do Gazebo revelou-se inviável devido a inconsistências na física dos modelos fornecidos pelo fabricante do robô (Yahboom). Apesar de a renderização visual ser correta, a simulação física não correspondia ao comportamento real do Transbot SE, criando um desfasamento significativo entre a simulação e a realidade. Além disso, os controladores específicos do robô não estavam definidos no modelo, o que impossibilitava o acesso e a operação simulada de componentes como as rodas ou a garra. Estas limitações comprometeram a utilidade da plataforma, levando a optar por uma abordagem mais pragmática. Todo o desenvolvimento e testes foram realizados diretamente em ambiente real, eliminando o risco de dependência de uma simulação imprecisa e acelerando o progresso do projeto.

Apesar desta decisão, foi iniciada a preparação de elementos que poderão ser reaproveitados no futuro, caso as limitações identificadas venham a ser resolvidas. Em particular, foram

modelados em Blender dois objetos 3D *low-poly*, correspondentes aos mesmos utilizados nos testes reais (descritos em detalhe na Secção 4.1.1), um cubo de LEGO na Figura 11 e um frasco de medicamentos na Figura 12. Os modelos foram concebidos com um número reduzido de polígonos e texturas simplificadas, de forma a equilibrar desempenho e realismo visual. A sua integração no Gazebo teria permitido realizar testes mais consistentes e variados, como diferentes condições de iluminação, ângulos de captura da câmara e interação com objetos com dimensões distintas.

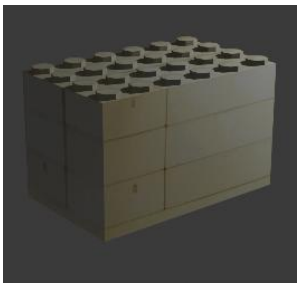

Figura 11 – Imagem do modelo 3D *low-poly* do cubo de LEGO

Figura 12 - Imagem do modelo 3D *low-poly* do frasco de medicamentos

3.4 Cenários de operação

O sistema foi concebido para funcionar em dois modos distintos, o cenário passivo e o cenário proativo. Estes representam diferentes formas de interação entre o robô e o utilizador, variando consoante o grau de iniciativa atribuído ao sistema.
Nesta secção, descrevem-se com maior detalhe as características e funcionamento de cada cenário, clarificando o seu papel na arquitetura global do sistema e o modo como influenciam a lógica de decisão e a gestão do ciclo de procura.

3.4.1 Cenário Passivo

No cenário passivo, o robô responde apenas a comandos diretos do utilizador, executando a tarefa de recolha e entrega de objetos com base em pedidos explícitos. A interação é simples e imediata, ficando a iniciativa sempre do lado do utilizador. Enquanto cumpre estas ordens, o sistema regista informação útil sobre os hábitos do utilizador, como os locais onde cada objeto

é mais frequentemente encontrado e os momentos em que os pedidos ocorrem. Estes registos servem de base para a construção de padrões de rotina que, mais tarde, permitem ao robô acionar autonomamente o cenário proativo e antecipar necessidades futuras.

3.4.2 Cenário Proativo

No cenário proativo, o robô utiliza os dados recolhidos durante o modo passivo para identificar padrões na localização dos objetos e nos momentos em que o utilizador costuma solicitar ajuda. Com base nesses padrões, pode iniciar autonomamente tarefas de procura e entrega, antecipando necessidades sem depender de um pedido explícito. Este comportamento permite ao sistema adaptar-se dinamicamente às rotinas do utilizador e ajustar-se a variações no ambiente ou nos hábitos registados. A abordagem proativa visa reduzir a dependência de comandos diretos e tornar a interação mais natural, reforçando a capacidade do robô de apoiar o utilizador no dia a dia.

3.5 Integração dos Componentes

A arquitetura do sistema foi representada através de um diagrama UML, na Figura 13, que ilustra a interação entre o robô Transbot SE, o servidor de processamento e o módulo auxiliar. O modelo evidencia o fluxo de dados da câmara e telemetria para o servidor, a execução de deteção via YOLO, a previsão de pontos prováveis pelo modelo preditivo e o retorno das instruções de navegação e manipulação para o robô.

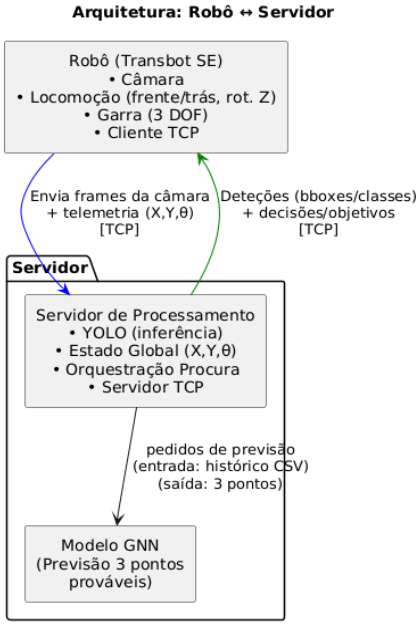

Figura 13 – Diagrama de vista arquitetural personalizada

O comportamento do robô foi estruturado como uma máquina de estados finitos, em que cada fase operacional corresponde a um estado distinto e as transições são desencadeadas pelas condições observadas durante a execução. O ciclo de procura inicia-se no estado de espera *(IDLE)*, em que o robô permanece parado até receber um comando do servidor. O trigger é sempre a receção da mensagem *search* com três pontos de procura, sendo esse envio despoletado pelo utilizador no cenário passivo ou pelo módulo de agendamento no cenário proativo. Em ambos os cenários, a transição é feita para o estado de procura de objeto *(SEARCH_OBJECT)*, no qual o robô realiza um varrimento de 360°. Se não forem detetados objetos, o robô desloca-se sequencialmente para os três pontos de interesse fornecidos pelo modelo preditivo. Assim que surge uma deteção válida, o sistema transita para o estado de aproximação *(APPROACH_OBJECT)*, em que a posição linear e angular é ajustada até que o objeto esteja centrado e suficientemente próximo na imagem da câmara. Uma vez satisfeitas estas condições, o robô entra no estado de captura *(CAPTURE_OBJECT)*, acionando a garra para agarrar o objeto. Segue-se o estado de aproximação ao ponto de entrega (por *default* ponto $(0,0)$)*(APPROACH_POINT)*, em que o robô calcula e executa a trajetória até ao local de entrega definido previamente no mapa, orientando-se com base nas coordenadas e ângulo fornecidos em tempo real pelo servidor. Quando alcança o destino, passa ao estado de libertação *(RELEASE_OBJECT)*, no qual abre a garra e deposita o objeto. Após esta operação, o robô envia a mensagem de *finished* ao servidor e regressa ao estado *(IDLE)*, ficando novamente em espera até à próxima instrução. Os dois diagramas que ilustram de forma mais clara as transições entre estados, correspondentes ao cenário passivo e ao cenário proativo, encontram-se no Apêndice B – Diagramas de transição de estados do robô nos cenários passivo e proativo.

Foram implementados mecanismos de recuperação para garantir robustez em situações de falha. Enquanto no estado *(SEARCH_OBJECT)* se nenhum objeto for encontrado durante o varrimento de 360º sobre um ponto, o robô avança para o ponto seguinte previsto. Caso percorra todos os pontos sem sucesso, regressa à posição inicial $(0,0)$ e encerra o processo. Em situações de perda de ligação ao servidor, interrompe a execução e permanece em estado de espera até à reposição da comunicação.

4 Experimentação

Este capítulo descreve o processo de experimentação seguido para avaliar o desempenho do sistema desenvolvido. A estrutura adotada começa pela caracterização do ambiente de testes e das condições em que os ensaios foram conduzidos, assegurando consistência e repetibilidade. Em seguida, é apresentado o plano de testes que define, de forma organizada, os diferentes componentes a avaliar, desde a deteção visual até à execução de cenários completos, incluindo tanto comportamentos reativos como proativos. Por fim, são discutidas as principais ameaças à validade e limitações inerentes à abordagem, de modo a contextualizar os resultados obtidos e clarificar o alcance das conclusões.

4.1 Ambiente e condições de teste

A experimentação foi conduzida em ambiente controlado, com o objetivo de garantir consistência e repetibilidade entre diferentes sessões de ensaio. O espaço de testes foi preparado para evitar interferências externas, assegurando condições homogéneas de piso, iluminação e ausência de obstáculos. No entanto, reconhece-se a importância de validar a solução em cenários mais heterogéneos, onde variáveis como iluminação irregular, superfícies distintas ou a presença de obstáculos introduzam desafios adicionais, ficando esta avaliação prevista como trabalho futuro (secção 6.2).

4.1.1 Objetos a pegar

Foram definidos dois objetos alvo para a experimentação, um cubo LEGO, mostrado na Figura 14 e um frasco de medicamentos, na Figura 15. A seleção destes elementos teve como critério principal a necessidade de testar a capacidade do robô em distinguir e manipular objetos de natureza distinta, garantindo diversidade tanto em termos geométricos como funcionais.

O cubo LEGO apresenta uma geometria regular com arestas bem definidas e superfícies laterais planas. Trata-se de um objeto com elevada simplicidade estrutural que permite avaliar a eficácia do sistema de deteção em condições favoráveis onde o contorno é nítido e a forma é previsível. Além disso, o cubo funcionou como um *baseline* experimental já que oferece baixa ambiguidade visual e constituiu um ponto de partida adequado para validar a pipeline de deteção, localização e preensão.

Figura 14 – Foto do cubo de LEGO

O frasco de medicamentos por sua vez introduz maior complexidade. A sua forma cilíndrica, a curvatura contínua das superfícies e os reflexos associados ao material diferem significativamente da geometria regular do cubo. A escolha deste objeto também tem uma justificação temática alinhada com o propósito do trabalho centrado em cenários de apoio em ambiente doméstico e de saúde.

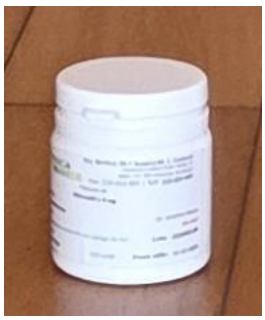

Figura 15 – Foto do frasco de medicamentos

A escolha não foi completamente arbitrária. Os objetos tiveram de respeitar restrições físicas impostas pela garra do robô que abre até aproximadamente 7 cm. Paralelamente foi considerada uma altura mínima de 2 cm para garantir que o objeto fosse visível e detetável pela câmara mesmo a distâncias superiores. Ambos os objetos cumprem estes requisitos. O cubo apresenta dimensões aproximadas de 3,5 cm por 3 cm por 5 cm em altura largura e comprimento. O frasco possui 5,5 cm de altura e 4 cm de diâmetro.

4.1.2 Espaço físico

Os testes foram realizados nas instalações do GECAD no ISEP, mais precisamente no topo do edifício I. A sala utilizada está orientada para sul e possui janelas em toda a extensão dessa fachada, o que garante uma incidência direta de luz natural. Para reduzir variações de luminosidade, todos os ensaios decorreram durante o verão, no mês de julho, entre as 14h e as 18h. Desta forma assegurou-se uma condição de iluminação constante e estável ao longo de todas as sessões.

O piso da sala é em madeira, o que contribuiu para uma boa aderência das rodas do robô e evitou situações de derrapagem durante a locomoção. A área de ensaio correspondeu a um

quadrado de aproximadamente 2 m por 2 m, totalmente livre de obstáculos. Esta configuração funcionou como uma zona desimpedida que permitiu avaliar o desempenho do robô em tarefas de deteção, navegação e manipulação sem recurso a estratégias de evitamento.

Ainda que a sala fosse por vezes ocupada por outras pessoas, os testes foram sempre conduzidos em períodos em que não existia interferência humana no espaço de experimentação. A disposição dos objetos dentro da área variou entre diferentes distâncias e ângulos em relação ao robô. Foram consideradas posições entre cerca de 0,5 m e 1,5 m de distância.

4.2 Plano de testes

Esta secção descreve o plano de testes previsto para avaliar o sistema desenvolvido, apresentando de forma organizada que componentes serão alvo de testes e com que finalidade. Em primeiro lugar estão definidos testes relativos ao modelo de deteção YOLOv5, recorrendo a imagens previamente captadas e usadas para aferir o seu desempenho. Seguem-se testes planeados para medir a taxa de sucesso da preensão e o tempo médio entre a deteção do objeto e o acionamento da garra. Numa terceira etapa, o plano de testes tem como objetivo verificar se o robô consegue manter precisão de posicionamento ao executar uma sequência de movimentos de procura. Finalmente, serão descritos testes ao modo proativo, centrados na geração de pontos prováveis através de GMM e no funcionamento do módulo de agendamento, recorrendo a dados sintéticos que simulam padrões de utilização.

4.2.1 Avaliação da deteção de objetos

A avaliação da componente de deteção foi realizada em regime offline, analisando o desempenho do modelo YOLOv5 aplicado a imagens previamente recolhidas para este fim. Esta opção permitiu verificar a capacidade de generalização do modelo e obter métricas de desempenho antes da sua integração no robô, sem depender de ensaios em tempo real durante esta fase inicial.

O modelo foi treinado com um conjunto de imagens que inclui tanto capturas feitas diretamente através da câmara do robô como imagens adicionais recolhidas em cenários distintos. Estas imagens apresentam variações nas posições relativas dos objetos, diferentes intensidades de luz, ângulos de observação e fundos. Algumas foram recolhidas em ambiente de teste controlado e outras em condições mais heterogéneas, com o objetivo de garantir diversidade visual durante o treino.

A validação da componente de deteção será feita com base nas métricas padrão da área, nomeadamente precisão, *recall* e F1-score por classe, bem como a taxa de falsos positivos e negativos. Estas métricas permitirão contextualizar o comportamento do modelo antes da sua integração com os restantes módulos do sistema.

