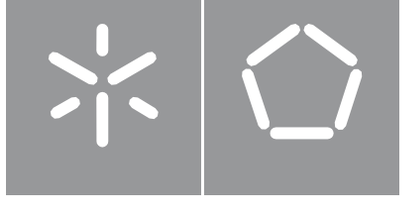
Universidade do Minho
Escola de Engenharia

João Diogo Santos Correia

Controlo da Navegação e Manobras de Acostagem de Manipuladores Móveis Autónomos em Ambientes de Logística Interna

janeiro de 2023

Controlo da Navegação e Manobras de Acostagem de Manipuladores Móveis Autónomos em Ambientes de Logística Interna
João Diogo Santos Correia
UMinho | 2023



Universidade do Minho
Escola de Engenharia

João Diogo Santos Correia

Controlo da Navegação e Manobras de Acostagem de Manipuladores Móveis Autónomos em Ambientes de Logística Interna

Dissertação de Mestrado
Mestrado Integrado em
Engenharia Eletrónica Industrial
e Computadores

Trabalho efetuado sob a orientação do(a)
Doutor Luís Filipe Castro Freitas Louro

janeiro de 2023

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Agradecimentos

Como as jornadas percorridas com aqueles que nos apoiam e motivam são as que nos impulsionam a alcançar os objetivos, e, portanto, as que deixam um marco na nossa história, esta secção é dedicada a todos vocês que estiveram sempre a meu lado.

Aos meus pais e à minha irmã, estou eternamente grato. Pelo apoio incondicional, pela paciência, pelo incentivo na minha formação, por sempre acreditarem em mim.

À minha prima Patrícia, ao meu primo Barreto e à minha tia Céu, pelo apoio e por me encorajarem a prosseguir os meus estudos.

Às "três Lenas", pelo apoio e por estarem sempre presentes ao longo do meu percurso.

Aos meus padrinhos, por sempre me acompanharem e pelo carinho dado.

Ao meu orientador, o professor Luís Louro, pelo apoio prestado, pela sua disponibilidade e pelo tempo despendido no auxílio e revisão durante o desenvolvimento desta dissertação.

A toda a equipa do MarLab, pela receção e integração no grupo, proporcionando um ambiente de amizade e propício ao progresso.

A todos os meus amigos que me acompanharam nesta jornada. Ao Pedro, Ricardo, Paulo, Filipa, Maria e Tiago, pelo incrível espírito de camaradagem, pelo apoio prestado e constante disponibilidade.

A todos vocês, um muito obrigado! Certamente que, com as pessoas certas ao nosso lado, vamos longe, independentemente dos desafios que nos esperam.

Declaração de Integridade

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração. Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Resumo

A automatização de processos industriais é já uma realidade consolidada, contudo, por vezes estes mecanismos revelam-se insuficientes para suprimir as atuais exigências do meio industrial, entre as quais a otimização e gestão de recursos. Este trabalho de dissertação pretendeu mitigar a problemática recorrendo a soluções mais recentes do âmbito da robótica, isto é, à robótica colaborativa. Estes tipos de sistemas têm a vantagem de poder partilhar o espaço de trabalho com operadores humanos.

O presente trabalho apresenta uma solução inteligente e flexível que permite a partilha de mecanismos operacionais entre os diferentes tipos de processos industriais distribuídos numa indústria. Para o efeito, foi proposto um sistema autónomo capaz de realizar as operações de navegação e de acostagem em ambiente industrial complexo. Esta dissertação está inserida no projeto de investigação MIAR (*Mobile Intelligent Autonomous Robots*), realizado no âmbito da parceria Bosch, UMinho e CCG (Centro Computação Gráfica), no qual é pretendido a conceção de uma solução que facilite o transporte e manipulação de peças em chão de fábrica de forma autónoma, recorrendo a robôs colaborativos. Para tal, recorreu-se a um manipulador móvel, composto por um braço manipulador e por uma plataforma omnidirecional, cujo controlo faz parte dos objetivos da presente dissertação. Dada a partilha do espaço com operadores humanos, o veículo deve estabelecer movimentos caracterizados como legíveis. Contudo, as operações de acostagem e de navegação em espaços exíguos devem ser estabelecidas por movimentos holonómicos. Por forma a incorporar as operações previstas, o sistema foi estruturado segundo uma arquitetura lógica, composta por módulos gerais que incorporam e auxiliam a execução dos diversos tipos de comportamentos considerados (de navegação e de acostagem). A integração e gestão da ativação dos comportamentos encontra-se incorporada por um algoritmo dedicado.

Para efeito de teste e de validação, foi desenvolvido um cenário de simulação, similar a um ambiente industrial, composto pelo modelo dinâmico da plataforma móvel e por locais estrategicamente desenvolvidos para a validação dos comportamentos, conforme os requisitos de projeto.

Os objetivos propostos na dissertação foram atingidos, tendo sido realizados testes em simulação em que o veículo se mostrou capaz de deslocar-se entre os diversos locais da fábrica, em ambiente partilhado com operadores e outros veículos, sendo também capaz de acostar junto a linhas de montagem e a máquinas-ferramentas. A acostagem ao local de park e de carregamento também foi contemplada.

Palavras-chave: Plataforma Omnidirecional, Manobras de Acostagem, Navegação Holonómica, Navegação Autónoma, Soluções Inteligentes e Flexíveis.

Abstract

The automation of industrial processes is already a consolidated reality, nevertheless, sometimes these mechanisms seem insufficient to overcome the actual demands of the industrial sector, including the optimization and management of resources. This dissertation intends to mitigate the problem by resorting to robotics' state of the art solutions, namely, collaborative robotics. These types of systems have the advantage of being able to share the workspace with human operators.

This work presents an intelligent and flexible solution that allows the sharing of operational mechanisms among different industrial processes. For this purpose, it is presented an autonomous system capable of performing navigation and docking operations in a complex industrial environment. This dissertation is integrated in the MIAR's (Mobile Intelligent Autonomous Robots) research project, carried out in the scope of the partnership between Bosch, UMinho and CCG (*Centro Computação Gráfica*), in which is pretended to design a solution that facilitates the transport and manipulation of components in factory floor autonomously, by using collaboratives robots. For such, a mobile manipulator was used, composed of a manipulator arm and an omnidirectional platform, whose control is part of this dissertation objectives. Since the vehicle shares space with human operators, it must establish movements characterized as readable, by this means, easily understandable by a human being. However, the docking and navigation operations in confined spaces should be performed by holonomic movements. In order to incorporate the planned operations, the system was structured based on a logical architecture, composed of general modules that incorporate and assist the execution of the considered behaviours (navigation and docking). The integration and behavioural activation management is incorporated in a dedicated algorithm.

For testing and validation purposes, a simulation scenario was developed, similar to an industrial environment, composed of the dynamic model of the mobile platform and by strategically developed locations for the behaviour validation, according to the design requirements. The proposed objectives of the dissertation were achieved, as simulation tests were carried out in which the vehicle proved capable of moving between the various plant sites, in an environment shared with operators and other vehicles, and also capable of docking beside assembly lines and machine tools. Docking to the park and charging points has also been contemplated.

Keywords: Omnidirectional Platform, Docking Manoeuvres, Holonomic Navigation, Autonomous Navigation, Intelligent and Flexible solutions.

CONTEÚDO

Índice de Figuras

Índice de Tabelas

Lista de Abreviaturas, Siglas e Acrónimos

Neste documento são utilizadas diversas abreviaturas e siglas, sistematizadas de seguida:

A* – **A**-***Star*** (**A**-Estrela)

AGV – ***A**utomated **G**uided **V**ehicle* (**V**eículo **G**uiado **A**utomaticamente)

AMR – ***A**utonomous **M**obile **R**obot* (**R**obô **M**óvel **A**utónomo)

API – ***A**pplication **P**rogramming **I**nterface* (**I**nterface de **P**rogramação de **A**plicações)

ArUco – ***A**ugmented **r**eality **U**niversity of **Co**rdoba* (**R**ealidade **A**umentada **U**niversidade de **Có**rdoba)

CAD – ***C**omputer **A**ided **D**esign* (**D**esenho **A**ssistido por **C**omputador)

CADRL – ***C**ollision **A**voidance with **Deep R**einforcement **L**earning* (Evitar Colisões via Aprendizagem Profunda por Reforço)

CCG – **C**entro de **C**omputação **G**ráfica

CPU – ***C**entral **P**rocessing **U**nity* (**U**nidade **C**entral de **P**rocessamento)

DWA – ***D**ynamic **W**indow **A**pproach* (Abordagem Janela Dinâmica)

FLC – ***F**uzzy **l**ogic **c**ontrol* (**C**ontrolo **L**ógico ***F**uzzy* ou difuso)

GPS – ***G**lobal **P**ositioning **S**ystem* (**S**istema **G**lobal de **P**osicionamento)

IIoT – ***I**ndiustrial **I**nternet **o**f **T**hings* (Internet Industrial das Coisas)

iiwa – ***i**ntelligent **i**ndustrial **w**ork **a**ssistant* (Assistente inteligente de trabalho industrial)

IMU – ***I**nertial **M**easurement **U**nit* (Sensor Inercial)

KMP - ***K**UKA **M**obile **P**latform* (**P**lataforma **M**óvel da **K**UKA)

KMR - ***K**UKA **M**obile **R**obot* (**R**obô **M**óvel da **K**UKA)

Laser – ***L**ight **a**mplification by **s**timulated **e**mission **of r**adiation* (**A**mplificação da **L**uz por **E**missão **Es**timulada de **R**adiação)

LBR – ***L**eichtbauroboter* (referido para o braço robótico da KUKA)

LIDAR – ***L**ight **D**etection **a**nd **R**anging* (Deteção de Luz e Alcance)

LQR – *Linear Quadratic Regulator* (**R**egulador **L**inear-**Q**uadrático)

MCAL_P – *Mobile robot Collision Avoidance Learning with Path* (Algoritmo de evitar colisões para robótica móvel por aprendizagem com planeamento de trajetórias)

MIAR – *Mobile Intelligent Autonomous Robots*

MPC – *Model predictive control* (**M**odelo de **C**ontrolo **P**reditivo)

MRCA – *Most Relevant Common Ancestor* (**A**ncestral **C**omum **M**ais **R**elevante)

MuJoCo – *Multi-Joint dynamics with Contact* (Dinâmica de Múltiplos Pontos com Contacto)

PID – *Proportional Integral Derivative* (Controlador **P**roporcional **I**ntegral **D**erivativo)

PRM – *Probabilistic roadmap* (Planeador Probabilístico de Trajetórias)

PWM – *Pulse with Modulation* (Modulador por Largura de Impulso)

QR – *Quick Response* (Código de "resposta rápida")

RL – *Reinforcement learning* (Aprendizagem por Reforço)

ROS – *Robot Operating System* (**S**istema **O**perativo **R**obótico)

SDNL - **S**istema **D**inâmico **N**ão-**L**inear

TEB – *Timed Elastic Band* (**B**anda **E**lástica **T**emporizada)

ToF – *Time of Flight* (Tempo de Voo)

1. INTRODUÇÃO

Neste capítulo é realizado uma breve contextualização da problemática e objetivos envolventes ao projeto de dissertação. Para o efeito, é apresentado o enquadramento do problema e as motivações ao desenvolvimento da presente solução. Por sua vez, dado o enquadramento do projeto, são indicados os requisitos e objetivos, sintetizados face à proposta global de investigação. Por último, por forma a facilitar a leitura e compreensão, é apresentada a estrutura adotada para a dissertação.

1.1. ENQUADRAMENTO E MOTIVAÇÃO

Atualmente, evidencia-se uma proliferação e generalização de aplicações robóticas nos quatro principais campos de atividade: manufatura e logística, serviços e cuidados médicos, robótica de serviço e ainda em tecnologias emergentes, esta última relacionada com as mais recentes inovações tecnológicas (Christensen, 2012). Efetivamente, ao nível do setor industrial, os sistemas robóticos possibilitam o aumento da produtividade e qualidade do produto final (Sharma et al., 2019), reduzindo ainda os custos internos. A globalização do mercado económico e a constante instabilidade comercial são alguns dos fatores externos que moldaram o modo *operandi* das empresas, impondo a procura de novos métodos e alternativas que tornem o seu negócio mais competitivo e rentável (Ignatiev et al., 2016). Por norma, a estratégia incide no aumento da eficiência e produtividade dos principais processos industriais, recorrendo a mecanismos automatizados. Contudo, as operações de transporte e de manipulação de matéria-prima e produto acabado são ainda maioritariamente processos manuais (Porunov & Trifonov, 2020), originando uma estruturação de processos dependentes e isolados entre si. Consequentemente, devido ao assincronismo, verifica-se um escalamento de tarefas ineficiente, prejudicando o desempenho operacional devido ao subaproveitamento dos recursos disponíveis. Esta dissertação insere-se no projeto de investigação MIAR, realizada no âmbito da parceria Bosch, UMinho e CCG no qual é pretendido o desenvolvimento de uma solução inteligente e flexível que possibilite a partilha das máquinas ferramentas entre diferentes tipos de processos e que auxilie o transporte e posicionamento dos bens materiais entre as linhas de montagem e as máquinas ferramentas, e vice-versa, dentro de um determinado tempo estipulado. A integração dos processos industriais, conjugada por soluções inteligentes de planeamento, controlo e gestão, permite obter um maior desempenho operacional, devido à maior rastreabilidade, reduzindo os tempos de inatividade e, por consequência, os custos de produção.

Os processos podem ser automatizados por robôs ou manipuladores industriais, podendo estes ser classificados como estáticos ou móveis. Os manipuladores estáticos, dependendo da especificação e configuração, podem executar diversas tarefas, desde soldadura e montagem até trabalhos de processamento e manipulação (Frankovsky et al., 2018). Todavia, estes processos industriais necessitam de ser alimentados extrinsecamente, recorrendo a sistemas complementares ou a operadores humanos. O mesmo é válido para a extração dos produtos manipulados. Por sua vez, os manipuladores móveis têm a capacidade de agilizar os processos, tornando-os mais flexíveis e interligados. Por definição, um manipulador móvel é um produto que integra um braço robótico numa plataforma móvel, formulando um sistema robótico flexível e ajustável às diversas aplicações industriais (Mestiri & Gonçalves José Lima Mohamed Aymen Slim, 2021). A integração com o conceito de robótica colaborativa possibilita a partilha do ambiente de trabalho com os operadores humanos, de forma segura, alargando a área de operação do sistema conforme as necessidades operacionais. Conforme a configuração, estes tipos de robôs podem realizar tarefas típicas dos manipuladores estáticos, tipo tarefas de *pick and place* ou montagem, bem como de realizar operações de transporte, alimentação de máquinas ferramenta, de armazenamento, entre outras (Figura 1). Apesar da eficiência do transporte estar associada às características do chão de fábrica, a sua influência pode ser minimizada recorrendo a plataformas móveis omnidirecionais e por sistemas de planeamento de rotas conforme o congestionamento. As plataformas móveis do tipo omnidirecional apresentam diversas vantagens em termos de mobilidade, incluindo a capacidade de operação em espaços confinados, em ambientes congestionados e locais exíguos. As características holonómicas permitem definir trajetórias complexas, como o movimento lateral e de rotação sobre o seu centro de massa. Estas particularidades têm especial relevância em operações de acostagem às zonas de trabalho e de navegação em espaços estreitos, reduzindo significativamente o número de manobras necessárias e, consequentemente, o tempo de operação. No entanto, e apesar da gama mais alargada de aplicações possíveis, as questões de segurança relacionadas com a mobilidade não devem ser desconsideradas, especialmente no caso de operação em ambientes dinâmicos.

(a) Alimentação máquina CNC pelo manipulador móvel Star-L

(b) Manipulação de um objeto pelo manipulador móvel RB-KAIROS+

(c) Processo de polimento realizado pelo manipulador móvel *KUKA KMR QUANTEC*

Figura 1: Exemplos de manipuladores móveis.

1.2. OBJETIVOS

O trabalho de dissertação insere-se num projeto de investigação que pretende conceber soluções inteligentes, flexíveis e automatizadas que maximizem a eficiência dos processos industriais, minimizando a inatividade das operações de processamento e manipulação, devido à sincronia entre processos. Para o efeito, foi proposta uma solução de otimização da utilização das máquinas ferramenta, garantindo, por exemplo, a operação ininterrupta das mesmas, a sincronia entre processos e tempo de entrega ou a partilha das máquinas ferramenta por vários processos industriais. A proposta de investigação viabilizou a troca de informação em tempo real entre diversos pontos de operação e outros sistemas internos de planeamento ou gestão, podendo ser realizado um diagnóstico contínuo dos processos de operação, no caso de este ser complementado por sistemas de comunicação internos, *IIoT* (*industrial internet of things*) e por métodos de análise preditiva.

O projeto de investigação dividiu-se em quatro temas de trabalho, concebendo, cada um, módulos especializados em tarefas específicas aos objetivos descritos, entre as quais: a geração de movimentos de um braço robótico colaborativo; o planeamento de rotas (planeamento global) podendo ainda ser considerado alguns fatores internos (congestionamentos, gestão de uma possível frota, ...); um método de localização interior (preciso) do veículo no espaço (posição e orientação). Contudo, o principal objetivo

desta dissertação consistiu em capacitar um veículo omnidirecional de navegar e transportar bens materiais em ambientes industriais dinâmicos, de forma consciente da presença humana e eficiente. Para alcançar o objetivo descrito, pretendeu-se:

- Dotar o veículo da capacidade de navegar em ambiente industrial complexo, considerando o comportamento de seguir para os locais de trabalho ou linhas de montagem, bem como o de evitar colisões com obstáculos.

- Implementar um método que possibilite a execução de manobras de forma autónoma aos locais de acostagem (zonas de trabalho ou linhas de montagem). As manobras devem ser estabelecidas por movimentos holonómicos, minimizando, deste modo, o espaço necessário e tempo de manobra.

- Conceber um algoritmo de controlo e gestão dos comportamentos do sistema, em função do seu estado e do tipo de tarefa recebida.

Na tarefa de transporte, pretendeu-se que o veículo navegasse em chão de fábrica complexo, tendo a capacidade de evitar colisões com obstáculos estáticos e dinâmicos, bem como de navegar em direção a um local desejado. De facto, foi requerido que os movimentos estabelecidos fossem legíveis aos operadores humanos, isto é, fossem tais que estes compreendessem intuitivamente as intenções de manobra do veículo. Contudo, esta condição pode não ser garantida no caso de operação em espaços considerados como mais exíguos e, portanto, desafiantes. Considerou-se a operação em espaços mais exíguos quando o espaço de manobra impossibilitasse a definição da trajetória segundo movimentos não-holonómicos.

As tarefas de manipulação de objetos foram realizadas em espaços de trabalho previamente definidos, como por exemplo nas máquinas ferramentas ou linhas de montagem. O trabalho foi realizado por um braço manipulador colaborativo. Para o efeito, o veículo teve de se imobilizar numa *pose* que maximizasse o volume de trabalho do braço robótico, possibilitando uma operação mais eficiente. As manobras de aproximação e afastamento aos locais de trabalho foram realizadas de forma autónoma e precisa, sendo estabelecidas por movimentos holonómicos. O controlo dos movimentos do braço robótico encontra-se definido como objetivo de outro trabalho de investigação do projeto apresentado.

De modo a conceber um sistema de controlo capaz de assegurar as condições e requisitos de operação enunciados, a problemática dividiu-se em objetivos mais específicos e elementares, facilitando a sua análise e implementação. Numa primeira fase, a cinemática do veículo foi estudada considerando as características e restrições operacionais. Do mesmo modo, o sistema de deteção de obstáculos que equipa o veículo foi estudado, tendo em consideração as distâncias e a sua gama de sensorização. Numa

segunda fase, o foco de estudo incidiu no desenvolvimento dos algoritmos que implementassem o controlo e geração de movimentos em piso fabril dinâmico. Com o intuito de verificar a ação dos métodos, estes foram validados num *software* de simulação com motor de física, considerando o modelo dinâmico da plataforma móvel. Nesta iteração, os comportamentos que implementam as tarefas de transporte e acostagem foram validados de forma isolada para diferentes condições. Por forma a orquestrar os comportamentos descritos, foi estabelecido um mecanismo de gestão de execução de tarefas em função do estado do sistema. Nesta fase, o comportamento do sistema foi aferido como um todo, validando a execução das operações previstas num ambiente de simulação, tendo em atenção os requisitos de projeto e condições de segurança durante a navegação em espaços dinâmicos complexos.

1.3. Estrutura da Dissertação

A dissertação encontra-se organizada em 10 capítulos, formulando uma estrutura que contribui para uma melhor perceção e apreciação dos métodos e estratégias adotados para a conceção da solução proposta.

O capítulo 1 contextualiza e apresenta a motivação para o desenvolvimento do projeto de dissertação. Ademais, enuncia os requisitos necessários para o sistema proposto, descrevendo-os como objetivos a serem estabelecidos. Por sua vez, no capítulo 2, é apresentada a revisão literária das metodologias adotadas e desenvolvidas em projetos similares. O capítulo 3 descreve os fundamentos teóricos, sintetizando os métodos necessários ao desenvolvimento da solução proposta. O capítulo 4 enuncia a composição da solução, descrevendo tanto os componentes associados à plataforma móvel como também o *software* de mediação e de simulação. Neste capítulo é ainda apresentado o modelo cinemático do veículo. No capítulo 5 é descrito a arquitetura do sistema, enunciando e caracterizando os módulos e respetivos constituintes que o compõem. Em seguida, nos capítulos 6 e 7 são apresentados e descritos os algoritmos que implementam os comportamentos de navegação holonómica e de execução de manobras de acostagem, respetivamente. Por sua vez, no capítulo 8, é descrito a implementação e integração dos módulos que definem a arquitetura da solução. É ainda apresentado a composição do cenário de simulação utilizado como mecanismo de teste ao sistema. No capítulo 9 são exibidos os resultados obtidos nos testes realizados à solução proposta, considerando os requisitos e objetivos do projeto. Por último, no capítulo 10, é realizado uma breve discussão ao trabalho desenvolvido, destacando as principais características e conclusões do projeto. Neste capítulo são ainda indicadas algumas perspetivas a serem aprofundadas como trabalho futuro.

2. Estado da Arte

Este capítulo aborda alguns dos projetos existentes relacionados com a temática de navegação autónoma e processos de acostagem. Para cada temática são ainda apresentadas as principais características e desempenho de cada método exposto. Esta metodologia de análise permite identificar quais as metodologias mais indicadas para o controlo de navegação para veículos em ambientes dinâmicos, considerando a presença de seres humanos, de modo a permitir o desenvolvimento de uma nova técnica de controlo para veículos omnidirecionais.

2.1. Robótica Móvel na Indústria

O estudo da evolução industrial permite aferir que a inovação não é um conceito recente. Em contexto industrial, desde a revolução vivida a partir do século 18, a inovação verifica-se na procura de técnicas e mecanismos que permitam a otimização dos processos industriais. O paradigma atual complementa o conceito de inovação indicado, diferenciado pela consciencialização da segurança e bem-estar dos operadores humanos.

Por norma, a estratégia das indústrias incide na introdução de automatismos e sistemas robóticos que permitam aumentar a eficiência e produtividade nos principais processos industriais. Ao nível dos processos de logística interna, as indústrias recorrem a soluções de robótica móvel. Esta vertente da robótica teve início no ano de 1953 com a introdução do primeiro robô móvel, um AGV (*automated guided vehicles*) desenvolvido pela empresa Barrett Electronics (Vitolo et al., 2022). Apesar de não existir uma definição única para o termo "robô móvel" entre a comunidade, este é muitas vezes referido como um dispositivo capaz de navegar de forma autónoma num certo ambiente de modo a estabelecer um determinado objetivo (Shneier & Bostelman, 2015).

A robótica móvel encontra-se atualmente a ganhar cota nas aplicações industriais, comparativamente a outros tipos de sistemas robóticos. Isto deve-se à atual procura por soluções de manipulação de materiais em ambientes de dimensões consideráveis (Shneier & Bostelman, 2015). Em contexto industrial, os robôs móveis são tendencialmente introduzidos em processos de logística interna, como o transporte de matérias-primas e produtos acabados e a colaboração com operadores humanos. Além do mais, esta solução robótica permite uma melhor distribuição de tarefas, segundo o valor acrescido, dignificando o trabalho dos operadores humanos e, simultaneamente, otimizar os processos de produção garantindo ainda tempos de resposta (Gigante et al., 2022).

Dependendo do tipo de tecnologia, os robôs móveis são normalmente categorizados por AGVs ou por AMRs (*autonomous mobile robots*). Os AGVs, por definição, são sistemas internos estabelecidos por veículos controlados autonomamente sob um mecanismo de guia fixo cuja tarefa primária consiste na manipulação de materiais. A Figura 2 ilustra uma condição de navegação de um AGV em ambiente industrial.

Figura 2: Operação de transporte automatizada de produtos singulares em ambiente industrial. Retirado de Hilke (2022).

A ação de navegação dos AGVs pode ser implementada por diferentes tipos de infraestruturas auxiliares introduzidas no ambiente de navegação, dependendo do tipo de tecnologia adotada (Markis et al., 2019):

- **Método de orientação lateral** ou **por fio**: Este método recorre a sensores indutivos de modo a detetar a presença de um fio previamente embutido em chão de fábrica;
- **Orientação do tipo inercial:** Este método recorre a *transponders*, embutidos no solo do ambiente de navegação, para controlo da trajetória do veículo. A estratégia é complementada por giroscópios cuja função consiste na deteção de possíveis desvios do sistema. Os ímanes são colocados em locais estratégicos;
- **Orientação do tipo *laser*** ou **sistema de triangulação:** Este método utiliza um transmissor-recetor, integrado no próprio veículo para a sensorização dos pontos de referência refletores, estes estrategicamente posicionados no ambiente de navegação. A localização do veículo é estabelecida pelo método de triangulação.

Contudo, apesar de nos últimos 50 anos terem demonstrado a sua viabilidade e eficácia, a natureza intrínseca do sistema restringe a gama de aplicações em operações estabelecidas por rotas fixas e desobstruídas (Melo & Corneal, 2020). A Figura 3 apresenta a dimensão do mercado global referente à compra de AGVs no ano de 2019, por região.

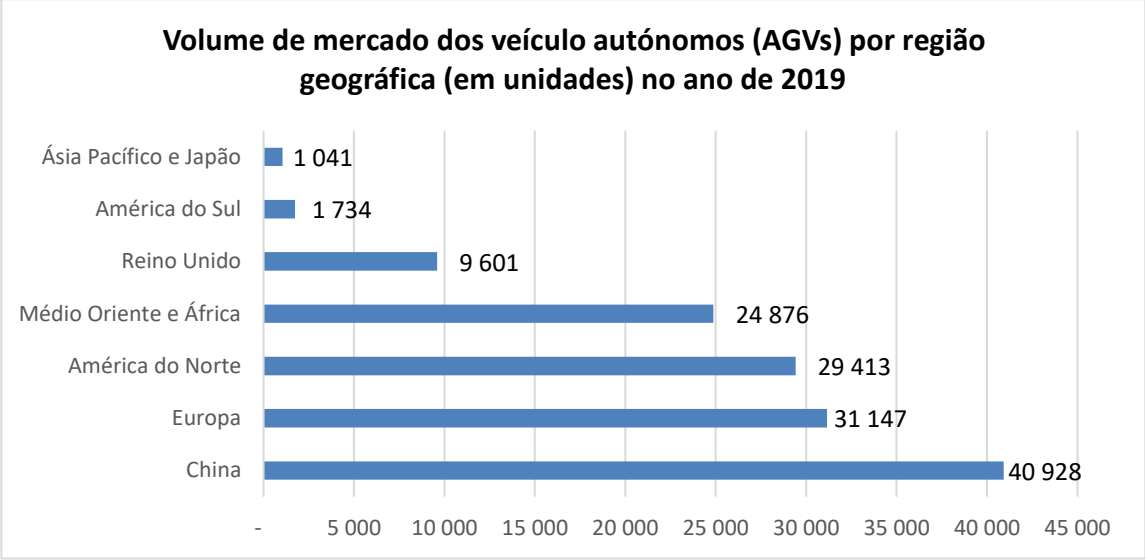


Figura 3: Volume de mercado dos robôs móveis do tipo AGV (em unidades) no ano de 2019 por região ou país. Dados publicados em janeiro de 2022 em Statista (2022).

Por sua vez, contrariamente aos AGVs, os AMRs correspondem ao estado da arte de investigação e desenvolvimento de soluções de logística interna (Markis et al., 2019). Estes tipos de sistemas têm capacidades adaptativas em tempo real (Melo & Corneal, 2020), sendo apropriados à navegação em ambientes dinâmicos (Papa et al., 2018), convergindo, deste modo, para os atuais requisitos de maior flexibilidade e agilidade associados aos processos industriais (Melo & Corneal, 2020). A flexibilidade associada torna o sistema mais rentável, dado que não requer a instalação de sistemas complementares externos, possibilitando ainda a incorporação em diversos processos industriais, isto é, não restringe a gama de operação (Melo & Corneal, 2020). A Figura 4 representa a operação de múltiplos AMRs numa tarefa de logística interna.

Figura 4: Operação de logística interna realizada por múltiplos robôs móveis do tipo AMR em ambiente dinâmico. Retirado de Hilke (2022).