4.2.2 Avaliação da função de preensão

Uma das componentes centrais a validar no sistema desenvolvido é a capacidade do robô em agarrar os objetos com sucesso após a deteção visual. Esta fase é especialmente crítica por depender diretamente da precisão do alinhamento, da resposta motora da garra e das dimensões físicas dos objetos relativamente à abertura máxima permitida.

O plano de testes nesta secção prevê a realização de ensaios repetidos em que o robô, após detetar um dos objetos, tenta alinhar-se e executar a preensão. O objetivo é avaliar de forma sistemática a taxa de sucesso da preensão para cada um dos dois objetos definidos, bem como o tempo médio decorrido entre a deteção do objeto e o acionamento da garra. Estes testes permitem verificar até que ponto o sistema consegue cumprir de forma fiável um ciclo simples de perceção e atuação.

Cada tentativa será registada individualmente, incluindo o tipo de objeto, sucesso ou insucesso da preensão, e a duração do intervalo entre a deteção validada e a ordem de fechar a garra. Esta métrica temporal permite ainda avaliar se existem padrões de atraso ou inconsistência que possam indicar problemas no encaminhamento da decisão ou no controlo motor. Com estes dados será possível estimar a robustez da manipulação e identificar potenciais limitações associadas a cada tipo de objeto.

4.2.3 Avaliação do desvio após rotina de locomoção

Esta parte do plano de testes tem como objetivo verificar se o robô consegue manter precisão de posicionamento quando executa uma sequência de movimentos de procura. Ao longo do teste o sistema foi colocado a realizar trajetórias compostas por várias rotações e deslocamentos, simulando um ciclo típico de procura de objetos em que há necessidade de alternar entre diferentes direções e pontos de referência. Ao terminar o percurso, espera-se que o robô retorne à posição inicial, o que permite avaliar o grau de desvio acumulado durante toda a execução. Importa referir que não foi utilizado qualquer método de planeamento de caminhos, como o algoritmo A*, uma vez que o robô não dispõe de sensores adequados para deteção de obstáculos. Com apenas uma câmara RGB (sem ser uma *depth camera*) ou LIDAR, a estratégia adotada consistiu em deslocações diretas até pontos predefinidos no espaço, sem desvio intermédio. Esta abordagem foi viável porque os testes foram realizados em ambiente controlado e sem obstáculos, evitando riscos de colisão e permitindo focar a análise na precisão de posicionamento.

A análise irá incidir sobre a diferença entre o ponto de partida e o ponto de chegada. Essa diferença, designada desvio, reflete o impacto dos erros acumulados na navegação, tanto em rotações como em deslocamentos lineares. A medição do desvio permitirá avaliar a estabilidade do sistema de movimentação e a fidelidade do registo interno de coordenadas, servindo como indicador da robustez do robô na execução de trajetórias completas.

4.2.4 Avaliação do modelo preditivo

Embora o verdadeiro objetivo final deste módulo seja melhorar a eficácia da procura com base em dados históricos, não foi realizada uma comparação quantitativa direta entre o desempenho com e sem histórico. Essa opção justifica-se pelo facto de o ambiente de testes ser relativamente reduzido e controlado, o que diminui o impacto prático da utilização de registos anteriores, e pela inexistência de dados reais recolhidos em interações regulares com utilizadores humanos que operem o robô de forma contínua. No entanto, esta limitação abre uma linha clara para trabalhos futuros, num contexto mais alargado e dinâmico. No futuro, este modulo permitirá avaliar de forma objetiva o contributo da integração de dados históricos para a eficiência global do sistema.

Assim, nesta fase do projeto, os testes estarão centrados na validação isolada do modelo GMM responsável pela previsão dos pontos prováveis e na avaliação do módulo de agendamento que determina os horários mais prováveis para iniciar uma procura. A cada vez que um utilizador desencadeasse uma procura manual, o sistema registaria a hora e a classe do objeto, e esses dados alimentariam o perfil de agendamento. Com base nesse processo, o módulo construiria um perfil de probabilidade sobre o ciclo diário e identificaria intervalos mais prováveis de necessidade futura.

Devido ao caráter prototípico do sistema e à ausência de utilização contínua por pessoas reais, todos os dados utilizados nestes testes serão gerados sinteticamente com base em padrões simulados de uso.

De forma a ilustrar este processo, apresentam-se de seguida exemplos de ambos os tipos de dados sintéticos gerados. A Tabela 7 apresenta exemplos de pontos de procura gerados sinteticamente. Cada registo corresponde a uma combinação entre o período do dia, a posição espacial (coordenadas X e Y) e o objeto associado. Estes dados simulam a distribuição temporal e espacial de ocorrências, permitindo que o modelo de previsão identifique padrões de procura plausíveis a partir de diferentes contextos.

Tabela 7 – Exemplo de pontos de procura gerados sinteticamente

Período do dia	**Coordenada X**	**Coordenada Y**	**Objeto identificado**
Manhã	1.0	1.2	Cubo de LEGO
Tarde	0.9	1.0	Frasco de medicamentos
Noite	1.1	1.3	Cubo de LEGO

A Tabela 8 mostra exemplos de registos de agendamento sintéticos. Cada entrada indica a data, a hora e o objeto solicitado, simulando a forma como o robô poderia receber pedidos em

cenários de utilização contínua. Estes dados permitem avaliar a consistência do módulo de agendamento quando alimentado com padrões artificiais de uso.

Tabela 8 – Exemplo de registos de agendamento gerados sinteticamente

Data	**Hora**	**Objeto solicitado**
2025-07-01	08:53	Cubo de LEGO
2025-07-01	11:55	Frasco de medicamentos
2025-07-01	21:10	Cubo de LEGO

Os testes servirão sobretudo para validar se o modelo GMM é capaz de gerar pontos de procura que façam sentido em relação à distribuição temporal e espacial definida nos dados sintéticos. A avaliação incidirá sobre a coerência estatística e lógica desses pontos, verificando se refletem corretamente os padrões artificiais introduzidos. Desta forma, garante-se que a integração entre geração de dados, previsão e agendamento funciona de forma consistente, preparando o sistema para fases posteriores em que será alimentado com registos reais de utilização.

4.3 Ameaças à validade e limitações

A experimentação foi realizada em ambiente controlado, o que significa que alguns fatores presentes em cenários reais não estão representados. O espaço de testes foi limitado a uma área reduzida, livre de obstáculos e com iluminação relativamente estável. Esta configuração simplifica a avaliação, mas também restringe a generalização dos resultados, já que em ambientes domésticos típicos existe maior variabilidade, presença de móveis, objetos aleatórios e alterações súbitas de luz.

Outro aspeto a considerar é que apenas dois objetos foram utilizados, um cubo LEGO e um frasco de medicamentos. Embora tenham sido escolhidos de forma a representar geometrias distintas, a variedade de objetos é ainda pequena, e os resultados obtidos podem não refletir o comportamento do sistema perante outros formatos, dimensões ou materiais.

O robô não dispõe de mecanismos de evitamento de obstáculos. Como consequência, todos os testes foram feitos em zonas desimpedidas e a avaliação não cobre situações em que seja necessário desviar-se de elementos inesperados no caminho, conforme já discutido na secção 4.2.3. Esta limitação foi considerada durante o planeamento, e avaliou-se a hipótese de recorrer a um modelo de deteção de obstáculos em tempo real. Contudo, mesmo que o sistema dispusesse de sensores adicionais, como LIDAR ou câmaras de profundidade, a execução local não seria viável devido às restrições de processamento do Raspberry Pi 5. A alternativa de realizar essa verificação externamente, no servidor, foi igualmente descartada, já que a latência introduzida impediria uma reação segura. Assim, optou-se por manter os testes em ambiente controlado, aceitando esta limitação no contexto experimental do protótipo.

Em termos operacionais, a autonomia energética do robô constitui também uma limitação prática. O sistema apresenta instabilidade quando a bateria está abaixo de aproximadamente 50% de carga, existindo risco de desligamento súbito devido a baixa tensão. Para garantir consistência, todos os ensaios foram realizados com o robô acima deste nível de carga, assegurando que as falhas observadas não estavam relacionadas com limitações da bateria.

Por fim, os testes realizados podem ser considerados apenas uma avaliação inicial exploratória, tanto pela simplicidade do robô como pela limitação de tempo disponível para experimentação. Ainda assim, procurou-se estruturar o plano de forma a extrair o máximo de informação relevante, fornecendo uma avaliação rigorosa dentro das condições possíveis. O resultado é uma caracterização realista do protótipo e das suas capacidades no estado atual de desenvolvimento.

5 Resultados e Discussão

Este capítulo apresenta os resultados obtidos durante a experimentação e analisa criticamente o desempenho do sistema. A estrutura segue o plano de testes definido no capítulo anterior (capítulo 4.2), cobrindo desde a componente de deteção visual até à execução de cenários completos e avaliação do modo proativo. Os resultados são discutidos à luz das métricas recolhidas, salientando tanto os pontos fortes como as limitações observadas, de forma a enquadrar o valor do protótipo no estado atual de desenvolvimento.

5.1 Resultados da deteção de objetos

A primeira componente a ser avaliada corresponde ao modelo de visão computacional responsável pela deteção dos objetos. Esta etapa é crítica porque sustenta todo o pipeline de funcionamento do robô, condicionando diretamente o sucesso da preensão e a execução de cenários completos. Para além de validar se o modelo YOLOv5m ajustado ao dataset consegue distinguir de forma fiável entre o cubo LEGO e o frasco de medicamentos, pretende-se também analisar as suas características de treino, equilíbrio das classes e limitações inerentes ao dataset utilizado.

5.1.1 Caracterização do dataset

O modelo de deteção foi ajustado recorrendo a *transfer learning* sobre a versão pré-treinada YOLOv5m, uma abordagem comum quando se dispõe de conjuntos de dados reduzidos. No total foram utilizadas 295 imagens, das quais 260 para treino e 35 para validação. A divisão foi feita da seguinte forma:

- Treino com 130 imagens contendo apenas o cubo LEGO, 115 imagens contendo apenas o frasco de medicamentos e 15 imagens com ambos os objetos.
- Validação com 20 imagens do cubo, 10 do frasco e 5 contendo os dois em simultâneo.

Apesar do número relativamente baixo de imagens, o dataset foi construído com alguma diversidade intra-classe, incluindo variações de iluminação, fundos e ângulos de observação. Esta diversidade contribui para reduzir o risco de *overfitting* e aumentar a robustez do modelo. Ainda assim, apenas uma fração reduzida do dataset inclui ambos os objetos em simultâneo. Esta separação garante que a validação cobre não apenas exemplos isolados de cada classe, mas também situações em que os dois objetos surgem em conjunto, o que é relevante para aferir a capacidade de generalização do modelo a cenários mais realistas. No contexto do protótipo esta limitação foi considerada aceitável, já que o robô manipula apenas um objeto de cada vez, mas para aplicações mais próximas do uso prático será essencial expandir o dataset com mais exemplos de coocorrência, assegurando desempenho fiável mesmo quando múltiplos alvos estão presentes no campo de visão. Ainda assim, a escala limitada do dataset permanece

uma restrição, tornando a escolha de *transfer learning* essencial para viabilizar o desempenho desejado no contexto de um protótipo.

A Figura 16 mostra a análise da distribuição de classes e anotações (*labels*). O gráfico de barras indica que o dataset se encontra relativamente equilibrado entre as duas classes, embora com uma ligeira predominância do cubo em relação ao frasco. Este equilíbrio é positivo, pois reduz o risco de enviesamento para apenas uma classe. No entanto, o número total de instâncias continua baixo, com aproximadamente 145 anotações para o cubo e 130 para o frasco, o que pode limitar a capacidade do modelo generalizar para condições muito diferentes das observadas no treino.

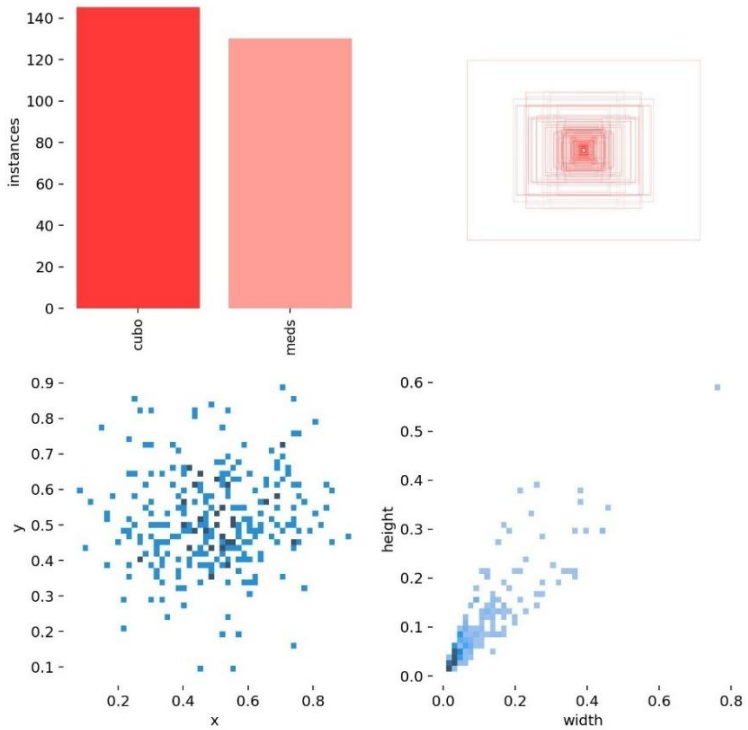

Figura 16 – Distribuição das instâncias e bounding boxes no dataset do YOLOv5m

A distribuição espacial dos bounding boxes revela que a maioria das anotações se encontra em posições centrais da imagem e com dimensões reduzidas. Assim, sugere que o modelo foi principalmente exposto a objetos pequenos, próximos do centro e em escalas semelhantes. A dispersão de largura e altura confirma essa tendência, mostrando concentração em tamanhos próximos e poucas amostras em escalas maiores. Em termos práticos, isto significa que o modelo está otimizado para deteções em condições semelhantes às do dataset, mas poderá apresentar dificuldades em identificar objetos fora do centro, em ângulos mais extremos ou com maior variação de tamanho. Ainda assim, a inclusão de imagens recolhidas diretamente a partir da câmara do robô no ambiente de teste contribuiu para reduzir esse risco, garantindo que o modelo estivesse exposto a condições muito próximas das que encontra durante a experimentação. Desta forma, embora o *fine-tuning* realizado com este conjunto de dados seja

funcional e suficiente para suportar os testes planeados no protótipo, os resultados devem ser interpretados tendo em conta as limitações de diversidade e escala do dataset.

5.1.2 Evolução do treino e convergência

A Figura 17 apresenta a evolução das principais perdas e métricas ao longo das 100 épocas de treino. No geral, observa-se um comportamento consistente entre treino e validação, sugerindo que o modelo se adaptou bem ao conjunto de dados.

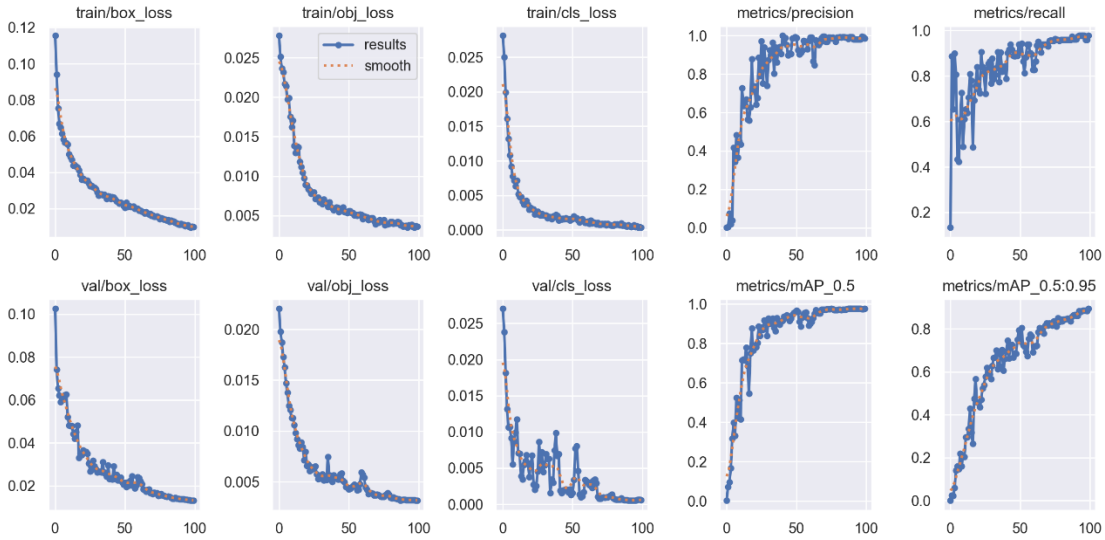

Figura 17 – Evolução do fine-tuning do modelo YOLOv5m ao longo de 100 épocas

Nas perdas de treino, tanto a regressão das *bounding boxes* (*train/box_loss*) como a *objectness* (*train/obj_loss*) e a classificação (*train/cls_loss*) apresentam a tendência típica. Todas decrescem rapidamente nas primeiras épocas e depois estabilizam de forma gradual em valores bastante baixos. A perda de *bounding box* reduz-se de cerca de 0.12 para aproximadamente 0.01, a de *objectness* de aproximadamente 0.025 para valores próximos de 0.002 e a de classificação aproxima-se de zero, o que indica boa correspondência entre as predições e os *labels* fornecidos.

As perdas de validação acompanham este comportamento, ainda que com maior variabilidade, sobretudo na perda de classificação (*val/cls_loss*), que oscila de forma visível até cerca da época 70 antes de estabilizar. Essa oscilação é expectável dada a dimensão reduzida do conjunto de validação (35 imagens) e o menor número de exemplos com ambas as classes. Apesar disso, as curvas de validação convergem para valores próximos dos de treino, o que sugere que o modelo não sofreu *overfitting* severo.

Nas métricas, observa-se uma subida rápida e estável. A precisão cresce logo no início para valores acima de 0.9 e mantém-se estável. A revocação evolui de forma mais gradual, atingindo valores próximos de 0.97 ao final do treino. O mAP@0.5 estabiliza perto de 0.97, enquanto o

mAP@0.5:0.95 converge para aproximadamente 0.89. Estes resultados confirmam que o modelo atinge elevado equilíbrio entre deteções corretas e minimização de falsos positivos.

Apesar dos valores muito elevados alcançados, é necessário interpretar os resultados com alguma prudência. O tamanho reduzido do dataset e as condições relativamente controladas das imagens podem facilitar a tarefa de deteção, o que ajuda a explicar métricas próximas do ideal. Assim, embora não haja indícios fortes de *overfitting* nas curvas, permanece a incerteza sobre a capacidade de generalização do modelo em contextos não representados no treino.

5.1.3 Desempenho final do modelo

O desempenho final do modelo YOLOv5m foi avaliado com base no conjunto de validação de 35 imagens. A análise dos resultados mostra que o modelo apresenta níveis elevados de precisão e recall em ambas as classes, alcançando métricas próximas do ideal.

A matriz de confusão na Figura 18, revela que a taxa de acerto é consistente para os dois objetos, com o frasco de medicamentos a atingir um recall praticamente perfeito (1.00) e o cubo LEGO a apresentar igualmente resultados muito elevados (0.96). Os erros registados traduzem-se em poucos falsos negativos na classe cubo, o que sugere que em alguns cenários específicos o modelo pode não identificar o objeto quando este se encontra em condições menos favoráveis, como ângulos mais extremos ou fundos semelhantes.

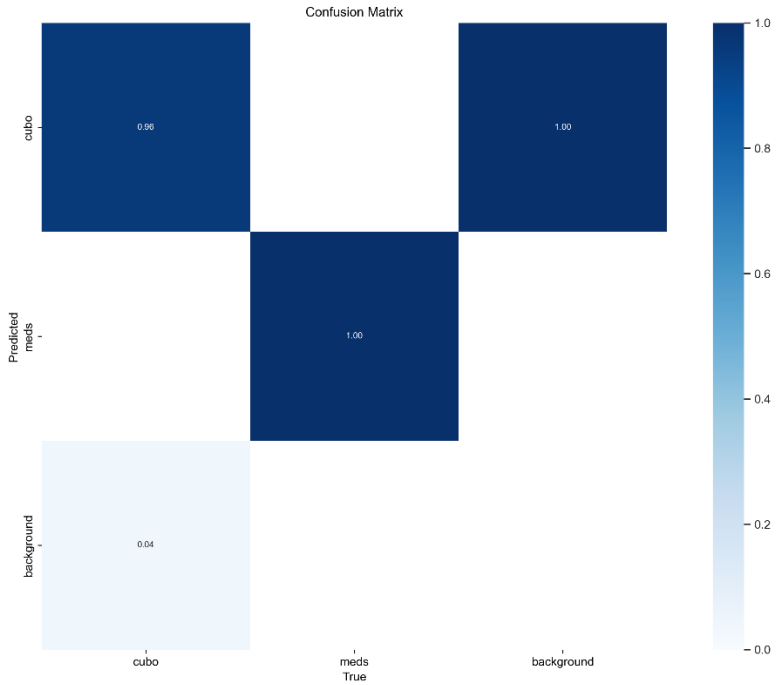

Figura 18 – Matriz de confusão do modelo YOLOv5m

A curva Precision–Recall na Figura 19, confirma este desempenho elevado, apresentando valores de mAP@0.5 de 0.977. Estes indicadores demonstram que o modelo mantém um equilíbrio robusto entre precisão e recall, mesmo em diferentes limiares de decisão.

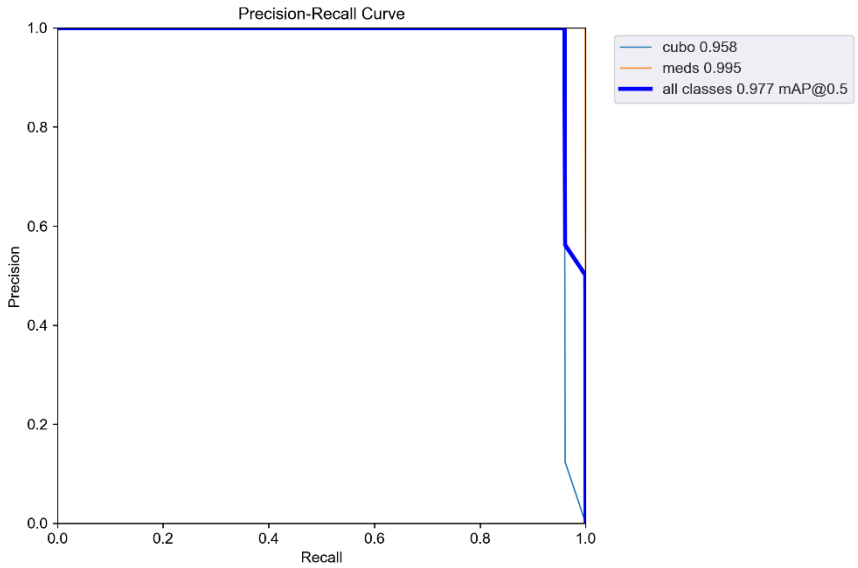

Figura 19 – Curva Precision–Recall do modelo YOLOv5m

Por fim, a curva F1–Confidence na Figura 20, mostra que o F1-score global atinge 0.98 e se mantém praticamente estável para valores de confiança entre 0.5 e 0.8. Isto significa que o modelo é pouco sensível a variações no threshold adotado, o que lhe confere robustez operacional, reduzindo a necessidade de ajustes finos de parametrização em cenários práticos.

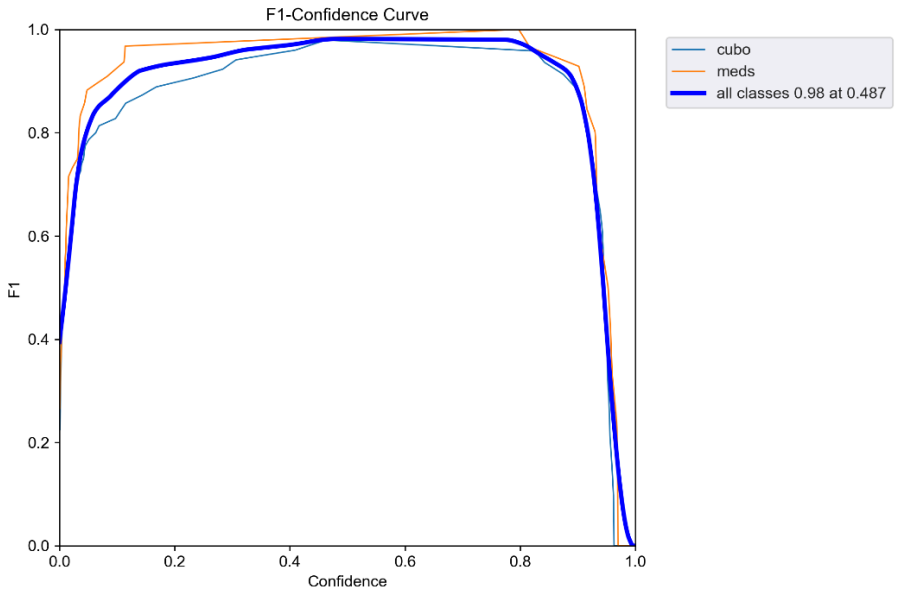

Figura 20 – Curva F1–Confidence do modelo YOLOv5m

Os resultados sugerem um desempenho muito elevado do modelo, sobretudo tendo em conta a dimensão relativamente pequena do conjunto de treino. Este nível de eficácia pode estar relacionado com o uso de *transfer learning*. Ainda assim, é importante notar que estes fatores ajudam a contextualizar os valores quase ideais obtidos, que à primeira vista podem parecer excessivamente elevados.

5.2 Resultados da função de preensão

Uma das componentes centrais validadas neste trabalho foi a capacidade do robô agarrar os objetos com sucesso após a sua deteção visual. Para esta análise foram realizados no total 60 ensaios de preensão, repartidos de forma equilibrada entre os dois objetos definidos. Assim, 30 tentativas foram realizadas sobre o cubo LEGO e outras 30 sobre o frasco de medicamentos.

Entre a realização de cada teste foi necessário ajustar manualmente os valores registados no ficheiro JSON responsável pela calibração entre a câmara e a garra. Como a câmara se desloca ligeiramente durante a movimentação do robô, os valores de calibração sofrem pequenas alterações e, se não forem corrigidos, a taxa de sucesso desce rapidamente. Esta necessidade de ajuste entre ensaios está associada à própria natureza experimental do protótipo, agravada pelo uso de uma câmara montada num suporte com 2 DOF (*pan* e *tilt*), que não está rigidamente fixa ao robô e tende a sofrer pequenos desvios, perdendo calibração devido às limitações de fabrico.

A Figura 21 apresenta a taxa de sucesso obtida para cada um dos dois objetos. Observa-se que o frasco de medicamentos foi agarrado com maior fiabilidade, registando uma taxa de 86,7%, enquanto o cubo LEGO apresentou uma taxa de 70,0%. Uma possível explicação para esta diferença reside no formato cilíndrico do frasco, que pode oferecer maior tolerância ao posicionamento da garra no momento da preensão. Em contraste, o cubo exige um alinhamento mais rigoroso devido às suas arestas definidas. Importa ainda referir que em todos os ensaios os objetos foram colocados em pé, nunca deitados ou invertidos, de forma a manter a consistência entre as tentativas.