Apesar de ainda não vulgarmente presentes na indústria, estão tendencialmente a serem introduzidos em ambientes industriais em tarefas de logística e manipulação de materiais (Papa et al., 2018). A Figura 5 ilustra o volume de mercado da robótica móvel nos anos de 2019 e 2020, sendo ainda apresentado as previsões de investimento até ao ano de 2025.

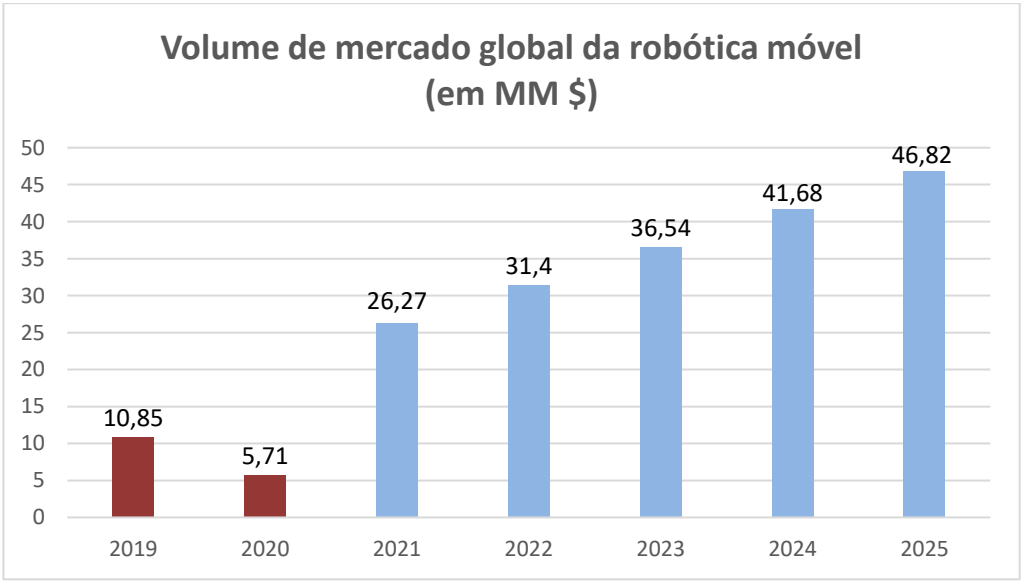


Figura 5: Volume de mercado da robótica móvel nos períodos de 2019 e 2020. Previsão de cota de mercado para os anos posteriores a 2021. Dados publicados em julho de 2021 em Statista (2022).

2.2. Controlo Navegação em Ambientes Dinâmicos

A temática de navegação autónoma tem-se tornado um tópico cada vez mais popular, tanto a nível de soluções industriais, como na execução de tarefas quotidianas. A nível industrial, as plataformas móveis autónomas permitem que o processo de logística seja mais eficiente e rentável (Louro et al., 2019). No quotidiano, os veículos autónomos permitem implementar soluções *follow me*, isto é, orientar seres humanos em ambientes por eles desconhecidos, como em aeroportos, permitem transportar bens alimentares em espaços de restauração, entre outras aplicações (Choi et al., 2021).

Existem diversas metodologias que implementam o algoritmo de navegação autónoma, no entanto, numa aplicação prática, todas estão dependentes de informação sensorial, quer a nível interno quer a nível externo. Os sensores internos permitem recolher informação relativa ao próprio veículo, como a sua *pose* no espaço e a velocidade (Choi et al., 2021). Por outro lado, os sensores externos permitem sensorizar o ambiente envolvente ao veículo, como aferir a distância a um obstáculo ou até a identificação do objeto (Choi et al., 2021) e (Louro et al., 2019).

Considerando que os veículos autónomos naveguem em espaços dinâmicos partilhados por operadores humanos, a segurança destes torna-se um requisito essencial (Louro et al., 2019). Como os seres humanos são considerados "seres não estáticos", torna-se essencial que o veículo execute movimentos legíveis durante a navegação (Stowers, 2018). Os movimentos legíveis permitem que o ser humano consiga compreender intuitivamente as intenções de movimento do veículo (Faria et al., 2021), e consequentemente aumentar o grau de confiança para com os veículos autónomos (Kruse et al., 2012).

2.2.1. Controlador *Fuzzy*

O controlo da navegação autónoma baseado nos controladores lógicos *fuzzy* tem estado em voga como alternativa de controlo. No entanto, segundo Basheer Essa et al. (2017), a grande maioria dos métodos propostos assumem a representação do robô como sendo um ponto, não consideram a dimensão nem a dinâmica do veículo, e usualmente analisam o comportamento do sistema em ambientes de simulação simplistas, com apenas obstáculos estáticos ou inexistentes. Cherni (2016) propôs um planeador de movimentos, baseado em controladores *fuzzy*, que consegue navegar em ambientes com obstáculos estáticos e dinâmicos.

A solução apresentada por Basheer Essa et al. (2017) consiste em implementar o controlo com base em dois controladores lógicos *fuzzy*, denominando o método por *fuzzy inference system*. O primeiro controlador foi projetado com a dinâmica de evitar obstáculos, recorrendo a informação sensorial. Com

base no *input* recebido, categorizado qualitativamente como "*near*" ou "*far*", a regra de controlo, estabelecida com base no conhecimento prévio do sistema, atribui o sentido e velocidade de rotação a cada roda motriz do veículo, conforme representado na Tabela 1. O segundo controlador *fuzzy* permite ao sistema convergir a direção de navegação para a orientação relativa ao alvo definido, minimizando a distância percorrida. O controlador categoriza o erro de orientação, o ângulo relativo à diferença entre a direção de navegação do veículo e a desejada. De forma similar ao primeiro controlador, a regra de controlo, estabelecida com base no conhecimento empírico, associa o erro de orientação ao *output* de cada roda motriz, conforme representado na Tabela 1. Como o comportamento do sistema não pode ser simultaneamente definido por ambos os controladores, foi apresentado um mecanismo de *switching* que seleciona qual dos controladores deve estar ativo, em função do estado. Na presença de obstáculos, apenas o primeiro controlador fica ativo, caso contrário, o controlo fica apenas estabelecido pelo segundo controlador. O comportamento do sistema foi testado e comparado com os resultados apresentados em Cherni et al. (2016). O método proposto por Basheer Essa et al. (2017) obteve melhor desempenho, estabelecendo um percurso ótimo e mais suave durante a navegação, em condições de simulação semelhantes.

Tabela 1: Regras base definidas à *priori* para o controlador que implementa o comportamento de orientar ao *target*. Adaptado de Basheer Essa et al. (2017).

Regra:	Variável de Entrada	Variável de Saída	
	θ_{dif}	$\omega_{direita}$	$\omega_{esquerda}$
1	"(erro reduzido)" > 0	"reverso lento"	"proceder lento"
2	"(erro significativo)" > 0	"reverso lento"	"proceder lento"
3	"(erro reduzido)" < 0	"proceder lento"	"reverso lento"
4	"(erro significativo)" < 0	"proceder lento"	"reverso lento"
5	"(erro nulo)"	"proceder"	"proceder"

2.2.2. Métodos Empíricos: Aprendizagem por Reforço em Redes Neuronais

A tarefa de navegação autónoma em ambientes dinâmicos revela ser exigente a nível de controlo devido aos requisitos e condições que devem ser considerados pelo controlador. As metodologias baseadas em informação empírica permitem simplificar as complexas considerações, simplificando a metodologia de controlo, face aos métodos tradicionais (Choi et al., 2021). O algoritmo *Collision Avoidance with Deep RL* (CADRL) capacita um sistema de identificar e evitar colisões com seres humanos (Chen YF et al., 2017).

A identificação depende da eficiência e eficácia do sistema de sensorização, obtendo informação relacionada com a posição, velocidade e tamanho. No entanto, o algoritmo limita a capacidade de evitar colisões a seres humanos que o sistema consiga identificar. Os restantes obstáculos, estáticos ou dinâmicos, não são evitados (Choi et al., 2021).

Por sua vez, a eficácia do algoritmo *Most Relevant Common Ancestor* (MRCA) não depende da identificação do tipo de obstáculo, mas apenas da sua deteção (Long P et al., 2018). Este algoritmo permite que a rede neuronal seja treinada com outros agentes, acelerando o processo de aprendizagem. Contudo, o comportamento prático pode ser diferente do analisado em simulação, uma vez que o MRCA não considera fatores dinâmicos (Choi et al., 2021).

O método proposto em Choi et al. (2021), *Mobile robot Collision Avoidance Learning with Path* (MCAL_P), foi baseado no MRCA. A aprendizagem da rede neuronal formula-se com base na informação sensorial *lidar*, na velocidade linear e angular e na distância ao alvo, definindo o estado do sistema. O mecanismo de recompensas orienta o comportamento do sistema conforme o requerido, pesando a componente de seguir em direção ao alvo, de evitar colisões com obstáculos e de respeitar os limites dinâmicos do veículo. O algoritmo de aprendizagem bonifica ou penaliza cada componente em função do comportamento de cada agente, conforme o mecanismo apresentado na Figura 6. O MCAL_P permite que o treino da rede neuronal seja partilhado por outros agentes, que partilham a mesma política de aprendizagem, e simultaneamente são considerados como obstáculos dinâmicos, acelerando o processo de aprendizagem.

$$\begin{cases} R_{goal} = +10;\ \text{se alcançou } target \text{ final} \\ R_{goal} = \text{Distancia}_{\text{anterior}} - \text{Distancia}_{\text{atual}};\ \text{caso contrário} \end{cases} \quad \left| \quad \begin{cases} R_{collision} = -10;\ \text{se ocorreu colisão} \\ R_{collision} = 0;\ \text{caso contrário} \end{cases} \right. \quad \left| \quad \begin{cases} R_{limits} = -0{,}1\lfloor\omega\rfloor;\ \text{se } \lfloor\omega\rfloor > 0{,}6 \\ R_{limits} = 0;\ \text{caso contrário} \end{cases} \right.$$

Figura 6: Mecanismo de recompensas implementado. Adaptado de Choi et al. (2021).

Devido ao conflito de interesses entre a componente de seguir em direção ao alvo e de evitar obstáculos, o sistema pode ser incapaz de alcançar o alvo final. O problema foi identificado e solucionado recorrendo a um planeador global, que discretiza o percurso global por vários *targets* intermédios (Choi et al., 2021). O comportamento do MCAL_P foi testado e comparado com os planeadores *Timed Elastic Band* (TEB) e *Dynamic Window Approach* (DWA), já implementados no *Robot Operating System* (ROS), em ambiente de simulação diferente do que a rede neuronal foi treinada, diferindo ainda o tipo de obstáculos e o seu movimento. A estratégia de fuga aos obstáculos difere significativamente, o MCAL_P tende a desviar-se dos obstáculos mantendo a velocidade linear constante, enquanto ambos os planeadores optam por diminuir a velocidade linear. A estratégia de diminuir a velocidade linear torna-se

ineficiente em obstáculos dinâmicos que se movimentam em rota de colisão com o veículo, tornando o método proposto em Choi et al. (2021) mais eficaz a evitar colisões com obstáculos dinâmicos. Os planeadores locais DWA e TEB são mais eficientes em ambientes compostos por obstáculos estáticos, optando por reduzir a velocidade linear e desviar a orientação de navegação do obstáculo. Durante a navegação, o MCAL_P apresenta movimentos bruscos, mesmo na ausência de obstáculos, não sendo adequado como método de controlo em veículos que transportem cargas sensíveis.

2.2.3. PLANEAMENTO *PROBABILISTIC ROADMAP*

O planeamento de percursos tendo em conta o formato 3D do veículo e do espaço envolvente revela ser útil, principalmente quando o veículo autónomo transporta carga que exceda o seu tamanho e consequentemente exista a possibilidade de colisão com obstáculos. Para o exemplo exposto, poder-se-ia recorrer a métodos tradicionais e considerar um veículo de dimensão coincidente com a carga transportada (Wang et al., 2021). Contudo, esta abordagem pode restringir o movimento do veículo desnecessariamente, limitando por exemplo a passagem entre obstáculos que poderia facilmente ultrapassar.

Contudo, os planeamentos que consideram a forma 3D apresentam problemas relacionados com a eficiência de cálculo, exigindo uma maior carga computacional. Existem algumas técnicas que permitem acelerar o tempo de cálculo, baseando-se, por exemplo na redução do número de amostras da posição do veículo recorrendo a regras heurísticas (Biao H et al., 2021) ou a redes neuronais (Li X et al., 2019). No entanto, estas técnicas demonstram ser pouco viáveis numa aplicação robótica de navegação autónoma em espaços partilhados por operadores humanos (Wang et al., 2021).

Em Murray et al. (2016) foi proposto um método que realiza o planeamento de trajetórias em tempo real, considerando informação 3D, mas para manipuladores fixos. O método recorre a um *roadmap* probabilístico (PRM), previamente criado. O PRM é estabelecido tendo em conta os *nodes*, que representam a *pose* 3D do manipulador, os *edges* que representam o percurso entre dois *nodes* e os *swept volumes*, que representam o movimento discretizado associado a cada *edge* (Murray et al., 2016).

O método proposto em Wang et al. (2021) consiste em adaptar a abordagem PRM, projetada para manipuladores fixos, a manipuladores móveis. A cada posição possível do veículo corresponde a oito possíveis *nodes*, ou seja, a posição mantém-se fixa, mas a orientação pode ser alterada com uma resolução de quarenta e cinco graus. Os *edges* apenas podem interligar *nodes* vizinhos, correspondendo sempre a um segmento de reta. O movimento do veículo está limitado a ser linear ou rotacional, não

ambos em simultâneo. Os *swept volumes* são calculados com base nos *edges* gerados. De modo a minimizar o tempo de cálculo, é introduzido um sistema hierárquico de PRMs, a três níveis. Um *hub*-PRM, que corresponde ao PRM gerado, e dois *sub*-PRM, o *front sub*-PRM e o *rear sub*-PRM que corresponde respetivamente à posição do veículo e posição final desejada. Os dois *sub*-PRM correspondem a uma rede quadrada do *hub*-PRM, obtendo uma maior resolução de movimento.

Como pode existir sobreposição de *edges* com obstáculos, foi também introduzido um algoritmo que verifica a possibilidade de existência de colisões. Este algoritmo está sempre ativo durante a fase de navegação. Caso um obstáculo seja detetado, o algoritmo verifica a existência de algum *edge* que o sobreponha, descartando-o.

O planeamento do percurso ocorre em duas fases. A primeira fase consiste em verificar quais os *nodes* mais próximos da posição do veículo e da posição *target*, no *hub*-PRM. O percurso entre esses *nodes* é estabelecido apenas pelos *edges* que não estejam sobrepostos a obstáculos. Embora a rota esteja estabelecida entre os *nodes* inicial e final, a posição inicial do veículo e a posição final pode não coincidir com a posição dos nodes no *hub*-PRM. Por essa razão, numa segunda fase determina-se com maior precisão os *edges* entre a posição inicial do veículo e o *node* inicial no *hub*-PRM e, de forma similar entre o *node* final no *hub*-PRM e a posição alvo, no nível hierárquico correspondente ao *sub*-PRM. A Figura 7 idealiza um possível planeamento de um percurso, com base na informação descrita anteriormente.