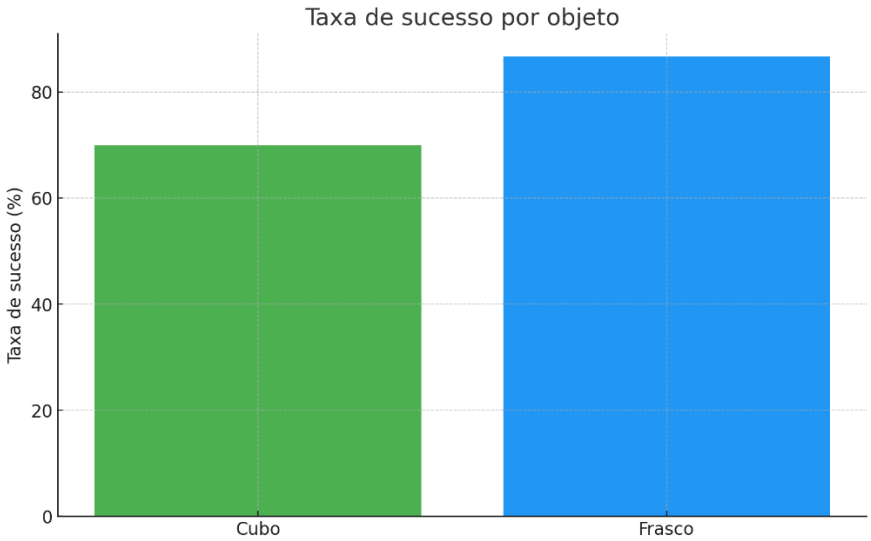

Figura 21 – Taxa de sucesso por objeto

Durante estes ensaios o robô foi sempre colocado aproximadamente na borda da área de teste. Os objetos foram posicionados de forma incrementalmente mais distante ao longo das 30 tentativas de cada conjunto; ou seja, no primeiro ensaio encontravam-se próximos do robô e, progressivamente, foram colocados mais afastados até à borda oposta da área de 2 m por 2 m. Como o objetivo principal era avaliar a taxa de sucesso da preensão e o respetivo tempo de execução, os objetos foram sempre colocados inicialmente dentro do campo de visão da câmara do robô, assegurando a sua deteção quase imediata. Para contabilizar o tempo de preensão foi considerado desde o instante da primeira deteção do objeto até à conclusão do movimento de agarrar.

Um aspeto que teria acrescentado maior detalhe aos resultados seria o registo preciso da posição relativa de cada objeto dentro da área de teste. Essa informação teria permitido relacionar os tempos e taxas de sucesso com a distância e o ângulo de colocação. No entanto, esses dados não foram guardados durante a execução, pelo que a análise centra-se apenas nos valores globais de desempenho.

A Figura 22 apresenta a distribuição dos tempos de preensão obtidos para os dois objetos. No caso do cubo LEGO, o tempo médio foi de 35,5s, valor mínimo de 19s e máximo de 51s. Já para o frasco de medicamentos o tempo médio foi ligeiramente superior, 38,7s, mínimo de 20s e máximo de 56s. Esta diferença pode estar relacionada com a geometria dos objetos e com pequenas variações no alinhamento necessário para a preensão.

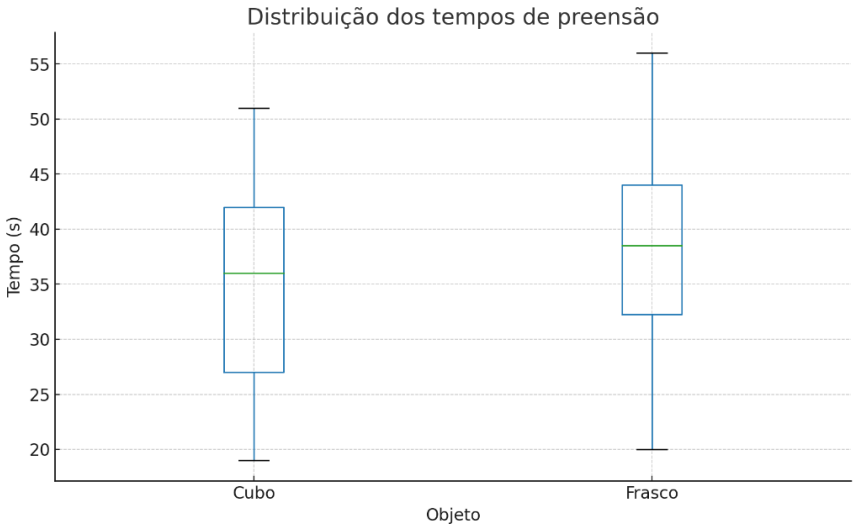

Figura 22 – Distribuição dos tempos de preensão

A Figura 23 mostra a evolução do tempo de preensão ao longo das 30 tentativas realizadas para cada objeto. Nos casos em que a preensão falhou, o tempo não foi registado, pelo que apenas os ensaios concluídos com sucesso estão representados.

Observa-se uma tendência de crescimento gradual do tempo à medida que os ensaios avançam, o que está em linha com o facto de os objetos terem sido colocados progressivamente mais distantes do robô, culminando no último teste junto à borda da área de ensaio. A análise da distribuição dos resultados não sugere uma correlação clara entre a distância e a probabilidade de falha de preensão. Tanto em distâncias curtas como longas ocorreram casos de sucesso e insucesso, o que indica que outros fatores, como a precisão do alinhamento ou pequenas variações nos ajustes de calibração, podem ter tido maior influência no resultado de sucesso da preensão.

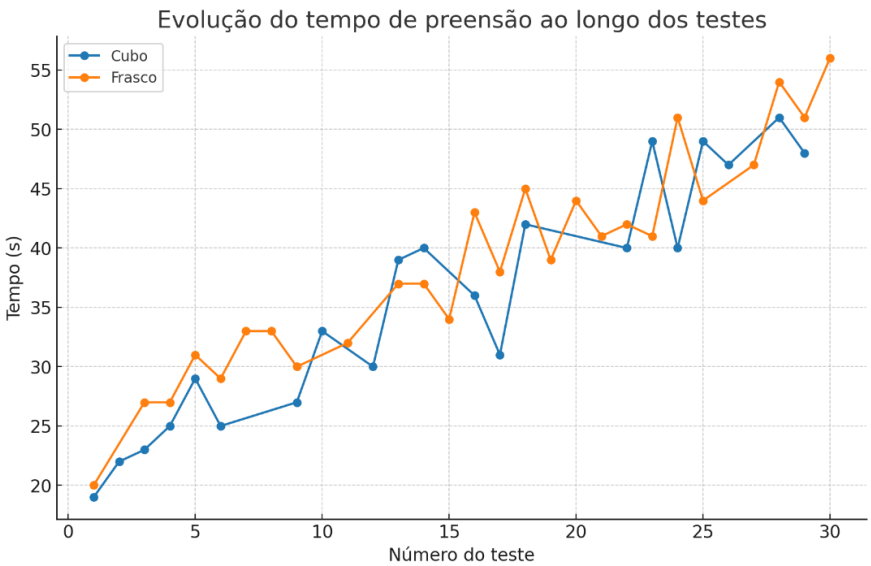

Figura 23 – Evolução do tempo de preensão ao longo dos testes

Durante os testes foi também validada a funcionalidade responsável por garantir que o robô apenas tenta agarrar o objeto mais próximo no campo de visão. Esta funcionalidade foi testada e verificou-se que funciona corretamente, mas não será alvo de uma análise mais detalhada. Este mecanismo é relativamente direto. No momento da inferência, o servidor calcula a média entre o ponto mais alto e o ponto mais baixo da deteção YOLO de cada objeto na imagem, determinando assim o ponto central. Entre todos os objetos detetados, é selecionado aquele cujo ponto central se encontra mais próximo da parte inferior da imagem, o que corresponde àquele que está mais próximo do robô. Apenas esse objeto é então comunicado ao robô, de modo que, do ponto de vista do sistema de controlo, nunca exista mais do que um alvo válido por imagem.

Pode-se argumentar que este método favorece objetos de menor altura, uma vez que o ponto médio tende a ficar mais baixo relativamente a objetos mais altos. Contudo, essa diferença só se manifesta de forma significativa quando os objetos se encontram a distâncias semelhantes do robô. Na prática, para a configuração utilizada nos testes, esta abordagem simples mostrou-se suficiente para distinguir de forma eficaz o alvo mais próximo sem necessidade de cálculos adicionais.

5.3 Resultados do desvio após rotina de locomoção

Para avaliar a precisão do sistema de locomoção foram realizados testes em que o robô executou uma trajetória de procura definida manualmente através de pontos *hard coded*. A sequência escolhida correspondeu às coordenadas (0,1), (1,1) e (0,0), formando um percurso triangular. Em cada ponto intermédio o robô executava primeiro uma rotação completa de 360°, simulando a rotina de varrimento visual usada no módulo de procura, antes de avançar para o ponto seguinte. Desta forma, o percurso incluía tanto deslocamentos lineares como rotações sucessivas, aproximando-se do comportamento esperado num cenário real de exploração. No final do ciclo, o robô regressava ao ponto de partida, permitindo comparar a posição inicial com a posição final registada e, assim, calcular o desvio acumulado. A Figura 24 ilustra graficamente a trajetória triangular e as rotações associadas.

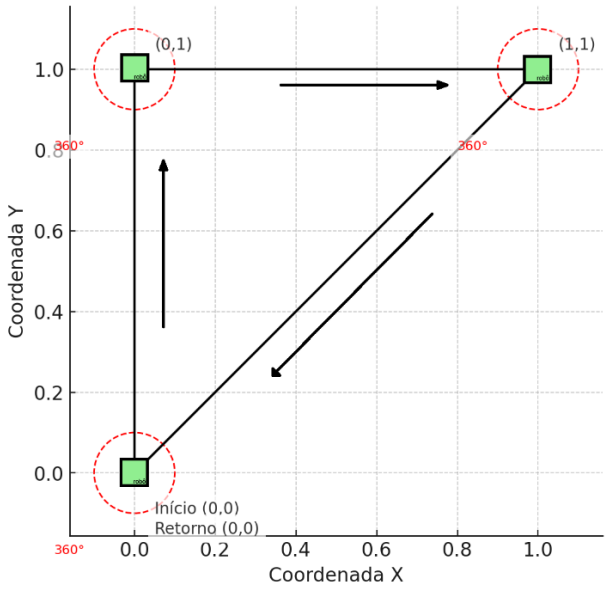

Figura 24 – Trajetória triangular definida para avaliação da precisão de locomoção

Foram conduzidos dez ensaios repetidos em condições semelhantes para garantir consistência dos resultados. A análise dos registos mostra que o desvio entre o ponto inicial e o ponto final se manteve sempre dentro da mesma ordem de grandeza, com uma média de 41,2 cm. A Tabela 9 apresenta os valores de desvio registados nos dez ensaios realizados, que serviram de base ao cálculo da média referida.

Tabela 9 – Resultados dos ensaios de desvio acumulado após trajetória de procura

Ensaio	**Desvio acumulado (cm)**
1	28
2	50
3	48
4	44
5	40
6	34
7	46
8	35
9	44
10	43

Este valor é expressivo tendo em conta que a área de ensaio total correspondia apenas a 2 m por 2 m, o que significa que o robô acumulou um erro equivalente a uma fração significativa da própria área de testes.

A presença de um desvio desta magnitude sugere que a integração entre rotação e translação não é suficientemente precisa para assegurar consistência espacial ao longo de trajetórias compostas. É provável que pequenos erros em cada deslocamento linear e cada rotação se acumulem, resultando num afastamento cada vez maior do ponto esperado. Como o sistema

depende apenas da contagem dos movimentos emitidos e não dispõe de sensores adicionais para correção contínua, não existe mecanismo para compensar esses erros.

Estes resultados indicam que, apesar de o robô ser capaz de executar a sequência de procura de forma funcional, o seu posicionamento absoluto não é fiável quando os movimentos se prolongam em ciclos mais longos. Na prática, isto limita a utilização do protótipo em cenários onde a precisão de localização é essencial. Ainda assim, os testes foram fundamentais para evidenciar este comportamento e confirmar que o sistema atual apresenta desvios relevantes, fornecendo uma base clara para justificar melhorias futuras em algoritmos de navegação e em mecanismos de correção de posição.

5.4 Resultados do modelo preditivo e módulo agendamento

Como já referido na secção 4.2.4, a avaliação do modelo preditivo foi realizada exclusivamente com recurso a dados sintéticos, dado tratar-se de um protótipo sem histórico de utilização real. Nesta secção apresentam-se os resultados obtidos com o GMM aplicado a duas vertentes distintas, a previsão espacial de pontos prováveis de procura e a previsão temporal associada ao módulo de agendamento. Em ambos os casos foram gerados conjuntos de dados artificiais que simulam padrões de utilização, permitindo avaliar a coerência estatística e lógica das previsões e verificar se o sistema está preparado para fases futuras em que será alimentado com registos reais.

5.4.1 Resultados do GMM e pontos previstos

Como referido na secção 3.3.3, a informação bruta recolhida pelo robô relativamente à posição dos objetos seria tipicamente posteriormente organizada no formato *(time_zone, x, y, label_id)*, representando a zona temporal do dia, as coordenadas de deteção e o identificador do objeto. O mesmo formato foi seguido para a criação do dataset artificial que serviu de base à avaliação do modelo de GMM.

O objetivo definido foi o de, dado um par *(time_zone, objeto)*, prever as três localizações mais prováveis *(x, y)*. Para tal, foi necessário recorrer a um modelo capaz de aprender distribuições espaciais multimodais. Para cada combinação de zona temporal e classe de objeto foi treinado um GMM com três componentes, usando apenas as coordenadas espaciais. Em fase de inferência, o modelo retorna as três médias dos componentes mais prováveis, ordenadas de acordo com o peso atribuído a cada componente. Todos os modelos foram organizados num dicionário e armazenados em um ficheiro (*gmm_models.pkl*) para utilização posterior.