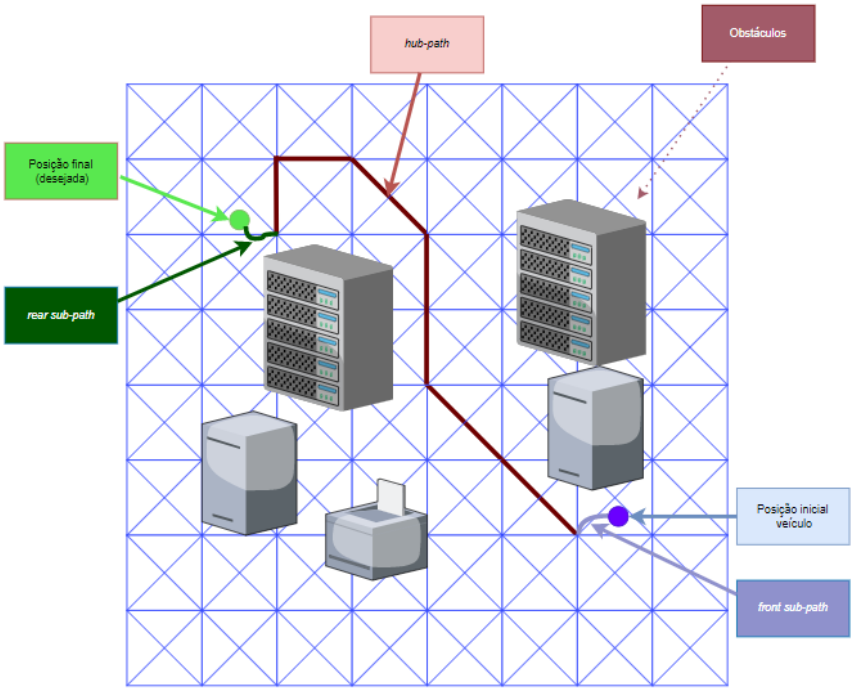


Figura 7: Exemplificação 2D do planeamento de percursos. Adaptado de Wang et al. (2021).

O comportamento do sistema foi verificado numa aplicação prática, acoplando ao veículo autónomo uma carga, disposta na horizontal, definindo, portanto, um veículo complexo. O ambiente foi composto por obstáculos cilíndricos, dispostos na vertical, com diferentes alturas. Apenas alguns destes obstáculos permitem que ocorra colisão com a carga acoplada ao veículo. O sistema demonstrou ser capaz de evitar a colisão, quer diretamente com o veículo, assim como com a carga. A cada obstáculo detetado, com possibilidade de colisão, o sistema recalcula o percurso, atribuindo novos *edges*. No teste realizado, o sistema demorou cerca de cem milissegundos a recalcular o percurso, mas num computador com elevada capacidade de processamento, o que impossibilita a implementação deste método num controlador com capacidade de processamento mais modesta.

2.2.4. Planeamento Híbrido

O tipo de controlo de *path tracking* difere entre os vários métodos de controlo existentes, no entanto, a maioria das técnicas recorre ao planeamento de percurso para orientar a direção de navegação dos veículos para o alvo final, principalmente em ambientes complexos e de grandes dimensões, como o verificado em (Choi et al., 2021) e (Louro et al., 2019).

O planeador de trajetórias pode subdividir-se em dois planeadores, o planeador global e o planeador local (Zhong et al., 2020). O planeador global requer conhecimento prévio do ambiente e assume que este seja estático. Por outro lado, para o planeador local o espaço envolvente não é conhecido *à priori*, permitindo implementar o comportamento de evitar obstáculos dinâmicos recorrendo à informação sensorial recolhida (Basheer Essa et al., 2017). Esta característica é fundamental para uma implementação prática, em ambientes dinâmicos complexos.

Um planeamento híbrido, que combina ambos os planeadores como método de controlo de navegação, foi exposto em Zhong et al. (2020). O método modifica o planeador global *A-Star* (A*) com o intuito de apenas definir percursos considerados como "seguros". Para tal, o algoritmo A* *safe* foi concebido, com base na configuração *C-Space*, alterando a função de custo do algoritmo original. A função considera o custo entre a posição inicial, *start point* e a posição do veículo, o custo estimado entre a posição do veículo e o nó final, o *goal point*, e o valor de risco do nó que o veículo se encontra na rede do algoritmo A*. Como é desejado que o veículo realize manobras suaves, em Zhong et al. (2020) foi introduzido uma técnica que extrai os "pontos chave" entre os nós que compõe o mapa. A técnica avalia o risco dos nós que definem o percurso projetado, selecionando aqueles que apresentarem menor

risco. Deste modo, o percurso fica definido por um subconjunto de "pontos chave", que o planeador local interpreta como alvos intermédios.

O planeador local implementa a tarefa de evitar colisões com obstáculos dinâmicos e, simultaneamente realiza o *path tracking* do percurso definido pelo planeador global. O algoritmo verifica as possíveis orientações de navegação "navegáveis", sem obstáculos, com base na dimensão do veículo e na informação sensorial. O método proposto por Zhong et al. (2020) prevê a possibilidade de existir múltiplas orientações que cumpram o requisito enunciado, sendo necessário definir um critério de seleção, com base no comportamento desejado. O comportamento é definido pela contribuição das componentes da direção ótima, da direção segura e da direção de navegação que requer menor esforço de manobra (Figura 8). Está associado um peso a cada parâmetro, por forma que seja possível moldar o comportamento do veículo. Com o intuito de adaptar a área de sensorização considerada na tarefa de seleção, foi implementado uma janela que se ajusta em função de cada situação, estabelecendo uma relação com uma referência calculada com base na direção de navegação e direções "navegáveis". Caso a referência seja considerada como "grande", a janela aumenta, caso contrário diminui.

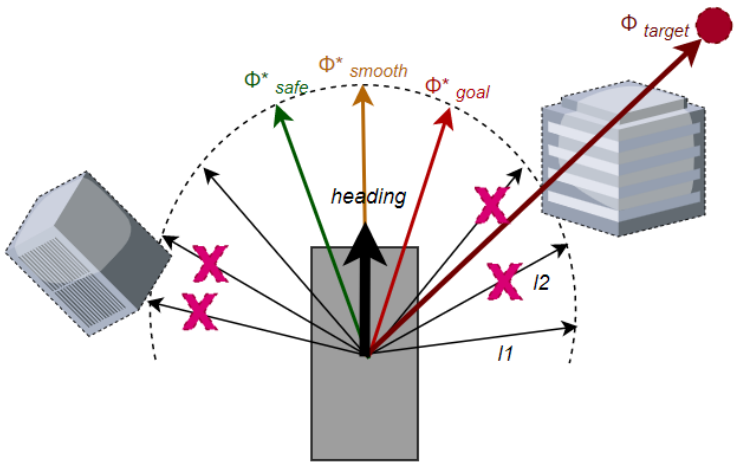


Figura 8: Seleção da direção de navegação. Adaptado de Zhong et al. (2020).

O comportamento do veículo foi verificado em simulação, num ambiente composto por obstáculos estáticos e dinâmicos. Embora o planeador global não estabeleça o trajeto mais curto, devido à alteração do algoritmo A*, o sistema conseguiu navegar em direção ao *goal point* e evitar colisões com os obstáculos detetados, devido ao *path tracking* e ao planeador local, respetivamente. Contudo, apesar do método apresentar resultados promissores em condições de simulação, o seu desempenho não foi validado em contexto real de navegação.

2.2.5. Sistemas Dinâmicos Não-Lineares

A evolução tecnológica experienciada nos últimos anos permitiu o desenvolvimento de plataformas móveis que naveguem, de forma autónoma, em ambientes dinâmicos complexos e partilhem o espaço com seres humanos (Louro et al., 2019). Os veículos autónomos permitem obter um centro de logística estruturado e organizado, beneficiando da flexibilidade, rastreabilidade e integração com os mecanismos de gestão (Louro et al., 2020).

Como já referido, um planeador local permite recalcular o percurso, previamente especificado pelo planeador global, a uma determinada taxa de atualização enquanto o veículo se move. Nestes tipos de planeadores, o mapa relativo ao ambiente envolvente fica delimitado pela distância de sensorização. Desse modo, um planeador local pode conjugar o comportamento de evitar obstáculos (estáticos ou dinâmicos) com o comportamento de seguir a trajetória previamente definida pelo planeador global, através dos alvos intermédios (Marin-Plaza et al., 2018). A estratégia de controlo de navegação autónoma apresentada em Louro et al. (2019) e Louro et al. (2020) consiste em desenvolver um planeador local, recorrendo a sistemas dinâmicos não-lineares. Contudo, foi também necessário conceber um planeador global que permita estabelecer uma rota ótima, entre a posição do veículo e a posição final desejada, assim como de evitar rotas que estejam mais congestionadas, caso o sistema seja composto por múltiplos veículos agentes.

Em Louro et al. (2019) e Louro et al. (2020) foi proposto um método de navegação autónoma em ambientes industriais dinâmicos complexos, onde a presença humana foi considerada. A técnica de controlo exposta minimiza os recursos necessários à implementação prática em chão de fábrica. A navegação foi definida recorrendo a sistemas dinâmicos não-lineares, que controlam as variáveis de velocidade linear e direção de navegação.

A dinâmica de navegação em Louro et al. (2019) é gerada pela contribuição da dinâmica do alvo, relativo ao comportamento de se dirigir para o alvo, da dinâmica de evitar colisões com obstáculos e com os operadores humanos. O comportamento do sistema é modulado pelos pesos atribuídos a cada parâmetro, na regra de controlo. Apesar da liberdade da parametrização, é fundamental que a componente relativa à dinâmica de evitar obstáculos seja prioritária, isto é, que tenha maior contribuição na dinâmica. A Figura 9 ilustra uma possível representação dos campos vetoriais que definem o controlo da orientação de navegação. Por sua vez, a velocidade linear pode ser definida pelas três componentes enunciadas, no entanto apenas uma pode influenciar a dinâmica. Caso o sistema não detete a presença de obstáculos, a dinâmica da velocidade linear define-se pela distância ao alvo, caso contrário, a dinâmica fica definida em função da distância ao obstáculo. Tendo em consideração as restrições de navegação e

limites dinâmicos do veículo, a velocidade linear e angular associadas ao *output* do sistema dinâmico encontram-se limitadas. Dado que os campos vetoriais variam em função da informação sensorial obtida, é essencial que os parâmetros de controlo, que definem as dinâmicas de navegação, estejam ajustados de forma que as variáveis de controlo convirjam para soluções assintoticamente estáveis (Louro et al., 2019).

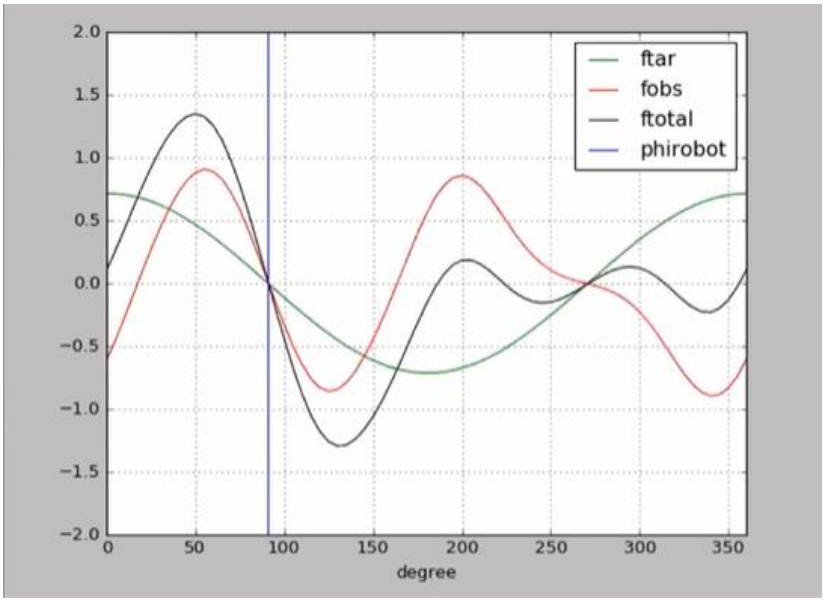


Figura 9: Possível representação dos campos vetoriais, em função da orientação de navegação. *ftar* representa o comportamento de seguir para o alvo, *fobs* o comportamento de evitar obstáculos, *ftotal* o somatório dos comportamentos individuais do sistema dinâmico não-linear (no caso *ftar* + *fobs*) e *phirobot* a direção de navegação atual do veículo.

Como os métodos propostos em Louro et al. (2019) e Louro et al. (2020) foram concebidos para uma aplicação real industrial, foi implementado um algoritmo de gestão de controlo de navegação em diversos módulos. O gestor de serviços atende aos pedidos de execução de tarefas requerido pelo servidor. O pedido é dividido por um conjunto de tarefas mais simplificado. Por sua vez, o gestor de tarefas interpreta o conjunto de tarefas e, com base no requerido, executa o planeamento global do percurso, atribuindo uma sequência de *via points*. O módulo de navegação recebe os *via points* sequencialmente, interpretando-os como alvos intermédios. Este módulo implementa a dinâmica do sistema não-linear, com base nos alvos intermédios, da *pose* do veículo em relação ao ambiente e na informação sensorial. Tendo em consideração a natureza crítica do processo de navegação autónoma em espaços partilhados por operadores humanos, foi ainda implementado um módulo que monitoriza as ações de controlo fornecidas pelo módulo de navegação, limitando-as caso os requisitos de segurança não se verifiquem.

O método proposto incide na integração do método exposto em Louro et al. (2019) com a capacidade de realizar manobras de acostagem, Louro et al. (2020). Na aplicação, a dinâmica de controlo da orientação pode ser definida pela dinâmica do alvo, de forma idêntica ao verificado em Louro et al. (2019), assim como pela dinâmica de seguir paralelamente a uma parede. A dinâmica de seguir em direção ao alvo encontra-se ativa na fase de navegação, entre os *via points*. Por sua vez, a dinâmica de seguir de forma paralela a uma parede fica ativa quando o sistema inicia o processo de *docking*. No caso específico de aplicação exposto, o comportamento de evitar obstáculos foi anulado para a dinâmica de controlo da direção de navegação, mantendo-se no controlo da velocidade linear. Deste modo, o veículo não se desvia de obstáculos, mas na presença de obstáculos em rota de colisão com o veículo, este para. Esta foi uma imposição da empresa onde os veículos iriam ser aplicados em chão de fábrica. No entanto, a dinâmica de controlo permite que o sistema seja também definido pelo comportamento de evitar obstáculos. O controlo da velocidade linear é estabelecido pela dinâmica do alvo, dos obstáculos e de seguir paralelamente a uma parede. Por sua vez, a dinâmica de seguir paralelamente a uma parede capacita o veículo de navegar de forma linear e a uma distância constante a uma parede. Este comportamento foi implementado com o intuito de auxiliar a manobra de acostagem, uma vez que foi considerado um veículo triciclo.