A simulação foi construída de forma a representar três zonas temporais do dia, manhã, tarde e noite, e os dois tipos de objetos, o cubo de LEGO e o frasco de medicamentos. O objetivo foi verificar se o GMM conseguiria identificar padrões espaciais coerentes em função do momento do dia e da classe do objeto. Para facilitar a análise, as previsões foram representadas em gráficos bidimensionais com perspetiva top-down. Em cada visualização, círculos vermelhos

correspondem às médias previstas para o frasco de medicamentos e quadrados azuis às médias previstas para o cubo, sendo que cada marcador indica uma das três localizações mais prováveis estimadas pelo modelo. Estes gráficos permitem observar de forma intuitiva como a distribuição dos objetos varia ao longo do dia.

5.4.1.1 Período da Manhã

No período da manhã (time_zone = 0) simulou-se um comportamento regular, em que os objetos tendem a surgir de forma estável em torno de localizações fixas. Para o cubo foram definidos três agrupamentos principais, centrados nos pontos (1,1), (2,2) e (3,3). Cada agrupamento gerou 30 amostras distribuídas segundo uma Gaussiana com dispersão reduzida ($\sigma \approx 0.3$), produzindo ao todo 90 entradas para esta classe. O frasco de medicamentos seguiu o mesmo princípio, mas com agrupamentos localizados em (−1,−1), (0,−1) e (1,−1), também com 30 amostras por cluster e um total de 90 entradas. No conjunto, a manhã resultou em 180 entradas organizadas em padrões regulares e bem definidos.

A Figura 25 mostra o resultado do GMM aplicado a este cenário. Os quadrados azuis correspondem às três localizações mais prováveis para o cubo e os círculos vermelhos às três localizações mais prováveis para o frasco de medicamentos. A visualização confirma que os padrões da manhã são bem separados e estáveis, refletindo a previsibilidade esperada neste período.

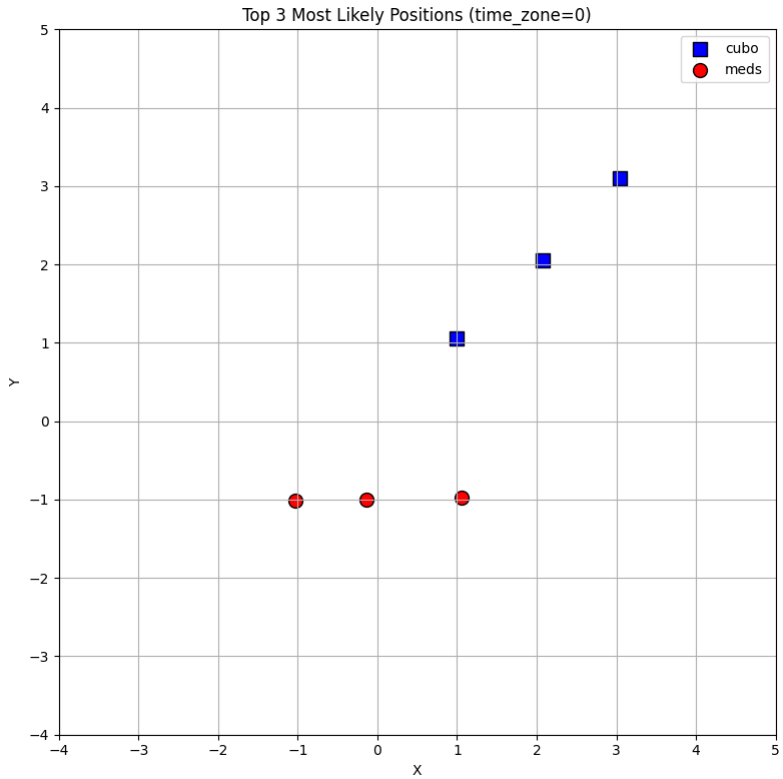

Figura 25 – Posições mais prováveis previstas pelo GMM no período da manhã

5.4.1.2 Período da Tarde

No período da tarde (time_zone = 1) procurou-se simular uma situação de maior imprevisibilidade, em que os objetos deixam de seguir padrões regulares de localização. Para isso foram removidos os agrupamentos estáveis e as coordenadas passaram a ser geradas de forma uniforme em toda a região [−4,4]. Cada classe recebeu 30 amostras, resultando num total de 60 entradas.

A Figura 26 apresenta os três pontos mais prováveis previstos pelo GMM para este cenário. Ao contrário do período da manhã, aqui não surgem clusters bem definidos, mas sim distribuições dispersas e pouco estruturadas. Esta aleatoriedade foi introduzida deliberadamente na geração dos dados, de forma a simular a ausência completa de padrões regulares. O resultado mostra que, mesmo nesse cenário, o GMM é capaz de produzir três pontos de previsão. Contudo, esses pontos apresentam menor consistência, tanto por estarem espacialmente mais espalhados como por terem probabilidades semelhantes entre si, o que dificulta a identificação de locais preferenciais claros. Este comportamento ilustra a aplicabilidade do modelo mesmo quando não existem hábitos estáveis nos registos.

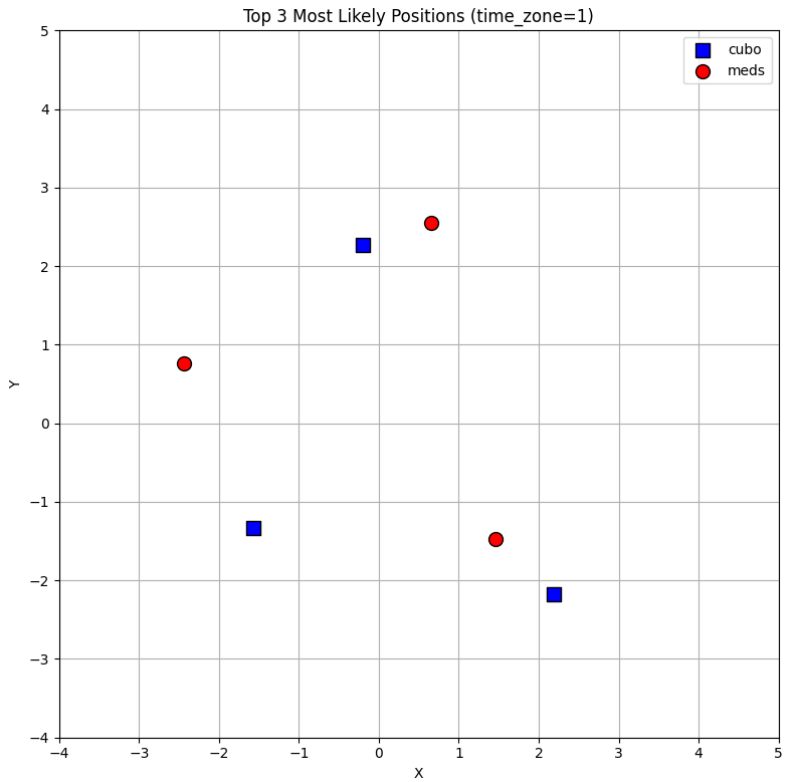

Figura 26 – Posições mais prováveis previstas pelo GMM no período da tarde

5.4.1.3 Período da Noite

A simulação noturna (time_zone = 2) foi concebida para introduzir maior complexidade e desafiar a capacidade do GMM em lidar com cenários menos estruturados. Optou-se por repetir

a lógica da manhã com papéis invertidos, mas com um desenho mais exigente. O frasco de medicamentos passou a ocupar as posições originalmente atribuídas ao cubo, localizadas em (1,1), (2,2) e (3,3), enquanto o cubo assumiu as localizações do frasco, em (−1,−1), (0,−1) e (1,−1).

Cada agrupamento gerou 30 amostras, mas foram adicionadas fontes extra de variabilidade. Os centros dos clusters sofreram maior deslocamentos aleatórios em cada eixo ($\sigma \approx 0.3$), e a dispersão deixou de ser simétrica, com variâncias distintas em x e y. Além disso, foram introduzidos 12 outliers espalhados de forma aleatória por toda a área [−4,4], com rótulos atribuídos de modo aleatório. Este conjunto totalizou 192 entradas (180 associadas a clusters e 12 outliers).

A Figura 27 mostra os três pontos mais prováveis previstos pelo GMM para este cenário. Ao contrário da manhã, os agrupamentos apresentam-se deslocados e menos compactos, refletindo a maior aleatoriedade introduzida.

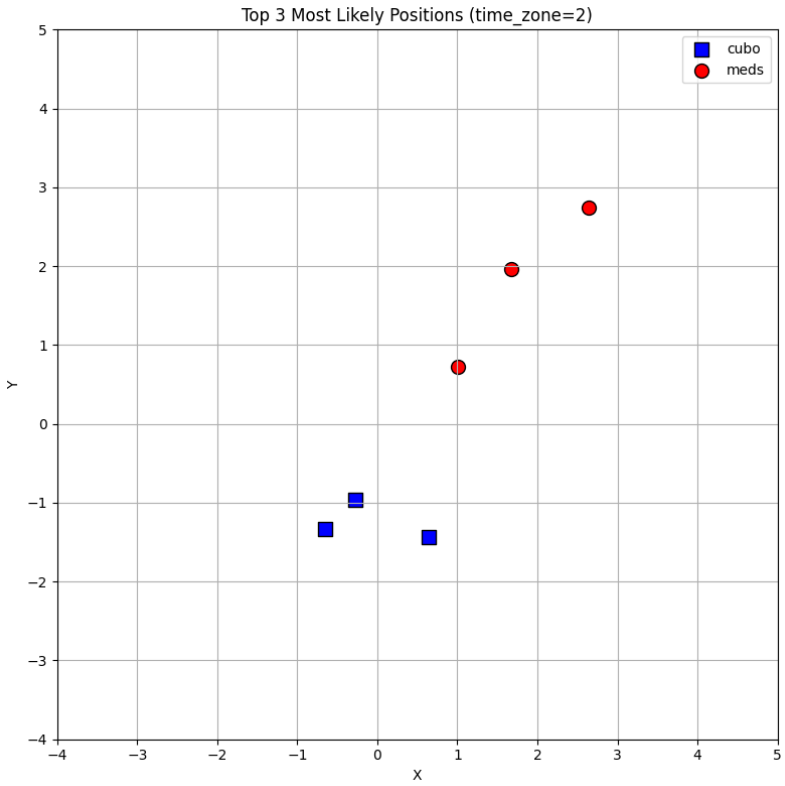

Figura 27 – Posições mais prováveis previstas pelo GMM no período da noite

5.4.1.4 Resumo

No total, o dataset sintético gerado integrou 432 entradas, distribuídas entre 180 na manhã, 60 na tarde e 192 na noite. Estes testes tiveram como objetivo validar se o modelo GMM consegue produzir pontos de procura coerentes com a distribuição temporal e espacial definida artificialmente. A avaliação centrou-se na consistência estatística e lógica dos pontos previstos, confirmando que refletem de forma adequada os padrões introduzidos. Esta validação assegura que a integração entre geração de dados, previsão e agendamento funciona de forma estável

e cria as bases necessárias para fases futuras em que o sistema será alimentado com registos reais de utilização.

5.4.2 Resultados do módulo de agendamento

O formato escolhido para os registos para o modulo de agendamento segue a estrutura *(data, hora, objeto)*, onde o campo objeto assume o valor 0 para cubo e 1 para medicamentos. O objetivo é identificar, com base em padrões temporais simulados, os intervalos do dia em que o robô seria mais frequentemente solicitado.

O dataset gerado corresponde a duas semanas de observações e contém ativações distribuídas de acordo com rotinas plausíveis. O cubo foi associado sobretudo ao período da manhã, em torno das 9h–10h, refletindo cenários em que este objeto é manipulado de forma mais frequente nesse intervalo. O frasco de medicamentos foi associado a momentos do dia mais fixos, nomeadamente ao meio-dia, fim da tarde (cerca das 18h) e início da noite (cerca das 21h). Para evitar padrões artificiais demasiado rígidos, introduziram-se ainda algumas ativações do robô aleatórias fora destes horários, funcionando como ruído de contexto.

Sobre este conjunto foi treinado um modelo GMM unidimensional aplicado às horas do dia. O número de componentes foi escolhido automaticamente com base no critério Bayesian Information Criterion (BIC), que mede a qualidade de ajuste do modelo penalizando ao mesmo tempo a complexidade excessiva. Em termos práticos, o BIC permite encontrar o equilíbrio entre explicação adequada dos dados e simplicidade do modelo, evitando tanto o subajuste (clusters a menos) como o sobreajuste (clusters a mais).

A Figura 28 apresenta os resultados obtidos para o cubo. O histograma em cinzento mostra a distribuição dos dados sintéticos, com concentrações fortes durante a manhã. A curva vermelha corresponde à densidade estimada pelo GMM e as linhas azuis marcam as médias dos clusters aprendidos. O modelo identificou dois agrupamentos principais em torno das 09h00 e 10h00, além de um terceiro picoao final do dia, próximo das 20h30. Estes resultados são consistentes com a forma como os dados foram simulados e demonstram que o modelo consegue distinguir múltiplas ativações no mesmo período do dia.

Figura 28 – Distribuição de chamadas para o cubo com estimativa GMM dos horários mais prováveis

Na Figura 29 estão representados os resultados para o frasco de medicamentos. Tal como esperado, surgem três agrupamentos bem definidos, localizados em torno das 12h00, 18h00 e 21h00. Mais uma vez, o GMM conseguiu reproduzir os padrões temporais subjacentes aos dados, isolando os intervalos de maior probabilidade de chamada.