Nesta técnica, o veículo encontra-se capacitado de realizar a navegação em espaços complexos dinâmicos e de realizar manobras de acostagem. Embora estas tarefas sejam controladas pelo sistema dinâmico não-linear, nas manobras de acostagem foi necessário adaptar o comportamento do veículo em função da operação que esteja a realizar, recorrendo para tal a uma máquina de estados. As manobras de acostagem são constituídas por um conjunto de operações elementares, facilmente controladas por máquinas de estado, sendo exemplo a operação de alinhamento do veículo com a parede, a rotação do veículo para alinhar-se com a palete, ou o alinhamento preciso com a palete em função dos dados provenientes do sistema de visão, sendo que em cada estado, são os sistemas dinâmicos não-lineares que controlam a navegação do veículo.

Em ambos os casos, o comportamento do sistema foi analisado em simulação e numa aplicação prática industrial. Para tal, foi definido previamente uma bateria de testes a realizar. Em Louro et al. (2019) foi analisado o desempenho do veículo a executar a operação de navegação autónoma, considerando, numa primeira fase, um ambiente com obstáculos estáticos e, posteriormente com obstáculos estáticos e dinâmicos. Dado que o veículo obteve uma boa performance, foi ainda verificado o comportamento do veículo na presença de operadores humanos na trajetória. Os resultados obtidos comprovaram que o sistema obteve um bom desempenho, nas três dinâmicas que definem o campo

vetorial resultante, que o possibilita de compartilhar o chão de fábrica com operadores humanos, enquanto realiza a tarefa de transporte. De modo similar, em Louro et al. (2020) foi verificado o comportamento do sistema, embora mais focado nas manobras de *docking*. Foram definidos testes práticos, em chão de fábrica, que possibilitaram a análise das tarefas que constituem a máquina de estados, assim como de verificar o comportamento do sistema em situações mais instigantes à falha. Os resultados demonstram que o método permitiu realizar as tarefas de *docking* e de navegação de forma segura e estável. Caso seja detetado algum impedimento, como a presença humana na fase de acostagem, o veículo permanece imóvel até que sejam verificadas todas as condições de segurança que permitam o retorno da manobra.

2.3. Controlo do Movimento em Veículos Omnidirecionais

Os requisitos exigidos pelos clientes e a atual globalização industrial dos mercados, influenciada pela concorrência, ditam a necessidade de mudança na gestão dos processos de fabrico. Os veículos autónomos permitem agilizar a logística de transporte de bens materiais, tornando o processo mais flexível (Jie Zhang & Xiaobo Liu-Henke, 2020). A maior flexibilidade e manobrabilidade apresentada pelos veículos omnidirecionais permite que seja possível realizar manobras em espaços mais exíguos (Qian Jia et al., 2019). Estes veículos, devido às rodas mecânicas, apresentam três graus de liberdade, referentes ao movimento lateral, longitudinal e à rotação sobre o centro de massa, possibilitando a realização de movimentos em qualquer direção e orientação (Han & Zhu, 2019).

2.3.1. Algoritmo *Fuzzy* Proporcional, Integral e Derivativo

A complexidade associada à dinâmica de um veículo omnidirecional e as características do ambiente dinâmico envolvente ao veículo impossibilitam a obtenção de um modelo matemático preciso, que defina com rigor todo o comportamento do sistema. Consequentemente, implementar o controlo do movimento com base no controlador Proporcional-Integral-Derivativo (PID) clássico torna-se inapropriado, devido aos parâmetros de controlo serem fixos (Wen & Tong, 2017).

A solução apresentada em Wen & Tong (2017) e em Qian Jia et al. (2019) consiste em implementar um sistema de controlo que combine o controlo *fuzzy* e a estrutura de controlo PID. Deste modo, o controlador apresenta a flexibilidade e adaptabilidade associada ao controlo *fuzzy*, assim como a robustez e estabilidade do controlo PID clássico (Qian Jia et al., 2019).

O método proposto por Qian Jia et al. (2019) consiste em conceber um controlador *fuzzy* que regula os parâmetros de controlo PID em função das condições verificadas e requisitos impostos. O controlador *fuzzy* recebe como *input* o erro de desvio, isto é, a diferença entre a posição requerida e a posição do veículo num dado instante, e a taxa de variação do desvio. O ajuste de cada parâmetro de controlo PID define-se por um conjunto de regras base *fuzzy*, definidas *à priori* com base na experiência, que relacionam os *inputs* com o respetivo *output fuzzy*. Considerando o ajuste dos três parâmetros de controlo, a regra *fuzzy* foi implementada com base em três tabelas distintas, uma para cada parâmetro de controlo. O *output* do controlador *fuzzy* corresponde à regulação de cada parâmetro em função dos parâmetros iniciais, ou seja, reflete o incremento ou decremento a realizar a cada parâmetro inicial do controlador PID (Figura 10). Similarmente ao controlador PID clássico, os parâmetros iniciais devem ser determinados, com base na experiência, de modo a minimizar o erro inicial. O controlador PID, com base no erro de desvio, determina a velocidade linear e angular que permita ao sistema convergir para o erro nulo. Posteriormente, a velocidade angular a aplicar a cada roda motriz do veículo é calculada em função das equações que definem a cinemática do veículo omnidirecional.

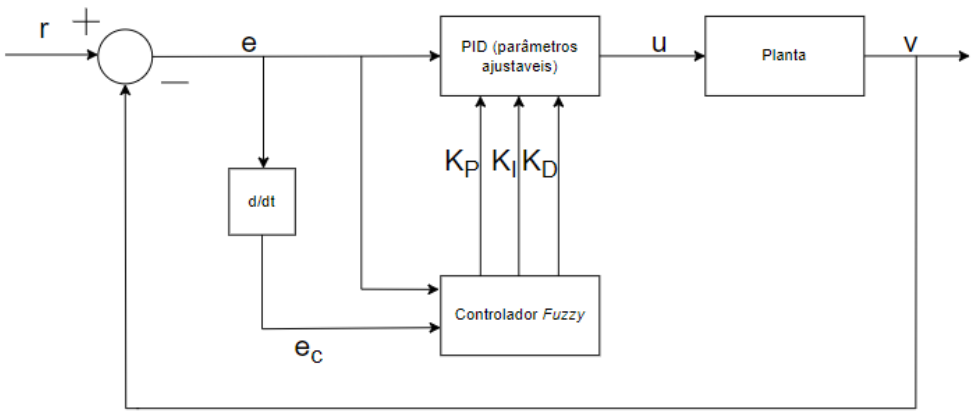


Figura 10: Representação sistema de controlo. Adaptado de Qian Jia et al. (2019).

O comportamento do método de controlo proposto foi analisado numa experiência prática, considerando um ambiente simples, sem obstáculos. O sistema apresentou inicialmente um desvio de dez centímetros em relação à trajetória retilínea requerida. Os parâmetros iniciais de controlo foram determinados em função do comportamento verificado em simulação. O veículo convergiu para a trajetória desejada após percorrer cerca de um metro. Como método de comparação de desempenho, foi realizada uma outra experiência, mas sob a regra de controlo PID clássico (parâmetros fixos), partilhando os parâmetros iniciais de controlo e o erro inicial da experiência anterior exposta. Nesta

situação, o mesmo veículo necessitou de percorrer três metros e meio para convergir para a trajetória imposta. Com base nos resultados, foi possível concluir que o sistema apresentou uma resposta mais estável e rápida sob o método proposto em Qian Jia et al. (2019), conferindo a adaptabilidade do comportamento do sistema.

A metodologia de controlo proposta em Wen & Tong (2017) apresenta uma solução de integração de um controlo *fuzzy* PID, semelhante ao presente em Qian Jia et al. (2019), com um planeador de trajetórias, previamente concebido. O planeador foi implementado tendo por base o planeador global A*. Este planeador executa um algoritmo de pesquisa do percurso ótimo em função da distância heurística entre dois nós da rede que compõe o mapa (Wen & Tong, 2017). No entanto, numa aplicação prática de robótica móvel autónoma, o algoritmo A* demonstra ser pouco eficiente, devido ao elevado tempo de computação, e não considera a distribuição geométrica dos nós, resultando numa atuação *on/off* do atuador. Consequentemente, o método propõe alterar a granularidade de pesquisa do algoritmo, de modo a acelerar o processo de calculo, assim como alterar os pesos associados aos nós que definem ângulos retos entre si, de noventa graus, na função que define o algoritmo A*. Segundo Wen & Tong (2017), após a modificação do algoritmo, o sistema reduziu o tempo de navegação em cinquenta porcento.

O controlador *fuzzy* PID estabelece o movimento do veículo com o objetivo de convergir a posição do veículo à posição imposta pelo planeador. O controlador *fuzzy* deriva os parâmetros de controlo PID em função de dois *inputs*, a grandeza de desvio e o gradiente do desvio. A regra *fuzzy* relaciona o valor qualitativo dos *inputs* com a regulação dos parâmetros de controlo, adaptando a resposta do sistema. Contudo, embora a solução *fuzzy* apresente resultados satisfatórios em desvios considerados como pequenos, tal não se verifica quando o veículo apresenta um erro de desvio elevado. A limitação do desempenho está associada às funções associadas *fuzzy*, por serem estáticas.

Por forma a mitigar a problemática, foi proposto um algoritmo genético no controlador *fuzzy* PID que adapte as funções associadas (Wen & Tong, 2017). O algoritmo codifica o conjunto de ações *fuzzy* em números binário, em que a cada número binário corresponde uma única função associada. As funções associadas têm de ser do tipo triangulares, de modo a garantir que esteja associado um *output* para cada *input*. No entanto, o formato pode variar, especificando os vértices num determinado domínio. Nas operações genéticas apenas os indivíduos "mais bem codificados" conseguem evoluir. Numa primeira etapa, verificasse a "aptidão" de cada indivíduo, selecionando os "mais aptos". Na segunda etapa ocorre o cruzamento, recorrendo ao *crossover operator*. Este operador deve assegurar que as características dominantes da geração anterior sejam herdadas na nova geração. Para tal, ocorre o cruzamento de

valores no *loci* (na posição de um gene num cromossoma), de forma apropriada, que correspondem à interseção dos *locus* (singular de *loci*) selecionados dos dois indivíduos pais, de forma aleatória. O operador de mutação é apenas usado como um operador auxiliar, que permite alterar certas características resultantes da operação de cruzamento que não sejam desejadas. Por fim, após verificar a condição de convergência, origina-se o novo indivíduo. A Figura 11 idealiza um fluxograma alto nível que elucida as iterações do método apresentado. De acordo com Wen & Tong (2017), após a implementação do algoritmo genético, o sistema obteve um comportamento mais estável, conseguindo convergir rapidamente para a rota desejada.

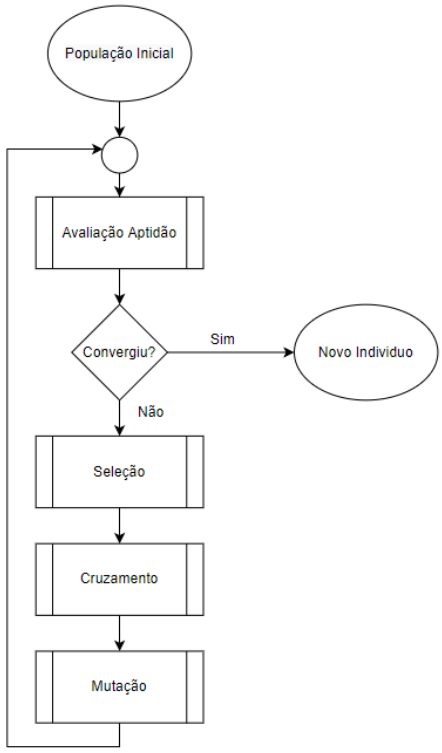


Figura 11: Fluxograma Algoritmo Genético. Adaptado de Wen & Tong (2017).

2.3.2. Controlador Lógico *Fuzzy*

O controlo de movimento para veículos autónomos deve garantir que o sistema consiga convergir para o percurso previamente definido. No entanto, para certas aplicações, a orientação de navegação durante o movimento revela ser também um aspeto relevante. Apesar das enumeras vantagens associadas ao veículo omnidirecional, estes, devido ao *design* das rodas mecânicas, estão sujeitos a ligeiras derrapagens durante o movimento (Fahmizal & Kuo, 2016).

O método proposto em Fahmizal & Kuo (2016) recorre ao controlador lógico *fuzzy* (FLC) e a sensores inerciais *IMU* para capacitar um veículo omnidirecional de percorrer uma trajetória previamente definida, e de corrigir a direção de navegação caso ocorra uma derrapagem, com principal destaque neste último comportamento. O sistema recorre aos dados fornecidos pelo *IMU* para determinar a direção de navegação. O método estabelece um filtro complementar que combina os dados do acelerómetro e giroscópio. Para o efeito, foram utilizados filtros digitais passa-baixo e passa-alto, respetivamente. Posteriormente, os dados são combinados com os fornecidos pelo sensor magnético, obtendo a orientação de navegação, o *yaw*. O erro de desvio corresponde à diferença entre a *pose* requerida pela trajetória definida e a *pose* do veículo, para um determinado instante (Figura 12).

O FLC implementa a regra de controlo que permite corrigir a orientação do veículo, mas ajustada para veículos de direção diferencial. A implementação do FLC segue os procedimentos *standard*, transformando um *crisp input* num *crisp output*. A regra *fuzzy* concebida em Fahmizal & Kuo (2016) relaciona o erro obtido e o erro anterior, relativos à diferença entre a orientação requerida pela trajetória e a obtida pelo *IMU*, estabelecendo um total de vinte e cinco regras, em termos linguísticos. O processo de *defuzzification* permite obter o *crisp output*, que corresponde ao valor do *duty cycle* a aplicar ao modulador por largura de impulso (PWM – *pulse with modulation*) que controla a velocidade de rotação de cada motor.

O comportamento da plataforma móvel foi analisado em ambiente de simulação simples, sem qualquer obstáculo (Figura 12), estabelecendo uma trajetória circular. Com base nos resultados presentes em Fahmizal & Kuo (2016) foi possível concluir que o erro de *tracking* do sistema foi ligeiramente inferior ao verificado caso o sistema de controlo fosse estabelecido unicamente pela cinemática do veículo omnidirecional. O veículo manteve a orientação de navegação constante, executando a manobra circular recorrendo unicamente aos dois primeiros graus de liberdade do veículo. O terceiro grau corresponde ao ajuste da orientação, caso ocorra derrapagem. O método apresentado em Fahmizal & Kuo (2016) não apresenta uma solução de implementação em ambientes dinâmicos, com obstáculos dinâmicos.