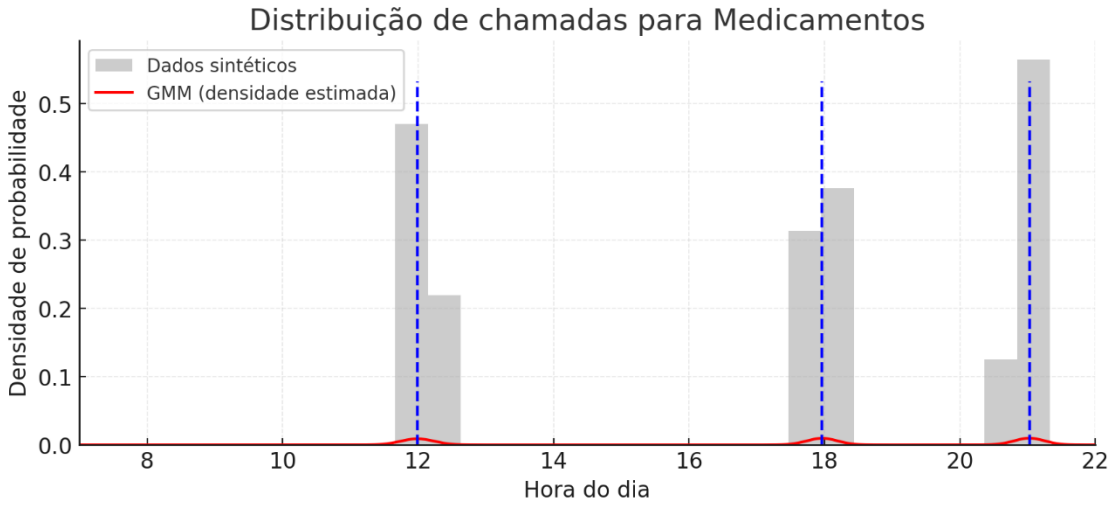

Figura 29 – Distribuição de chamadas para o frasco de medicamentos com estimativa GMM dos horários mais prováveis

No contexto deste trabalho, esta validação assegura que o sistema pode, a partir de registos históricos, antecipar de forma estatisticamente fundamentada os períodos mais prováveis de utilização. Embora os dados tenham sido gerados artificialmente, a coerência dos resultados confirma que a integração do módulo de agendamento com dados reais será viável em fases futuras de desenvolvimento.

5.5 Discussão geral

A análise dos resultados obtidos permite identificar limitações relevantes que condicionaram a robustez do sistema. Em todos os casos foi possível recolher evidência do funcionamento do robô e dos módulos desenvolvidos, mas os resultados devem ser enquadrados com espírito crítico, sublinhando aspetos que necessitam de melhoria.

A deteção de objetos baseada no YOLOv5 (secção 5.1) demonstrou resultados sólidos, com métricas de precisão e recall consistentes para as duas classes utilizadas. Apesar dos resultados encorajadores, há fatores que podem ter contribuído para valores tão elevados, nomeadamente o tamanho limitado e a relativa homogeneidade do dataset, assim como o impacto do *transfer learning*, que já incorpora conhecimento prévio de objetos semelhantes. Estes aspetos podem inflacionar artificialmente a perceção de robustez. O processo de treino beneficiou de um dataset diversificado, que incluiu imagens captadas diretamente pela câmara do robô e outras recolhidas em cenários distintos, garantindo alguma variabilidade em termos de ângulos de visão, intensidade de luz e fundos. Esta abordagem revelou-se eficaz e permitiu obter um modelo suficientemente robusto para suportar os restantes módulos do sistema. Ainda assim, existem oportunidades claras de expansão. Aumentar a dimensão do dataset e adicionar mais variação nas condições de fundo e iluminação permitiria reforçar a generalização do modelo. Outra melhoria possível seria a exploração de técnicas de *data augmentation* mais avançadas, de forma a simular cenários mais realistas sem depender apenas da recolha manual. A introdução de mais classes de objetos, mantendo o equilíbrio entre elas no dataset, também contribuiria para testar a escalabilidade do sistema e a sua capacidade de adaptação a ambientes mais próximos da utilização prática.

Nos testes de preensão (secção 5.2), foi possível validar que o robô consegue, em vários casos, agarrar os objetos identificados. No entanto, a configuração experimental acabou por limitar a diversidade de situações analisadas. A maioria dos testes considerou os objetos colocados em posições relativamente previsíveis e em orientações diretas em relação à câmara. Faltaram variações mais agressivas dos objetos, como posicionamentos de lado ou mesmo virados ao contrário, que poderiam ter demonstrado melhor até onde vai a robustez da pipeline de deteção e manipulação. Além disso, não foi anotada de forma sistemática a posição exata de cada objeto dentro do espaço de treino, o que impossibilita análises mais finas sobre a influência da localização relativa nas taxas temporais e de sucesso. O registo dessa informação teria permitido compreender melhor se o desempenho do sistema se degrada com determinados ângulos.

No que respeita ao cenário de desvio acumulado após uma trajetória de rotina de locomoção (secção 5.3), a metodologia adotada forneceu um primeiro indicador relevante, mas ainda bastante limitado. Foram realizados apenas dez testes, número reduzido para retirar conclusões sólidas. A média obtida, de 41,2 cm de desvio, constitui já um indício claro de que a telemetria atual não é suficientemente precisa. Ensaios mais extensos, com pelo menos algumas dezenas de repetições, permitiriam validar de forma mais rigorosa se o erro se acumula sempre da mesma maneira ou se resulta de fatores contextuais como pequenas variações no

piso ou no atrito das rodas. De qualquer forma, os resultados obtidos sugerem a necessidade de recorrer a tecnologias adicionais que possam reduzir o desvio acumulado. Soluções como sensores inerciais, ou abordagens de visual-SLAM poderiam ser integradas de forma complementar o sistema atual, fornecendo referências externas para corrigir os erros de pose que se acumulam com movimentos sucessivos. Estas alternativas já se encontram bem estabelecidas em sistemas robóticos móveis e, mesmo em protótipos experimentais, poderiam melhorar de forma significativa a estabilidade da navegação e a fiabilidade do posicionamento final. Outro aspeto importante prende-se com a natureza controlada do ambiente experimental. Os testes foram realizados em área reduzida, plana e sem obstáculos, o que simplificou a análise mas, inevitavelmente, retirou realismo. O robô não enfrentou cenários de maior complexidade visual, nem precisou de se adaptar a iluminação variável, objetos móveis ou interferências externas. Como consequência, embora os resultados mostrem que a pipeline base de perceção e ação é funcional, a generalização para contextos domésticos reais permanece em aberto.

No módulo preditivo (secção 5.4), a decisão de recorrer a dados sintéticos foi inevitável dadas as limitações do protótipo e a ausência de utilizadores reais. Essa abordagem permitiu testar a integração técnica entre geração de pontos prováveis e módulo de agendamento, mas não oferece garantias de que o comportamento se mantenha coerente com padrões reais de utilização. Esta escolha, sublinha a necessidade de uma segunda etapa de validação em ambiente vivo com registos obtidos de interações humanas genuínas.

Em síntese, os testes realizados mostram que o sistema cumpre as funções básicas para as quais foi concebido, mas também deixam claro que existem fragilidades estruturais. A estabilidade da câmara, a diversidade dos cenários de manipulação, a quantidade limitada de repetições nos ensaios de desvio e a artificialidade dos dados usados no módulo preditivo são pontos críticos que devem ser melhorados. Estas limitações não invalidam os progressos alcançados, mas contextualizam os resultados como uma primeira demonstração funcional de um protótipo e não como uma avaliação definitiva da sua eficácia em contexto real.

6 Conclusão

Neste capítulo, são explorados os resultados obtidos ao longo do desenvolvimento deste trabalho, com destaque para as principais descobertas e contributos alcançados. O envelhecimento da população exige soluções inovadoras para promover a autonomia e qualidade de vida dos idosos. A robótica apresenta-se como uma resposta promissora, combinando assistência física com estímulos cognitivos, através do desenvolvimento de um assistente robótico que localize, recolha e entregue objetos de forma eficiente e intuitiva. As conclusões são analisadas à luz dos objetivos definidos, oferecendo uma visão sobre o possível impacto do assistente robótico proposto na promoção da autonomia e qualidade de vida dos idosos. Adicionalmente, delineiam-se os próximos passos, identificando oportunidades de melhoria e expansão do projeto.

6.1 Resultados

O trabalho desenvolvido demonstrou a viabilidade de criar um assistente robótico funcional, capaz de localizar e recuperar objetos em ambiente doméstico, mesmo com recursos de hardware limitados. A solução assentou numa arquitetura distribuída que equilibrou de forma eficaz a execução local de tarefas críticas e o processamento remoto de operações intensivas, provando ser uma estratégia adequada para contornar limitações do robô físico.

O sistema mostrou ser capaz de integrar, de forma coerente, módulos de visão computacional, navegação, manipulação e procura orientada por histórico, permitindo que o robô executasse ciclos completos de deteção, aproximação e recolha de objetos. Apesar de ter sido validado em condições controladas, o projeto evidenciou o potencial de evolução da abordagem para contextos mais desafiantes, abrindo caminho à exploração de cenários proativos e interações mais ricas com utilizadores.

Em termos globais, a dissertação cumpriu os objetivos inicialmente definidos: concebeu e implementou um protótipo real, testado em ambiente físico, que responde ao problema identificado e confirma a pertinência de soluções robóticas como apoio à autonomia em contexto doméstico. Para além da concretização prática, o trabalho consolidou um enquadramento metodológico e técnico que poderá servir de base a futuras investigações e melhorias, reforçando a relevância do sistema como contributo académico e tecnológico na área da robótica assistiva.

6.2 Próximos passos

A análise crítica do sistema desenvolvido permite identificar várias oportunidades de evolução que podem reforçar a robustez e a utilidade prática do protótipo.

No domínio da perceção, o modelo de deteção revelou-se sólido, mas a sua generalização beneficiaria de datasets maiores, mais diversificados e com condições de iluminação e fundos menos controlados. Além disso, durante o treino o YOLO foi principalmente exposto a objetos pequenos, próximos do centro da imagem e em escalas semelhantes, o que limita a sua capacidade de lidar com cenários mais variados. A inclusão de novas classes de objetos, a recolha de exemplos em diferentes posições e distâncias e a aplicação de técnicas avançadas de *data augmentation* abririam caminho para uma utilização mais próxima da realidade de um ambiente doméstico.

Na manipulação, embora a garra tenha conseguido executar tarefas de preensão, o número limitado de cenários explorados reduziu a variabilidade dos testes. Futuros trabalhos devem incorporar situações mais desafiantes, como objetos em orientações complexas, em movimento ou parcialmente ocultos, de forma a testar os limites da pipeline de deteção e ação. A anotação rigorosa da posição e da orientação dos objetos em cada ensaio também permitiria caracterizar melhor a influência de fatores espaciais no desempenho da recolha. Para além disso, será importante explorar métodos de recalibração automática da câmara em relação à garra, avaliar o impacto da velocidade de execução na taxa de sucesso, e considerar a integração de feedback tátil ou de força na garra. A capacidade de lidar com múltiplos objetos no mesmo campo de visão (de uma forma mais concreta) e a robustez em cenários dinâmicos representam igualmente linhas promissoras de investigação.

Relativamente à navegação, os desvios acumulados nas trajetórias confirmaram a necessidade de mecanismos adicionais de correção de pose. Importa notar que o número de ensaios realizados nesta vertente foi reduzido, pelo que as conclusões devem ser vistas como indicativas e não definitivas. A integração de sensores complementares, como encoders de maior precisão, unidades inerciais ou soluções de visual-SLAM, poderia reduzir de forma significativa os erros de odometria e aumentar a fiabilidade do posicionamento. Em perspetiva futura, a fusão de dados de múltiplos sensores através de filtros de Kalman, bem como a adoção de técnicas de *loop closure* para corrigir desvios quando o robô regressa a pontos já visitados, representariam avanços relevantes. Do mesmo modo, ensaios em trajetórias mais longas e em ambientes com maior diversidade de superfícies e obstáculos permitiriam avaliar a escalabilidade e robustez do sistema. A integração de algoritmos de planeamento de caminhos, suportada pela implementação prévia de um detetor de obstáculos, poderia complementar a locomoção, contribuindo para mitigar erros acumulados e aproximar o protótipo de cenários domésticos mais realistas.

Outro ponto crítico identificado relaciona-se com falhas inesperadas de comunicação entre o robô e o servidor. Estas situações poderiam ser abordadas de duas formas distintas. Uma possibilidade consiste em manter o robô parado até que a ligação seja restabelecida, exigindo que ambos os lados (robô e servidor) possuam mecanismos de persistência do estado do cenário, de modo a reconhecer que a tarefa ficou a meio e permitir a sua retoma sem reinício completo. Em alternativa, o sistema poderia assegurar que o robô conhece continuamente a sua posição absoluta no espaço, atualizada em permanência pelo servidor. Nesse caso, perante a perda de ligação, o robô regressaria autonomamente ao ponto inicial e aguardaria novo

despoletamento de cenário. Ambas as abordagens aumentariam a resiliência do sistema perante interrupções de comunicação e encontram-se alinhadas com a necessidade de reforçar a fiabilidade operacional do protótipo.

O módulo preditivo demonstrou viabilidade técnica com recurso a dados sintéticos, mas a sua validação em contextos reais é indispensável. A recolha de registos baseados em interações genuínas com utilizadores permitiria ajustar os modelos às dinâmicas humanas e aferir o impacto da previsão na eficiência das procuras.

Os próximos passos devem concentrar-se na diversificação das condições experimentais, na incorporação de sensores adicionais, na validação com dados reais de utilização e na expansão das capacidades do sistema em termos de classes de objetos e complexidade ambiental. Será igualmente necessário reforçar a gestão de falhas de comunicação, garantindo que o robô é capaz de parar em segurança e retomar cenários a meio ou, em alternativa, regressar ao ponto inicial após uma perda de ligação. Estes avanços permitirão transformar o protótipo, que hoje se afirma como uma prova de conceito funcional, num assistente robótico mais completo, robusto e alinhado com os desafios da aplicação prática em ambientes domésticos.