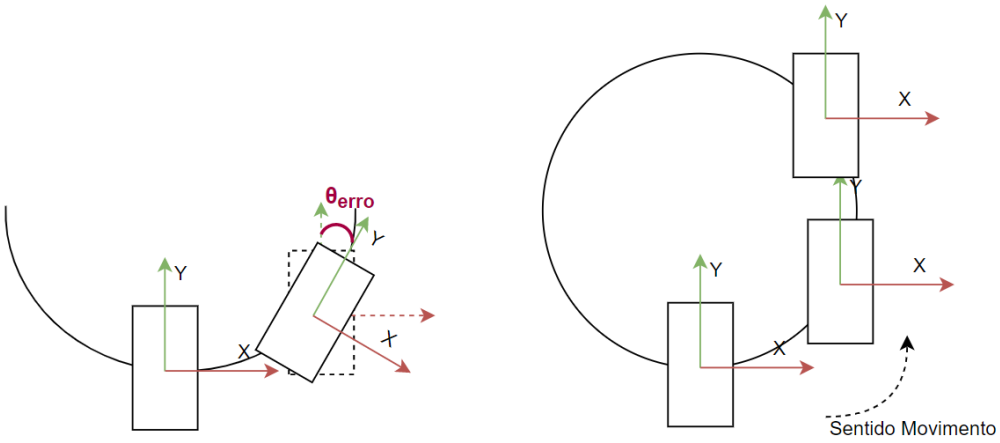


Figura 12: Ajuste da direção de navegação. Adaptado de Fahmizal & Kuo (2016).

2.3.3. Controlo Ótimo

Numa aplicação prática, o movimento do veículo deve ser definido tendo em consideração as restrições e condições impostas de movimento, como por exemplo executar movimentos que minimizem o consumo de energia (Jie Zhang & Xiaobo Liu-Henke, 2020) ou que respeitem os limites estruturais (Han & Zhu, 2019).

A abordagem presente em Jie Zhang & Xiaobo Liu-Henke (2020) propõe um sistema de controlo de movimento para veículos omnidirecionais, com base na sua modelização. Para tal, foi necessário deduzir a cinemática, obtendo uma relação entre a velocidade de rotação de cada roda motriz e a velocidade linear e angular do veículo, assim como deduzir a dinâmica, estabelecendo uma relação entre o binário de cada motor e a aceleração de cada roda. Como a regra de controlo define-se pelo controlador *linear–quadratic regulator* (LQR), foi necessário representar o comportamento do sistema num formato de espaço de estados. O vetor de estados é composto pela *pose* do veículo assim como pelas respetivas componentes de velocidade, isto é, pela velocidade linear (decomposta no plano cartesiano) e pela velocidade angular (Figura 13). O vetor de entrada é descrito pelo binário a aplicar ao modelo de cada roda (Figura 13). O sistema de controlo encontra-se dividido num sistema hierárquico, composto por dois controladores, o controlador global, que implementa o controlo ótimo LQR e o controlador local, que controla a corrente a aplicar a cada motor com base na velocidade angular requerida pelo controlador global.

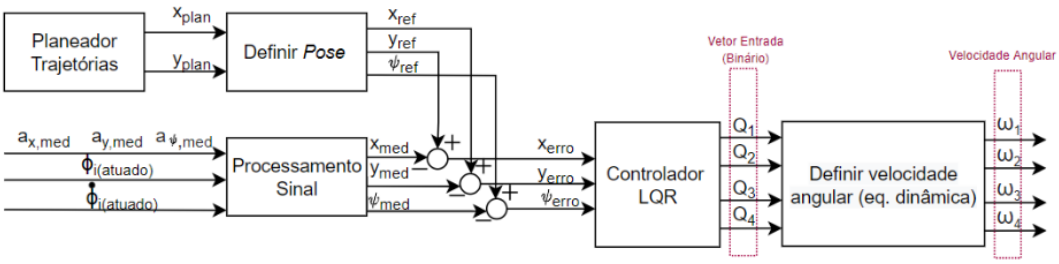


Figura 13: Esquemático da estrutura de controlo. Adaptado de Jie Zhang & Xiaobo Liu-Henke (2020).

A função de custo foi definida pelos fatores Q e R que pesam respetivamente o vetor de estados e o vetor de entrada do sistema de controlo. O ganho ótimo é calculado de forma a minimizar a função de custo definida. No caso específico apresentado em Jie Zhang & Xiaobo Liu-Henke (2020), a função de custo foi definida pelo erro de controlo, o esforço de atuação e as variáveis de estado do sistema. O comportamento do sistema, em estado estacionário, foi melhorado recorrendo ao controlo *feed-forward*.

O algoritmo de controlo exposto em Jie Zhang & Xiaobo Liu-Henke (2020) foi validado em simulação. A trajetória foi definida por um vetor de posições temporal. Os resultados apresentados permitiram concluir que o veículo consegue descrever a rota requerida de forma precisa e com mínimo atraso. No entanto, apesar do comportamento satisfatório, o sistema foi apenas simulado em condições ideais, sem qualquer tipo de perturbações. A dinâmica de controlo não considera a deteção de obstáculos dinâmicos, ou seja, o veículo apenas consegue navegar em ambientes estáticos. Existe a possibilidade de ocorrer colisão com obstáculos que não estejam considerados no planeador de trajetórias.

A técnica de controlo exposta em Han & Zhu (2019) apresenta uma abordagem de controlo ótimo recorrendo ao modelo de controlo preditivo (MPC). Embora ainda seja uma área de investigação ativa, o MPC encontra-se cada vez mais presente nos processos de controlo, devido à capacidade de otimização em tempo real e à facilidade em considerar as perturbações e as incertezas relativas à dinâmica (Han & Zhu, 2019). De modo similar ao controlador LQR, o MPC necessita do modelo matemático do sistema. A problemática de controlo anunciada consiste em capacitar um veículo omnidirecional de seguir uma dada trajetória, previamente definida.

O método exposto em Han & Zhu (2019) determina o modelo de espaço de estados do veículo omnidirecional, determinando o vetor de estados e o vetor de entrada. O vetor de estados é constituído pelo erro lateral e erro da direção de navegação do veículo, determinados pela diferença entre a *pose* do veículo e a projeção do próprio veículo na trajetória (que corresponde à posição desejada). A posição e orientação do veículo é determinada pela cinemática do veículo. Por sua vez, o vetor de entrada é

composto pela taxa de variação da aceleração linear, decomposta pelas componentes longitudinal e lateral e pela aceleração angular. As acelerações são calculadas pelas equações do modelo dinâmico, estabelecidas *à priori*.

O sistema de controlo implementado em Han & Zhu (2019) restringe os limites de movimento do veículo, assim como os de atuação. Relativamente aos limites de movimento, foi definido os valores máximos admitidos do erro lateral e da orientação de navegação, definindo um corredor de movimento válido. A atuação encontra-se limitada pela gama de valores de velocidade válidos (velocidade linear e angular) assim como da taxa de variação das acelerações (lineares e angular). O problema de controlo foi formulado com o intuito de minimizar o erro de trajetória e assegurar a robustez do sistema. Por conseguinte, o problema de controlo foi definido por condições que minimizem o desvio lateral, o erro de orientação e o erro das velocidades lineares e angular. Foi ainda definido um termo de custo final relativamente ao desvio lateral e erro na orientação final.

O desempenho do controlo MPC foi verificado em ambiente de simulação simples, sem obstáculos. O veículo realizou o *tracking* de uma trajetória circular, com cinco metros de raio. Como referência de comparação, foi ainda implementado e testada nas mesmas condições, o comportamento do veículo sob o método de controlo PID clássico. Os resultados apresentados em Han & Zhu (2019) permitiram concluir que o sistema sob o método de controlo MPC obteve um menor erro de *tracking* durante a navegação. No entanto, apesar de apresentar um comportamento satisfatório, o problema de controlo ótimo foi definido de forma complexa, exigindo recursos computacionais de maior desempenho. Por essa razão, verificasse a necessidade de estabelecer um problema de controlo mais otimizado (Han & Zhu, 2019).

2.4. Manobras de Acostagem

O processo de *docking* (acostagem) permite orientar o movimento do veículo de forma que este consiga realizar a aproximação final ao local desejado, minimizando o erro de *pose*. Desta forma, este subcapítulo enuncia alguns dos métodos presentes atualmente na literatura relacionados com a temática de manobras de acostagem autónomas. Embora o foco de trabalho incida no controlo de veículos omnidirecionais, inicialmente são expostas algumas técnicas de controlo de acostagem para veículos de direção diferencial, com o intuito de verificar as estratégias adotadas para a gestão de movimentos.

2.4.1. Manobra de Acostagem em Veículos com Direção Diferencial

As técnicas de controlo de acostagem autónoma permitem que os processos de transporte e manipulação de bens materiais em chão de fábrica dinâmicos sejam executados com a mínima intervenção humana (Louro et al., 2020). Além do mais, permite capacitar os veículos de realizar o *docking* de forma autónoma em locais de carga, abstraindo o problema de gestão de carga dos operadores humanos (Romanov & Tararin, 2021) e (Fan Guangrui & Wang Geng, 2017).

O método proposto em Romanov & Tararin (2021) consiste em capacitar um veículo de realizar manobras de acostagem, de forma autónoma, num local de carga previamente definido. Para tal, numa primeira fase o sistema recorre a uma câmara para identificar a posição e orientação da estação de carga, identificada por um marcador *ArUco* em relação ao sistema de coordenadas da câmara. Os marcadores *ArUco* são identificadores de geometria quadrangular, constituídos por uma borda preta que alberga uma matriz interna que armazena um código binário identificador (Figura 14), similar a um código QR (*quick response*). Uma vez identificado o local de *docking*, o sistema define a trajetória de acostagem e executa a manobra. A trajetória é definida com o auxílio de um ponto intermédio, o *centering point* entre a posição inicial do veículo e o local de acostagem.

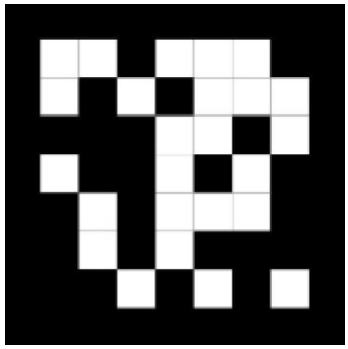


Figura 14: Marcador *ArUco* definido por uma matriz interna (7x7). Retirado de Siki & Takács (2021).

O sistema de controlo exposto em Romanov & Tararin (2021) executa um movimento rotacional com o intuito de procurar o marcador *ArUco* da estação de carregamento. Após ser localizado, é calculado o ângulo entre a orientação do veículo e a orientação do local de *docking*, possibilitando a convergência da orientação do veículo para a desejada. Posteriormente, o veículo desloca-se para o ponto auxiliar de aproximação, o *centering point*, que corresponde ao ponto que interseta o centro do marcador da estação de carregamento e que simultaneamente seja perpendicular ao plano do veículo. Uma vez atingido o *centering point*, o sistema realiza a aproximação final, realizando pequenos ajustes de orientação. A

velocidade linear é ajustada em função da distância ao local de paragem. O sistema verifica a presença de obstáculos recorrendo a sensor *lidar* e a um sensor de colisão. Caso seja detetado algum obstáculo, o veículo permanece imóvel. O desempenho do algoritmo foi verificado, realizando testes em ambiente de simulação simples. Com base nos resultados verificados em Romanov & Tararin (2021) conclui-se que o método de controlo obteve um comportamento satisfatório, tendo em média uma precisão de acostagem de cinco centímetros. Contudo, apesar da técnica de controlo apresentada em Romanov & Tararin (2021) definir com sucesso a manobra de acostagem em ambiente de simulação controlado, a técnica proposta depende da capacidade de cálculo do ponto central, que estabelece a aproximação longitudinal para o local de *docking*.

Em Fan Guangrui & Wang Geng (2017) foi apresentado um método de controlo de *docking* por referência visual. De modo similar ao verificado em Romanov & Tararin (2021) , o local de acostagem encontra-se identificado por um marcador fiduciário, o *AprilTag* (Figura 15).

Figura 15: Marcador *AprilTag* utilizado como recurso à computação da posição 3D e orientação do veículo no espaço. Retirado de Fan Guangrui & Wang Geng (2017).

A técnica de aproximação define-se em duas fases. Na primeira fase, o veículo movimenta-se em direção a um ponto auxiliar de manobra, previamente estabelecido. Como o ponto auxiliar de aproximação dista em cinco metros do *goal point*, o posicionamento do veículo pode ser determinado admitindo um certo erro. Consequentemente, nesta fase o sistema recorre ao *ORB-SLAM* para se localizar no espaço. Após verificada a convergência para o ponto auxiliar, o controlador corrige o erro de *pose* com base na informação do referencial fiducial. Para tal, foi necessário atribuir um sistema de eixos coordenados para o veículo, para a câmara acoplada e para o marcador *AprilTag*, relacionados entre si (Figura 16). Deste modo, recorrendo às relações geométricas, determina-se a distância cartesiana relativa entre a posição do veículo e o local de *docking*, obtendo um ajuste preciso do movimento até que seja atingida a *pose* final. Ao contrário do verificado em Romanov & Tararin (2021), o comportamento

da técnica exposta foi verificado em chão de fábrica, considerando a presença de obstáculos estáticos. Com o intuito de verificar a robustez do método, foram definidas cinco posições iniciais distintas, todas com a mesma distância ao local de acostagem. Para cada posição inicial, foi ainda atribuído três pontos auxiliares de acostagem possíveis, alterando apenas a distância longitudinal entre eles. Com base na análise dos resultados, conclui-se que o método apresentou uma eficácia média de noventa e sete porcento (Fan Guangrui & Wang Geng, 2017). No entanto, o desempenho da técnica apresentada depende da deteção do marcador fiduciário. O sistema pode demorar até dois minutos até que este seja detetado, introduzindo uma incerteza nos sistemas de gestão dos processos industriais.

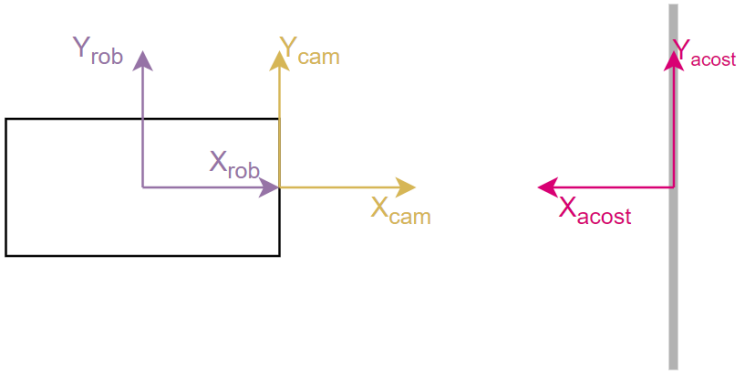


Figura 16: Sistema de eixos coordenados atribuídos. Adaptado de Fan Guangrui & Wang Geng (2017).