Referências

[1] «Fertility statistics - Statistics Explained - Eurostat». Acedido: 15 de Setembro de 2025. [online]. Disponível em: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Fertility_statistics

[2] «Portal do INE». Acedido: 15 de Setembro de 2025. [online]. Disponível em: https://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_pesquisa&frm_accao=PESQUISAR&frm_show_page_num=1&frm_modo_pesquisa=PESQUISA_SIMPLES&frm_texto=t%C3%A1bua+de+mortalidade&frm_modo_texto=MODO_TEXTO_ALL&frm_data_ini=&frm_data_fim=&frm_tema=QUALQUER_TEMA&frm_area=o_ine_area_Destaques&xlang=pt

[3] «Population structure and ageing - Statistics Explained - Eurostat». Acedido: 15 de Setembro de 2025. [online]. Disponível em: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Population_structure_and_ageing

[4] J. Ma, Q. Chen, X. Chen, J. Fan, X. Li, e Y. Shi, «An inevitably aging world -- Analysis on the evolutionary pattern of age structure in 200 countries», Fev. 2024, Acedido: 28 de Novembro de 2024. [online]. Disponível em: https://arxiv.org/abs/2402.04612v1

[5] «Portal do INE». Acedido: 28 de Novembro de 2024. [online]. Disponível em: https://www.ine.pt/xportal/xmain?DESTAQUESdest_boui=406534255&DESTAQUESmodo=2&xlang=pt&xpgid=ine_destaques&xpid=INE

[6] A. SOFIA, «Adiar a maternidade: uma questão demográfica e contemporânea», FMUC, Coimbra, 2019.

[7] P. Maresova, J. Hruska, B. Klimova, S. Barakovic, e O. Krejcar, « Activities of Daily Living and Associated Costs in the Most Widespread Neurodegenerative Diseases: A Systematic Review », *Clin Interv Aging*, vol. 15, pp. 1841–1862, Out. 2020, doi: 10.2147/CIA.S264688.

[8] N. Shoval *et al.*, «Use of the global positioning system to measure the out-of-home mobility of older adults with differing cognitive functioning», *Ageing Soc*, vol. 31, n. 5, pp. 849–869, Jul. 2011, doi: 10.1017/S0144686X10001455.

[9] B.-F. Cao *et al.*, «Association Between Mobility Limitations and Cognitive Decline in Community-Dwelling Older Adults: The English Longitudinal Study of Ageing», *Gerontologist*, vol. 64, n. 12, Dez. 2024, doi: 10.1093/GERONT/GNAE139.

[10] I. E. M. Evans, A. Martyr, R. Collins, C. Brayne, e L. Clare, «Social Isolation and Cognitive Function in Later Life: A Systematic Review and Meta-Analysis», *Journal of Alzheimer's Disease*, vol. 70, n. s1, pp. S119–S144, Out. 2019, doi: 10.3233/JAD-180501/ASSET/IMAGES/10.3233_JAD-180501-FIG2.JPG.

[11] R. M. Gyasi, K. Abass, e S. Adu-Gyamfi, «How do lifestyle choices affect the link between living alone and psychological distress in older age? Results from the AgeHeaPsyWel-HeaSeeB study», *BMC Public Health*, vol. 20, n. 1, Jun. 2020, doi: 10.1186/S12889-020-08870-8.

[12] E. J. Groessl *et al.*, «Physical Activity and Performance Impact Long-term Quality of Life in Older Adults at Risk for Major Mobility Disability», *Am J Prev Med*, vol. 56, n. 1, pp. 141–146, Jan. 2019, doi: 10.1016/j.amepre.2018.09.006.

[13] S. Padhan, A. Mohapatra, S. K. Ramasamy, e S. Agrawal, «Artificial Intelligence (AI) and Robotics in Elderly Healthcare: Enabling Independence and Quality of Life», *Cureus*, vol. 15, n. 8, p. e42905, Ago. 2023, doi: 10.7759/CUREUS.42905.

[14] B. Sawik *et al.*, «Robots for Elderly Care: Review, Multi-Criteria Optimization Model and Qualitative Case Study», *Healthcare*, vol. 11, n. 9, p. 1286, Mai. 2023, doi: 10.3390/HEALTHCARE11091286.

[15] X. Y. Lin, J. Moxley, J. Sharit, e S. J. Czaja, «Beyond the Digital Divide: Factors Associated with Adoption of Technologies Related to Aging in Place», *J Appl Gerontol*, vol. 44, n. 6, p. 959, Jun. 2025, doi: 10.1177/07334648251318789.

[16] «AI Act | Shaping Europe's digital future». Acedido: 15 de Setembro de 2025. [online]. Disponível em: https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai

[17] «High-level summary of the AI Act | EU Artificial Intelligence Act». Acedido: 15 de Setembro de 2025. [online]. Disponível em: https://artificialintelligenceact.eu/high-level-summary/

[18] Y. K. Dwivedi *et al.*, «Artificial Intelligence (AI): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy», *Int J Inf Manage*, vol. 57, Abr. 2021, doi: 10.1016/J.IJINFOMGT.2019.08.002.

[19] Y. Shi, K. Yang, T. Jiang, J. Zhang, e K. B. Letaief, «Communication-Efficient Edge AI: Algorithms and Systems», *IEEE Communications Surveys and Tutorials*, vol. 22, n. 4, pp. 2167–2191, Out. 2020, doi: 10.1109/COMST.2020.3007787.

[20] N. Thompson, K. Greenewald, K. Lee, e G. F. Manso, «The Computational Limits of Deep Learning», Jun. 2023, doi: 10.21428/BF6FB269.1F033948.

[21] S. Padhan, A. Mohapatra, S. K. Ramasamy, e S. Agrawal, «Artificial Intelligence (AI) and Robotics in Elderly Healthcare: Enabling Independence and Quality of Life», *Cureus*, vol. 15, Ago. 2023, doi: 10.7759/CUREUS.42905.

[22] S. Kamio e H. Iba, «Adaptation technique for integrating genetic programming and reinforcement learning for real robots», *IEEE Transactions on Evolutionary Computation*, vol. 9, n. 3, pp. 318–333, Jun. 2005, doi: 10.1109/TEVC.2005.850290.

[23] A. Sehgal, H. La, S. Louis, e H. Nguyen, «Deep Reinforcement Learning Using Genetic Algorithm for Parameter Optimization», *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pp. 596–601, Mar. 2019, doi: 10.1109/IRC.2019.00121.

[24] H. Taheri e Z. C. Xia, «SLAM; definition and evolution», *Eng. Appl. Artif. Intell.*, vol. 97, Jan. 2021, doi: 10.1016/J.ENGAPPAI.2020.104032.

[25] S. Y. Chen, «Kalman Filter for Robot Vision: A Survey», *IEEE Transactions on Industrial Electronics*, vol. 59, n. 11, pp. 4409–4420, Nov. 2012, doi: 10.1109/TIE.2011.2162714.

[26] G. Welch e G. Bishop, «An Introduction to the Kalman Filter», Acedido: 5 de Fevereiro de 2025. [online]. Disponível em: http://www.cs.unc.edu/~gb

[27] R. Song, Y. Liu, e R. Bucknall, «Smoothed A* algorithm for practical unmanned surface vehicle path planning», *Applied Ocean Research*, vol. 83, pp. 9–20, Fev. 2019, doi: 10.1016/J.APOR.2018.12.001.

[28] M. H. Abidi, M. K. Mohammed, e H. Alkhalefah, «Predictive Maintenance Planning for Industry 4.0 Using Machine Learning for Sustainable Manufacturing», *Sustainability*, vol. 14, n. 6, Mar. 2022, doi: 10.3390/SU14063387.

[29] K. Yousif, A. Bab-Hadiashar, e R. Hoseinnezhad, «An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics», *Intelligent Industrial Systems 2015 1:4*, vol. 1, n. 4, pp. 289–311, Nov. 2015, doi: 10.1007/S40903-015-0032-7.

[30] Y. Li, «DEEP REINFORCEMENT LEARNING: AN OVERVIEW», 2018, Acedido: 9 de Fevereiro de 2025. [online]. Disponível em: https://arxiv.org/abs/

[31] Z. Li, F. Liu, W. Yang, S. Peng, e J. Zhou, «A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects», *IEEE Trans Neural Netw Learn Syst*, vol. 33, n. 12, pp. 6999–7019, Dez. 2022, doi: 10.1109/TNNLS.2021.3084827.

[32] «CNNs architectures commonly used in medical imaging.(a) One CNN with 2... | Download Scientific Diagram». Acedido: 10 de Fevereiro de 2025. [online]. Disponível em: https://www.researchgate.net/figure/CNNs-architectures-commonly-used-in-medical-imaginga-One-CNN-with-2-convolutional_fig2_333771086

[33] F. Wang e D. M. J. Tax, «Survey on the attention based RNN model and its applications in computer vision», Jan. 2016, Acedido: 10 de Fevereiro de 2025. [online]. Disponível em: https://arxiv.org/abs/1601.06823v1

[34] P. Jiang, D. Ergu, F. Liu, Y. Cai, e B. Ma, «A Review of Yolo Algorithm Developments», *Procedia Comput Sci*, vol. 199, pp. 1066–1073, Jan. 2022, doi: 10.1016/J.PROCS.2022.01.135.

[35] «YOLO-v2 model detection. First partition the image into an S × R grid... | Download Scientific Diagram». Acedido: 10 de Fevereiro de 2025. [online]. Disponível em:

https://www.researchgate.net/figure/YOLO-v2-model-detection-First-partition-the-image-into-an-SR-grid-and-for-each-grid_fig1_361243861

[36] V. A. Bakale, Y. Kumar, V. C. Roodagi, Y. N. Kulkarni, M. S. Patil, e S. Chickerur, «Indoor Navigation with Deep Reinforcement Learning», *Proceedings of the 5th International Conference on Inventive Computation Technologies, ICICT 2020*, pp. 660–665, Fev. 2020, doi: 10.1109/ICICT48043.2020.9112385.

[37] P. De Almeida Afonso e P. R. Ferreira, «Autonomous Navigation of Wheelchairs in Indoor Environments using Deep Reinforcement Learning and Computer Vision», *Proceedings - 2023 Latin American Robotics Symposium, 2023 Brazilian Symposium on Robotics, and 2023 Workshop of Robotics in Education, LARS/SBR/WRE 2023*, pp. 260–265, 2023, doi: 10.1109/LARS/SBR/WRE59448.2023.10332918.

[38] J. P. Valcourt, F. M. Chandler, C. Avrelus, J. Y. Lee, A. I. Güllü, e S. H. Shah, «Practical Implementation of Q-Learning and Object Detection for Mobile Robot Path Planning», *International Conference on Advanced Robotics and Intelligent Systems, ARIS*, 2024, doi: 10.1109/ARIS62416.2024.10679961.

[39] S. James, K. Wada, T. Laidlow, e A. J. Davison, «Coarse-to-Fine Q-attention: Efficient Learning for Visual Robotic Manipulation via Discretisation», *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2022-June, pp. 13729–13738, Jun. 2021, doi: 10.1109/CVPR52688.2022.01337.

[40] E. Erkan e M. A. Arserim, «Mobile Robot Application with Hierarchical Start Position DQN», *Comput Intell Neurosci*, vol. 2022, n. 1, p. 4115767, Jan. 2022, doi: 10.1155/2022/4115767.

[41] Y. Liu e H. Jebelli, «Intention-aware robot motion planning for safe worker–robot collaboration», *Computer-Aided Civil and Infrastructure Engineering*, vol. 39, n. 15, pp. 2242–2269, Ago. 2024, doi: 10.1111/MICE.13129.

[42] C. D. de S. Bezerra, F. H. T. Vieira, e A. da S. Soares, «Deep-Q-Network hybridization with extended Kalman filter for accelerate learning in autonomous navigation with auxiliary security module», *Transactions on Emerging Telecommunications Technologies*, vol. 35, n. 2, p. e4946, Fev. 2024, doi: 10.1002/ETT.4946.

[43] C. Cardenas-Perez *et al.*, «XBG: End-to-end Imitation Learning for Autonomous Behaviour in Human-Robot Interaction and Collaboration», Jun. 2024, Acedido: 28 de Dezembro de 2024. [online]. Disponível em: https://arxiv.org/abs/2406.15833v1

[44] J. Guo *et al.*, «Research on the autonomous system of the quadruped robot with a manipulator to realize leader-following, object recognition, navigation and operation», *IET Cyber-Systems and Robotics*, vol. 4, n. 4, pp. 376–388, Dez. 2022, doi: 10.1049/CSY2.12069.

[45] U. A. Zahidi *et al.*, «Optimising robotic operation speed with edge computing via 5G network: Insights from selective harvesting robots», *J Field Robot*, vol. 41, n. 8, pp. 2771–2789, Dez. 2024, doi: 10.1002/ROB.22384.

[46] M. Kutbi, Y. Chang, B. Sun, e P. Mordohai, «Learning to navigate robotic wheelchairs from demonstration: Is training in simulation viable?», *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, pp. 2522–2531, Out. 2019, doi: 10.1109/ICCVW.2019.00309.

[47] L. Zhang, H. Zhang, H. Yang, G. Bin Bian, e W. Wu, «Multi-target detection and grasping control for humanoid robot NAO», *Int J Adapt Control Signal Process*, vol. 33, n. 7, pp. 1225–1237, Jul. 2019, doi: 10.1002/ACS.3031.

[48] J. Rong, L. Hu, H. Zhou, G. Dai, T. Yuan, e P. Wang, «A selective harvesting robot for cherry tomatoes: Design, development, field evaluation analysis», *J Field Robot*, vol. 41, n. 8, pp. 2564–2582, Dez. 2024, doi: 10.1002/ROB.22377.

[49] M. Sorokin, W. Yu, S. Ha, e C. K. Liu, «Learning Human Search Behavior from Egocentric Visual Inputs», *Computer Graphics Forum*, vol. 40, n. 2, pp. 389–398, Nov. 2020, doi: 10.1111/cgf.142641.

[50] D. Liu, Z. Li, Z. Wu, e C. Li, «Digital Twin/MARS-CycleGAN: Enhancing Sim-to-Real Crop/Row Detection for MARS Phenotyping Robot Using Synthetic Images», *J Field Robot*, 2024, doi: 10.1002/ROB.22473.

[51] Q. Fang, X. Xu, X. Wang, e Y. Zeng, «Target-driven visual navigation in indoor scenes using reinforcement learning and imitation learning», *CAAI Trans Intell Technol*, vol. 7, n. 2, pp. 167–176, Jun. 2022, doi: 10.1049/CIT2.12043.

[52] Y. T. Hwang *et al.*, «IAMEC, an Intelligent Autonomous Mover for Navigation in Indoor People Rich Environments», *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems, AICAS 2021*, Jun. 2021, doi: 10.1109/AICAS51828.2021.9458563.

[53] M. Gualtieri e R. Platt, «Learning Manipulation Skills Via Hierarchical Spatial Attention», *IEEE Transactions on Robotics*, vol. 36, n. 4, pp. 1067–1078, Abr. 2019, doi: 10.1109/TRO.2020.2974093.

[54] S. Mashhouri, M. Rahmati, Y. Borhani, e E. Najafi, «Reinforcement Learning based Sequential Controller for Mobile Robots with Obstacle Avoidance», *2022 8th International Conference on Control, Instrumentation and Automation, ICCIA 2022*, 2022, doi: 10.1109/ICCIA54998.2022.9737166.

[55] D. P. Losey e M. K. O'Malley, «Learning the Correct Robot Trajectory in Real-Time from Physical Human Interactions», *ACM Transactions on Human-Robot Interaction (THRI)*, vol. 9, n. 1, pp. 1–19, Dez. 2019, doi: 10.1145/3354139.

[56] T. C. Chi, M. Eric, S. Kim, M. Shen, e D. Hakkani-Tur, «Just Ask:An Interactive Learning Framework for Vision and Language Navigation», *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pp. 2459–2466, Dez. 2019, doi: 10.1609/aaai.v34i03.5627.

[57] P. Wang, R. Ma, Z. Yang, e Q. Hao, «Robotic Camera Array Motion Planning for Multiple Human Face Tracking Based on Reinforcement Learning», *IEEE Sens J*, vol. 24, n. 15, pp. 24649–24658, 2024, doi: 10.1109/JSEN.2024.3415954.

[58] S. D. Sierra M., M. Garzón, M. Múnera, e C. A. Cifuentes, «Human–Robot–Environment Interaction Interface for Smart Walker Assisted Gait: AGoRA Walker», *Sensors 2019, Vol. 19, Page 2897*, vol. 19, n. 13, p. 2897, Jun. 2019, doi: 10.3390/S19132897.

[59] A. Taghibakhshi, N. Ogden, e M. West, «Local Navigation and Docking of an Autonomous Robot Mower Using Reinforcement Learning and Computer Vision», *2021 13th International Conference on Computer and Automation Engineering, ICCAE 2021*, pp. 10–14, Mar. 2021, doi: 10.1109/ICCAE51876.2021.9426091.

[60] G. Labartkava, N. B. Suhaimi, T. Silva, e L. Kirischian, «Autonomously Reconfigurable Robot Coupled with Intelligent Vision and Control System for Rapid Runtime Adaptation», *2024 IEEE 5th World AI IoT Congress, AIIoT 2024*, pp. 429–434, 2024, doi: 10.1109/AIIOT61789.2024.10579027.

[61] H. Wang, W. Liang, L. Van Gool, e W. Wang, «DREAMWALKER: Mental Planning for Continuous Vision-Language Navigation», *Proceedings of the IEEE International Conference on Computer Vision*, pp. 10839–10849, Ago. 2023, doi: 10.1109/ICCV51070.2023.00998.

[62] T. Bhuiyan, H. Flach, J. Lambrecht, S. Gasper, J. Funder, e J. Bickendorf, «Collision-Free Welding Torch Pose Determination Through Imitation Learning with Expert Demonstrations», *2024 9th International Conference on Control and Robotics Engineering, ICCRE 2024*, pp. 27–32, 2024, doi: 10.1109/ICCRE61448.2024.10589880.

[63] N. Xu, «Active object searching on mobile robot using reinforcement learning», *Proceedings - 2021 2nd International Conference on Computing and Data Science, CDS 2021*, pp. 296–300, Jan. 2021, doi: 10.1109/CDS52072.2021.00058.

[64] «Transbot SE». Acedido: 30 de Julho de 2025. [online]. Disponível em: https://www.yahboom.net/study/Transbot-SE

[65] «YOLO 1 through 5: A complete and detailed overview». Acedido: 18 de Agosto de 2025. [online]. Disponível em: https://www.kaggle.com/code/vikramsandu/yolo-1-through-5-a-complete-and-detailed-overview

[66] «Explore Ultralytics YOLOv8 - Ultralytics YOLO Docs». Acedido: 18 de Agosto de 2025. [online]. Disponível em: https://docs.ultralytics.com/models/yolov8/

[67] C. Y. Wang, A. Bochkovskiy, e H. Y. M. Liao, «YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors», *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2023-June, pp. 7464–7475, Jul. 2022, doi: 10.1109/CVPR52729.2023.00721.

[68] «aws-robotics/aws-robomaker-small-house-world: A house world with multiple rooms and furniture for AWS RoboMaker and Gazebo simulations.» Acedido: 18 de Agosto de 2025. [online]. Disponível em: https://github.com/aws-robotics/aws-robomaker-small-house-world

Apêndice A – Artigos resultantes do PRISMA

Tabela 10 - Tabela com os artigos resultantes da seleção PRISMA

Nome do Artigo	**Descrição**
1. Collision-Free Welding Torch Pose Determination Through Imitation Learning with Expert Demonstrations	Desenvolvimento de um sistema baseado em *imitation learning* para criar trajetórias seguras e livres de colisões para uma tocha de soldar.
2. Dreamwalker: Mental Planning for Continuous Vision-Language Navigation	Integração de visão computacional com compreensão de linguagem para permitir a navegação contínua de robôs em ambientes dinâmicos.
3. iAMEC: An Intelligent Autonomous Mover for Navigation in Indoor People-Rich Environments	Sistema de navegação autónomo para ambientes internos povoados, utilizando sensores de visão para deteção de obstáculos humanos.
4. Indoor Navigation with Deep Reinforcement Learning	Aplicação de *Deep Reinforcement Learning* para ensinar robôs a navegar em ambientes internos com redes neuronais.
5. Learning Manipulation Skills via Hierarchical Spatial Attention	Método de atenção espacial hierárquica para ensinar robôs a manipular objetos de forma precisa e eficiente.
6. Learning to Navigate Robotic Wheelchairs from Demonstration: Is Training in Simulation Viable?	Avaliação da eficácia em treinar cadeiras de rodas robóticas automáticas.
7. Active Object Searching on Mobile Robot Using Reinforcement Learning	Abordagem de *reinforcement learning* para busca ativa de objetos em ambientes internos.
8. Autonomous Navigation of Wheelchairs in Indoor Environments using Deep Reinforcement Learning and Computer Vision	Combinação de visão computacional e *Deep Reinforcement Learning* para navegação autónoma de cadeiras de rodas.

9. Autonomously Reconfigurable Robot Coupled with Intelligent Vision and Control System for Rapid Runtime Adaptation	Sistema robótico capaz de adaptação rápida a diferentes cenários através de visão binocular e controle inteligente.
10. Coarse-to-Fine Q-attention: Efficient Learning for Visual Robotic Manipulation via Discretisation	Método de discretização para manipulação robótica visual.
11. Local Navigation and Docking of an Autonomous Robot Mower Using Reinforcement Learning and Computer Vision	Uso de *reinforcement learning* e visão computacional para navegação local e atracagem de robôs.
12. Practical Implementation of Q-Learning and Object Detection for Mobile Robot Path Planning	Implementação prática de *Q-Learning* para planeamento de trajetórias com deteção de objetos.
13. Reinforcement Learning-Based Sequential Controller for Mobile Robots with Obstacle Avoidance	Controlador sequencial baseado em *reinforcement learning* para evitar obstáculos em robôs móveis.
14. Robotic Camera Array Motion Planning for Multiple Human Face Tracking Based on Reinforcement Learning	Planeamento de movimento para rastreamento facial usando *reinforcement learning*.
15. End-to-End Imitation Learning for Autonomous Behaviour in Human-Robot Interaction and Collaboration	Uso de *imitation learning* para ensinar robôs a colaborar com humanos.
16. Learning the Correct Robot Trajectory in Real-Time	Sistema de aprendizagem em tempo real para ajuste de trajetórias robóticas.
17. An Interactive Learning Framework for Vision and Language Navigation	Estrutura de aprendizagem interativa que combina navegação visual e linguagem natural.
18. Human–Robot–Environment Interaction Interface for Smart Walker Assisted Gait	Interface de interação humano-robô-ambiente para um *walker* inteligente.
19. Multi-Target Detection and Grasping Control for Humanoid Robot NAO	Deteção de múltiplos alvos e controlo de preensão em robôs humanoides.
20. Target-Driven Visual Navigation in Indoor Scenes Using Reinforcement Learning	Navegação visual em ambientes internos baseada em *reinforcement learning*.

21. Mobile Robot Application with Hierarchical Start Position DQN	Aplicação de um modelo hierárquico de *Deep Q-Network* para navegação de robôs móveis.
22. Intention-Aware Robot Motion Planning for Safe Worker-Robot Collaboration	Planeamento de movimento considerando previsões de intenção humana.
23. Learning Human Search Behavior from Egocentric Visual Inputs	Modelagem do comportamento de busca humana para robôs.
24. Research on the Autonomous System of the Quadruped Robot with a Manipulator	Desenvolvimento de um robô quadrúpede autónomo com manipulador.
25. Digital Twin MARS-CycleGAN Enhancing Sim-to-Real Crop Row Detection	Uso de *Digital Twin* com *CycleGANs* para transferência de simulação para o mundo real.
26. A Selective Harvesting Robot for Cherry Tomatoes: Design and Development	Desenvolvimento de um robô seletivo de colheita de tomates-cereja.
27. Optimizing Robotic Operation Speed with Edge Computing via 5G Networks	Otimização da velocidade operacional de robôs com *edge computing* e redes 5G.
28. Deep-Q-Network Hybridization with Extended Kalman Filter	Hibridização de *Deep-Q-Network* com *Kalman filter* para navegação autónoma.

Apêndice B – Diagramas de transição de estados do robô nos cenários passivo e proativo

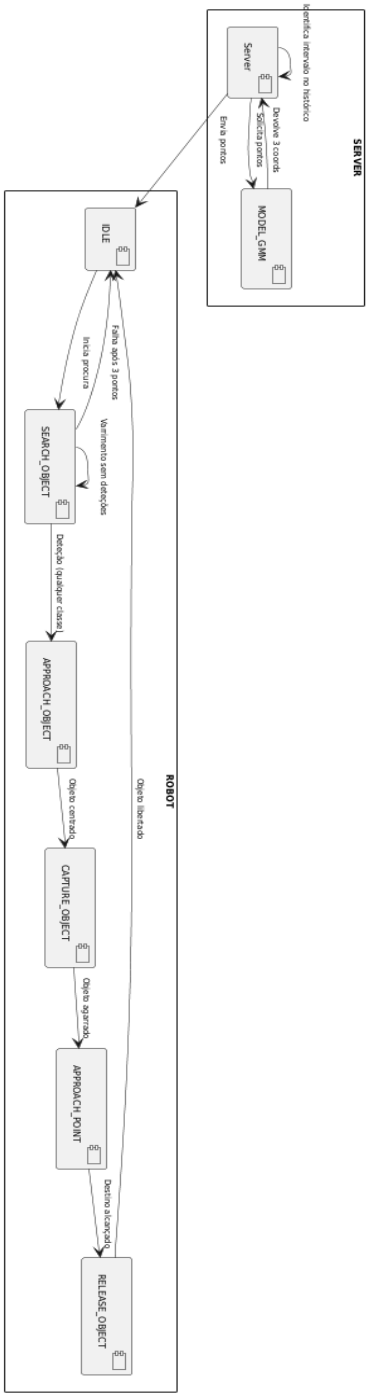

Figura 30 – Diagrama de transição de estado do cenário proativo

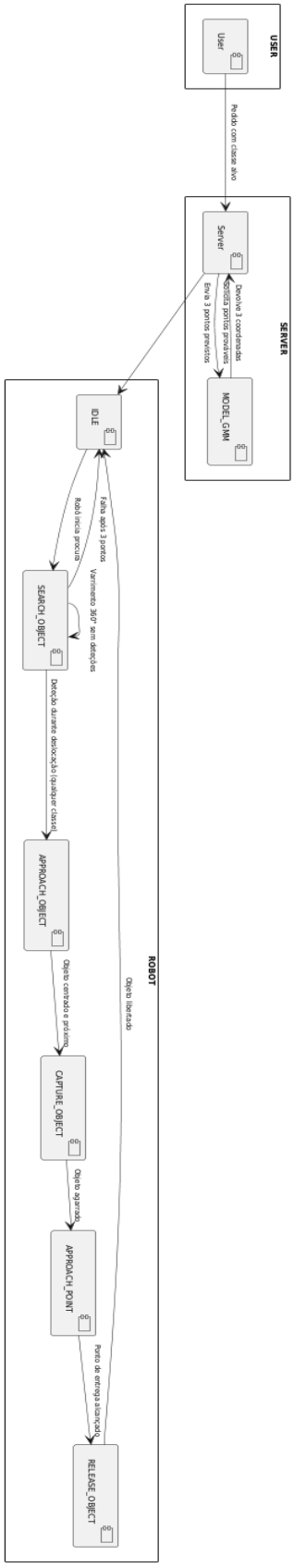

Figura 31 – Diagrama de transição de estado do cenário passivo