Como já referido no subcapítulo Sistemas Dinâmicos não-Lineares, também em Louro et al. (2020) é apresentado um método que permite capacitar um veículo móvel de realizar manobras de acostagem, de modo autónomo. A técnica implementa um mecanismo de gestão de comportamentos que permite conjugar o comportamento de navegar de forma autónoma em ambientes complexos com a capacidade de realizar o processo de *docking*, baseando-se nos sistemas dinâmicos não-lineares. Para o efeito, foi implementado um gestor de tarefas (gere uma máquina de estados) que interpreta a operação a realizar, com base na tarefa solicitada pelo servidor, ou em função da última operação realizada, que corresponde ao estado do sistema. Uma vez identificado qual o estado do sistema, o controlador executa as operações complementares ao respetivo estado. Na aplicação enunciada em Louro et al. (2020), o sistema foi definido por nove estados distintos, contudo alguns apresentam similaridades entre si. De modo geral, os estados que definem a manobra de acostagem estabelecem três pontos de manobra auxiliares, dois referentes a *targets* intermédios e outro que representa o ponto de serviço. Numa primeira fase, o veículo realiza um movimento longitudinal, definido pela dinâmica de "seguir a uma parede", até que o sistema convirja para o primeiro *target* auxiliar. Após ser verificado a convergência, o sistema realiza a

aproximação ao segundo *target* auxiliar, recorrendo à dinâmica de "seguir o *target*". Posteriormente, o veículo ajusta a sua direção de navegação para a orientação esperada do local de acostagem. Na aplicação específica exposta em Louro et al. (2020) foram ainda implementadas algumas operações intermédias que garantem a segurança e a funcionalidade dos sensores incorporados no veículo. Por último, a *pose* do veículo é corrigida pela informação recolhida pela câmara acoplada em função da localização do local de *docking*, até que seja alcançado o *point of service*, o último ponto auxiliar definido. O comportamento do método foi verificado em ambiente industrial dinâmico, especificando operações que envolvem a manobra de acostagem. Os resultados apresentados demonstram que o método de controlo proposto é robusto, mesmo na presença de obstáculos durante a manobra. O veículo cumpriu com as especificações de segurança impostas e estabeleceu com sucesso as manobras de acostagem.

2.4.2. Manobra de Acostagem em Veículos Omnidirecionais

A introdução de veículos omnidirecionais autónomos na indústria introduz uma maior flexibilidade e manobrabilidade nos processos relacionados com o transporte de matérias-primas, em chão de fábrica (Jie Zhang & Xiaobo Liu-Henke, 2020). Dessa forma, e tendo em consideração as manobras de acostagem, os veículos omnidirecionais permitem agilizar e reduzir o espaço e duração das manobras de aproximação às zonas de trabalho, que consequentemente aumenta a eficiência de execução de tarefas nos diversos processos industriais.

A maioria das técnicas atuais recorre a métodos de perceção, baseados em informação sensorial, para realizar o processo de acostagem. Este tipo de método induz um erro de posicionamento final, que dificulta a posterior manipulação dos objetos na zona de trabalho (Bavelos et al., 2021).

Um método de controlo de acostagem de uma plataforma móvel omnidirecional foi proposto em TECNALIA (2018). A técnica recorre a informação visual, fornecida por uma câmara montada na zona inferior do veículo, com o intuito obter a posição e orientação do ponto de acostagem. Na aplicação exposta, a referência de acostagem foi estabelecida recorrendo a um marcador. O controlo do movimento de *docking* foi definido por um controlo proporcional, que garante e mantém a posição do veículo em relação à posição de referência. O controlador, com base na realimentação do erro de posicionamento, determina a velocidade a aplicar ao sistema de tração. O parâmetro de controlo relativo à referência permite estabelecer a relação entra a posição final requerida, no caso a posição do marcador, e a posição final do manipulador móvel, sendo possível ajustar a tolerância de aproximação.

O comportamento do manipulador móvel na fase de acostagem foi verificado numa aplicação prática. No ensaio enunciado em TECNALIA (2018), o veículo movimenta-se, de forma autónoma, até à posição de aproximação definida. Uma vez atingida a posição de aproximação, o controlo de movimento fica definido pelo método de controlo de acostagem. Os resultados apresentados em TECNALIA (2018) permitem concluir que a técnica de controlo é robusta, apresentando um comportamento estável durante a fase de aproximação. No entanto, apesar do veículo ser omnidirecional, o método de controlo apresentado não recorre à maior flexibilidade apresentada por estes veículos, durante o processo de acostagem. A técnica de controlo apenas permite compensar a distância longitudinal e corrigir o movimento lateral do veículo.

Em Bavelos et al. (2021) foi apresentado um outro método de controlo de navegação e acostagem para um manipulador móvel omnidirecional. O método descrito divide-se em duas etapas, a primeira que realiza o processo de navegação autónoma, recorrendo unicamente a informação sensorial *laser*. Numa segunda fase, no processo de acostagem, o sistema compensa o erro de posicionamento, admitido no processo de navegação, recorrendo a informação visual. Ou seja, a segunda etapa conjuga o algoritmo de navegação, que utiliza dados recolhidos pelos sensores *laser*, com a informação visual. A posição final de acostagem é definida por um marcador visual, no caso específico um marcador fiducial, referenciada pelo sistema de visão.

O controlo do movimento de acostagem encontra-se definido pelo controlador Proporcional-Integral-Derivativo (PID), com realimentação do erro de posicionamento. O erro foi estabelecido pela diferença entre a posição final, determinada pelo sistema de visão, e a posição do veículo. Os parâmetros de controlo foram determinados com base na análise do comportamento. Com base no erro, o controlador determina a velocidade desejada do veículo, que serve de *input* ao sistema de tração do manipulador móvel omnidirecional. Nesta técnica, a precisão de acostagem pode ser regulada pelo ajuste dos parâmetros de controlo ou pela restrição do erro de aproximação aceitável.

O comportamento do movimento do manipulador móvel em chão de fábrica foi analisado em Bavelos et al. (2021). Apesar do sistema apresentar um erro de cinco centímetros durante a fase de navegação, um valor aceitável para a aplicação enunciada, este foi reduzido na fase de acostagem, recorrendo ao sistema de visão. Nos testes realizados, o erro de posicionamento máximo obtido foi de apenas um centímetro. Embora o método proposto apresente bom desempenho e exija uma carga computacional reduzida, o algoritmo de controlo apenas permite que o veículo se aproxime da área de trabalho com movimentos laterais ou longitudinais, dependendo do local da instalação da câmara na plataforma omnidirecional, não aproveitando a maior flexibilidade apresentada pelos veículos omnidirecionais.

Em Xiuzhi Li et al. (2018) foi apresentado uma outra possível abordagem de controlo de acostagem, também baseado em controlo por referência visual. Embora o caso de estudo realizado seja redirecionado para um veículo omnidirecional de transporte de utentes que requerem cuidados médicos, o processo pode ser adaptado a uma aplicação industrial. Na aplicação enunciada, o veículo necessita de se aproximar de uma cama auxiliar, de forma autónoma e precisa. Devido à natureza da manobra, desaconselha-se o controlo manual devido à possibilidade de ocorrência de colisões. Consequentemente, o autor propõe um método de controlo visual de *docking*, com realimentação da posição, para veículos omnidirecionais, mitigando a ocorrência de colisões.

O controlo do movimento deve assegurar que o veículo fique paralelo e adjacente à posição desejada de *docking*. A tarefa de acostagem envolve, numa primeira fase, a determinação da *pose* do veículo, de modo a ser possível estabelecer o erro correspondente à diferença entre a *pose* final desejada e a do veículo. Por sua vez, o sistema recorre ao controlo servo que estabelece o comportamento do movimento entre a posição inicial e a posição desejada, com base no *feedback* da posição (Figura 17).

Embora a posição e orientação do local de acostagem seja fixo ao referencial do "mundo", a *pose* do veículo e consequentemente a da câmara acoplada variam, sendo necessário atribuir referenciais a cada constituinte do sistema, relacionados entre si (Figura 17). Deste modo, deduz-se a matriz de transformação geral do robô em relação ao ambiente. Com base na matriz, é possível determinar a posição e orientação do veículo em relação ao referencial do "mundo". O método de controlo servo visual requer como *input* a diferença entre a *pose* do veículo, obtida na matriz de transformação, e a *pose* do local de acostagem (o local desejado), calculando a distância relativa. A regra de controlo aplica ganhos proporcionais aos erros da distância e da orientação calculados, obtendo as variáveis de controlo, correspondentes à velocidade linear e angular do veículo. Por último, com base na cinemática inversa de veículos omnidirecionais, calcula-se a velocidade angular a aplicar a cada roda de modo a obter o vetor de velocidade desejado.

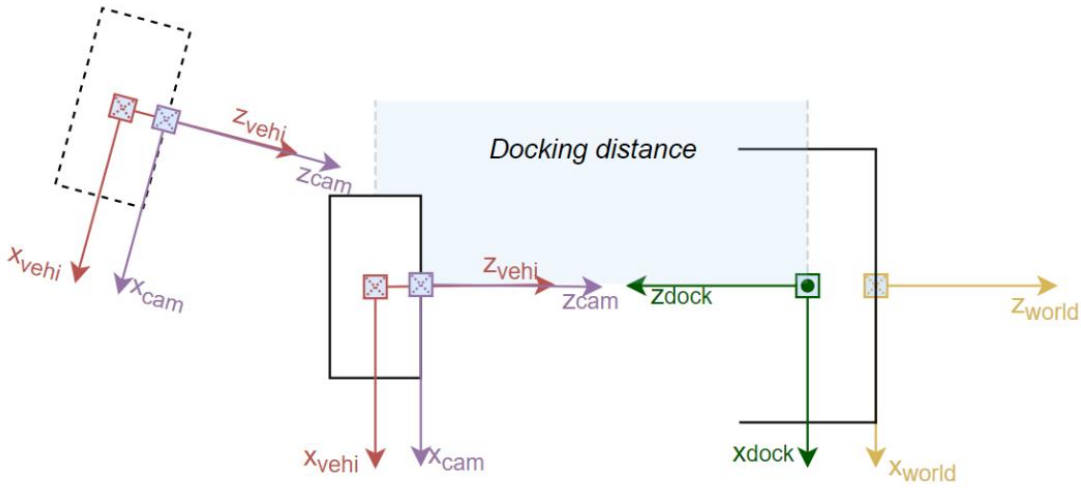


Figura 17: Ajuste de posição e orientação na fase de docking. Adaptado de Xiuzhi Li et al. (2018).

O comportamento do sistema foi verificado em contexto real. Em Xiuzhi Li et al. (2018) foram especificados cinco testes semelhantes entre si, alterando apenas a *pose* inicial, de modo a analisar a precisão da manobra de acostagem, mantendo a *pose* final inalterada. Os testes comprovaram a eficácia e viabilidade do método proposto, obtendo, em média, um erro longitudinal inferior a um centímetro e um erro lateral de cinco centímetros. A orientação obteve um desvio médio inferior a meio grau. Os resultados comprovam que o sistema convergiu gradualmente para a solução de forma estável e precisa.

No método de acostagem proposto em Xiuzhi Li et al. (2018), em contraste com as restantes abordagens de *docking* expostas, implementa tanto o controlo da posição bem como da orientação. Por conseguinte a técnica permite definir a *pose* final de acostagem do veículo, em relação ao referencial do "mundo".

2.5. Discussão da Literatura

Neste capítulo foram expostos alguns dos métodos presentes na literatura referentes à navegação autónoma e manobras de acostagem. Relativamente à navegação autónoma, a estratégia de controlo deve capacitar o sistema de navegar, de forma segura e consciente da presença de operadores humanos, em ambientes complexos dinâmicos. Desse modo, das propostas analisadas, destaca-se o método fundamentado nos sistemas dinâmicos não-lineares. Este planeador local incorpora os comportamentos de seguir um determinado alvo, bem como o de evitar colisões com obstáculos estáticos e dinâmicos em tempo real. Os resultados enunciados demonstram que a estratégia apresenta um comportamento apropriado e tempos de resposta a eventos satisfatórios. Por sua vez, o método revela ser versátil e

adaptável à introdução de novas variáveis de controlo, característica útil para a geração de movimentos holonómicos.

Relativamente às operações de acostagem, para o estudo em causa, é necessário implementar um método que possibilite o controlo da *pose* do veículo durante a execução da manobra. Para o efeito, destaca-se a abordagem proposta em Xiuzhi Li et al. (2018), permitindo o controlo da posição e orientação do veículo, tendo como referência a *pose* de acostagem desejada.

3. FUNDAMENTOS TEÓRICOS

Neste capítulo são abordados os conhecimentos teóricos necessários ao desenvolvimento da solução proposta. Para o efeito, é apresentado a estratégia base da navegação autónoma em espaços dinâmicos, os sistemas dinâmicos não-lineares, abordando os métodos de controlo e conceitos latentes à estratégia. Por sua vez, é exposta a noção de cinemática, a sua relevância para a modelação do movimento de corpos e a sua importância para a robótica, com especial enfoque para a robótica móvel. Por fim, são referidas as componentes necessárias à identificação de um dado objeto no espaço.

3.1. SISTEMAS DINÂMICOS NÃO-LINEARES

O método de controlo relativo à navegação autónoma deve capacitar a plataforma omnidirecional de se movimentar em ambientes fabris dinâmicos complexos, considerando ainda a presença de operadores humanos. Deste modo, o planeador local selecionado deve considerar a ação de navegar em direção a uma determinada posição desejada, definida no referencial do mundo, e o comportamento de evitar colisões com obstáculos estáticos e dinâmicos presentes no ambiente.

Considerando os requisitos enunciados para o caso em estudo, foi selecionada a dinâmica de navegação definida pelos sistemas dinâmicos não-lineares, baseada na estratégia apresentada em Bicho (1999), como planeador local. O planeador tem a vantagem de não requerer o conhecimento prévio do ambiente de navegação, dependendo apenas da informação sensorial recebida relativa ao ambiente circundante. Esta característica permite que seja altamente flexível e ajustável a eventuais alterações do ambiente de navegação, seja este, por exemplo, um ambiente fabril dinâmico.

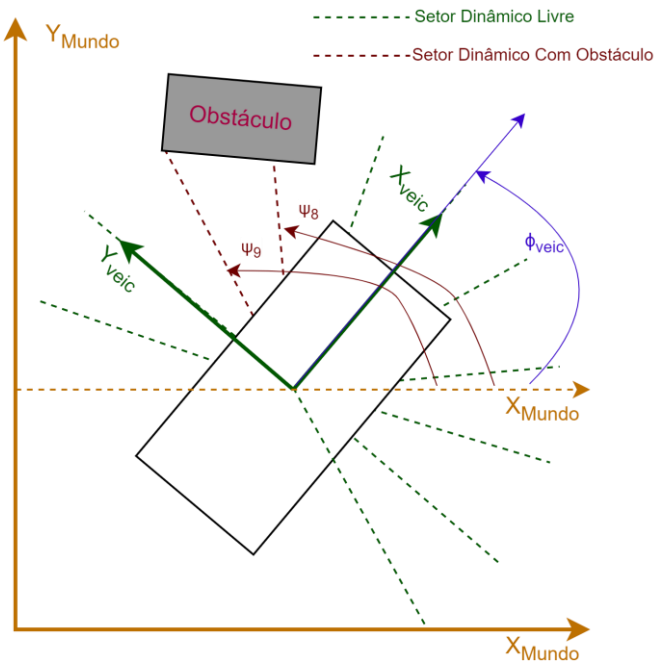


Figura 18: Informação sensorial considerada para o comportamento de navegação autónoma em ambientes dinâmicos.

A dinâmica de controlo gera os movimentos tendo em consideração a influência de dois comportamentos, o de convergência em direção à posição de um alvo e o de evitar colisões com obstáculos detetados. Com base nestes comportamentos, o controlador define a direção de navegação que melhor se adequa às condições verificadas. Durante a fase de navegação autónoma, o movimento do veículo necessita de ser legível e previsível aos operadores humanos, isto é, ser descrito de modo que que estes compreendam intuitivamente as intenções de manobra do veículo. Com efeito, apesar das capacidades holonómicas da plataforma móvel, durante a navegação, o veículo deve apenas se movimentar longitudinalmente, alterando ainda a orientação de navegação. Deste modo, para o caso em estudo, uma das variáveis comportamentais selecionada corresponde à orientação do veículo, em relação ao referencial do mundo, isto é Φ_{veic}, como o verificado na Figura 18. Similarmente ao analisado para a dinâmica de controlo da orientação de navegação, também a velocidade linear, v_x, revela ser uma componente pertinente para a geração de movimentos. Por essa razão, esta é, da mesma forma, considerada como uma variável comportamental a ser definida pelos sistemas dinâmicos não-lineares.

A geração do movimento define-se pela atribuição de valores às variáveis comportamentais ao longo do tempo (Louro et al., 2019). Estes valores são obtidos como sendo soluções do sistema dinâmico, isto é, pontos fixos estáveis do sistema dinâmico, descritos nas equações 1 e 2.

$$\frac{d\Phi_{veic}}{dt} = F(\Phi_{veic}, parametros) = f_{alvo}(\Phi_{veic}) + f_{obs}(\Phi_{veic}) \quad (1)$$

$$\frac{dv_x}{dt} = G(v_x, parametros) = c_{alvo}\, g_{alvo}(v_x) + c_{obs}\, g_{obs}(v_x) \quad (2)$$

Como presente em Bicho (1999), $\frac{d\Phi_{veic}}{dt}$ representa o sistema dinâmico que descreve a taxa de variação da orientação de navegação em função do tempo, $\Phi_{veic}(t)$. O mesmo se verifica para a variação da velocidade longitudinal, $v_x(t)$. A dinâmica de $F(\Phi_{veic})$ define-se pela influência simultânea dos comportamentos de convergência em direção a um determinado alvo, $f_{alvo}(\Phi_{veic})$, e de evitar colisões com obstáculos, $f_{obs}(\Phi_{veic})$. Do mesmo modo, a dinâmica de $G(v_x)$ depende dos comportamentos descritos em $F(\Phi_{veic})$, no entanto, neste caso, somente um dos comportamentos pode influenciar a dinâmica. Dessa forma, e em função do estado do sistema, determina-se o valor das variáveis de ativação, c_{alvo} e c_{obs}, moldando o peso associado a cada comportamento.

As contribuições que definem os sistemas dinâmicos criam pontos atratores num determinado valor desejado, no caso correspondente à direção do ponto alvo, e repulsores nos valores indesejados, como no caso da direção dos obstáculos. Os parâmetros de cada contribuição são ajustados de modo que o sistema fique apenas definido por soluções assimptoticamente estáveis, ficando robusto perante perturbações e ruídos estocásticos.

3.1.1. Dinâmica Controlo da Orientação de Navegação

A dinâmica de controlo da orientação de navegação é definida pela dinâmica $F(\Phi_{veic})$. Esta é constituída pelos comportamentos de navegação em direção a um determinado ponto alvo e de evitar colisões com obstáculos. $f_{alvo}(\Phi_{veic})$ define a ação de convergir a orientação do veículo em direção a um valor desejado, no caso, a orientação do próximo ponto intermédio ou alvo considerado. Por sua vez, $f_{obs}(\Phi_{veic})$ implementa o comportamento de divergir a orientação de navegação daquelas que estejam obstruídas com obstáculos detetados, evitando, assim, que ocorra colisões com os mesmos.

a) Convergir Orientação de Navegação em Direção a um Alvo

A dinâmica que implementa o comportamento de seguir para o alvo desejado define-se por $f_{alvo}(\Phi_{veic})$. No caso em estudo, os alvos são considerados como coordenadas cartesianas, definidas

no referencial do mundo. Considera-se que as posições dos alvos são conhecidas previamente, sendo estas fornecidas pelo planeador global.

Para um dado ponto alvo intermédio, determina-se a sua direção, relativamente à própria plataforma móvel, tendo em consideração a posição do veículo no espaço, $p_{veic} = (x_{veic}, y_{veic})$, e a coordenada do alvo em relação ao referencial do mundo, $p_{alvo} = (x_{alvo}, y_{alvo})$. Deste modo, a direção do alvo em relação ao veículo, é determinada segundo a equação 3.

$$\Psi_{alvo} = \tan^{-1}\left(\frac{y_{alvo} - y_{veic}}{x_{alvo} - x_{veic}}\right) \tag{3}$$

Uma vez que a direção Ψ_{alvo} corresponde a uma orientação de navegação desejada, o sistema dinâmico relativo a $f_{alvo}(\Phi_{veic})$ deve introduzir um ponto atrator em $\Phi_{veic} = \Psi_{alvo}$, com uma taxa de relaxamento definida por λ_{alvo} (> 0). A premissa de convergência em direção ao alvo deve ser garantida independentemente da orientação do sistema. Como tal, a força atratora deve influenciar a dinâmica em todo o domínio válido, isto é, para o vetor circular de 360 °. Entre outras possíveis soluções, os requisitos enunciados são cumpridos pelo modelo matemático descrito pela equação 4, como verificado em Bicho (1999).

$$\frac{d\Phi_{veic}}{dt} = f_{alvo}(\Phi_{veic}) = -\lambda_{alvo}\sin(\Phi_{veic} - \Psi_{alvo}) \tag{4}$$

b) Evitar colisões com obstáculos

O comportamento de evitar colisões com obstáculos, sejam estes estáticos ou dinâmicos, define-se pelo somatório de $f_{obs,i}(\Phi_{veic})$, como modelado na equação 5. Na estratégia enunciada em Bicho (1999), foi considerado a influência de sete sensores infravermelhos individuais, demonstrando desse modo a robustez e fiabilidade do método. No entanto, para o projeto em causa, foi necessário definir um método de conversão de feixes *laser* em setores de interesse à dinâmica responsável por evitar obstáculos, como o enunciado no capítulo 2. O método permite reduzir a informação fornecida pelos dois sensores *lidar*, cerca de 1080 feixes *laser*, em n setores de interesse à dinâmica, impondo $n \ll 1080$.

$$f_{obs_{Total}}(\Phi_{veic}) = \sum_{i=1}^{n} f_{obs,i}(\Phi_{veic}) \quad (5)$$

A estratégia adotada consiste em formular um ponto fixo repulsor na direção do obstáculo, sendo esta coincidente com a orientação do respetivo setor, isto é, em $\Psi_{obs,i} = \Phi_{veic} + \theta_i$, representando um obstáculo virtual para cada setor. Em função da distância e orientação dos obstáculos, dados fornecidos pelos setores dinâmicos, modula-se o comportamento responsável por evitar obstáculos segundo a equação 6.

$$f_{obs,i}(\Phi_{veic}) = \lambda_{obs,i}.\left(\Phi_{veic} - \Psi_{obs,i}\right).e^{-\frac{(\Phi_{veic}-\Psi_{obs,i})^2}{2.(\sigma_{obs,i})^2}} \quad (6)$$

A força de repulsão, representada por $\lambda_{obs,i}$ (> 0), deve diminuir em função do aumento da distância ao obstáculo virtual, d_i. Dessa forma, $\lambda_{obs,i}$ pode ser modelada por uma função do tipo exponencial, ajustável pelos parâmetros β_1 e β_2, como na equação 7. Nessa equação, β_1 representa a força máxima de repulsão, para a condição de $d_i \approx 0$, enquanto β_2 define a taxa de decaimento em função distância ao obstáculo virtual.

$$\lambda_{obs,i} = \beta_1 e^{\frac{-d_i}{\beta_2}} \quad (7)$$

O gráfico ilustrado na Figura 19 demonstra a influência do termo β_2 no cálculo de $\lambda_{obs,i}$ em função da distância aos obstáculos. Considerando um β_1 constante e igual para ambos os casos representados, verifica-se que quanto menor for o valor de β_2, menor será a força de repulsão para a mesma distância ao obstáculo.

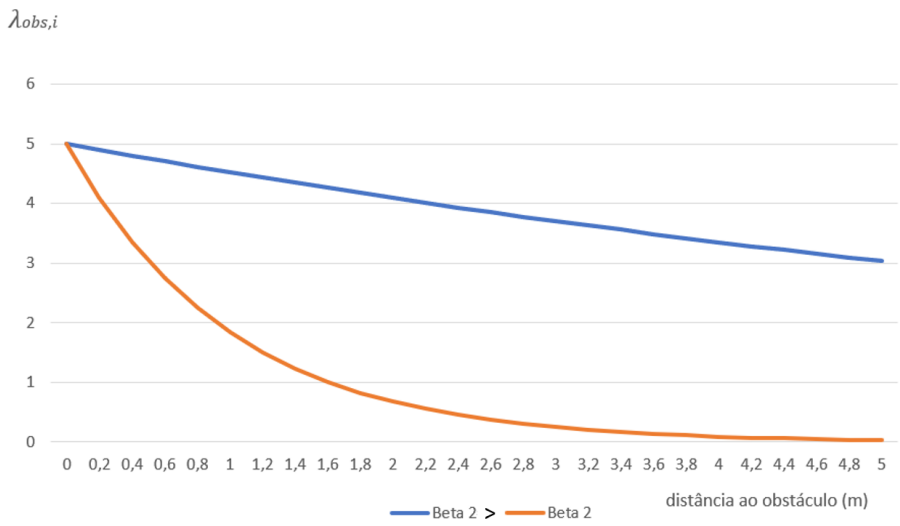


Figura 19: Influência do parâmetro β_2 no cálculo da força repulsora. Quanto maior β_2 maior será a influência da força repulsora na dinâmica de controlo, para a mesma distância ao obstáculo, minimizando a variação do tipo exponencial.

A gama angular sobre a qual a força exerce o seu efeito repulsor define-se pelo parâmetro $\sigma_{obs,i}$, sendo esta dependente da sensibilidade, $\Delta\theta$, assim como da distância ao obstáculo, d_i, conforme descrito na equação 8. Ademais, as dimensões da plataforma móvel devem ser consideradas de modo a definir a margem de segurança entre os obstáculos durante a navegação.

$$\sigma_{obs,i} = \tan^{-1}\left(\tan\left(\frac{\Delta\theta}{2}\right) + \frac{\left(\frac{Largura}{2}\right)}{\left(\frac{Comprimento}{2} + d_i\right)}\right) \quad (8)$$

A Figura 20 ilustra a gama angular sob influência da força repulsora relativa ao setor i. Verifica-se que quanto maior for a distância d_i menor será a margem de segurança, devido ao menor ângulo subentendido aquando da projeção do veículo ao obstáculo detetado, como analisado em Bicho (1999).

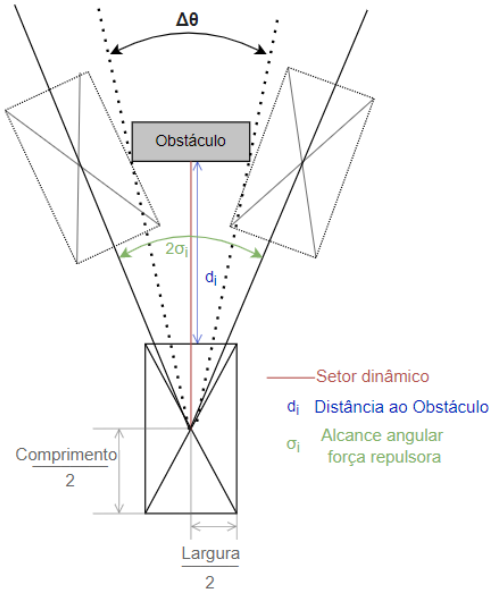


Figura 20: Gama angular sobre a qual a força repulsora exerce o efeito. A gama angular, além de considerar a orientação do obstáculo, deve garantir a passagem do veículo ao lado do obstáculo.

c) Integração dos Comportamentos Considerados (convergir alvo e desviar de obstáculos)

A tarefa de navegação autónoma deve não só garantir que o veículo alcance uma determinada posição desejada no espaço, correspondente a um alvo desejado, como ainda de ter a capacidade de evitar colisões com possíveis obstáculos durante o trajeto. Com efeito, o comportamento deve ser influenciado pela dinâmica de seguir para um alvo, $f_{alvo}(\Phi_{veic})$ e pela dinâmica de evitar colisões com obstáculos detetados pelos setores da dinâmica, $f_{obs_{Total}}(\Phi_{veic})$. Deste modo, a dinâmica define-se pela integração de ambos os comportamentos, como presente na equação 9.

$$\frac{d\Phi_{veic}}{dt} = f_{alvo}(\Phi_{veic}) + f_{obs_{Total}}(\Phi_{veic}) \quad (9)$$

A dinâmica de controlo de direção pode ser definida simultaneamente por ambos comportamentos, contudo, deve ser privilegiado o de fuga aos obstáculos, associando, deste modo, um maior peso à dinâmica $f_{obs,i}(\Phi_{veic})$. Isto implica que, matematicamente, a força de repulsão deve ser superior à taxa de relaxamento relativa ao comportamento de convergência ao alvo, isto é, $\lambda_{alvo} << \lambda_{obs}$.

A dinâmica resultante do sistema dinâmico pode ser definida por múltiplos pontos fixos, sendo variável em função do espaço circundante à plataforma móvel sensorizado. A Figura 21 exemplifica uma possível

situação na qual a rota de navegação se encontra obstruída por dois obstáculos espaçados entre si a uma distância inferior à largura da plataforma móvel.

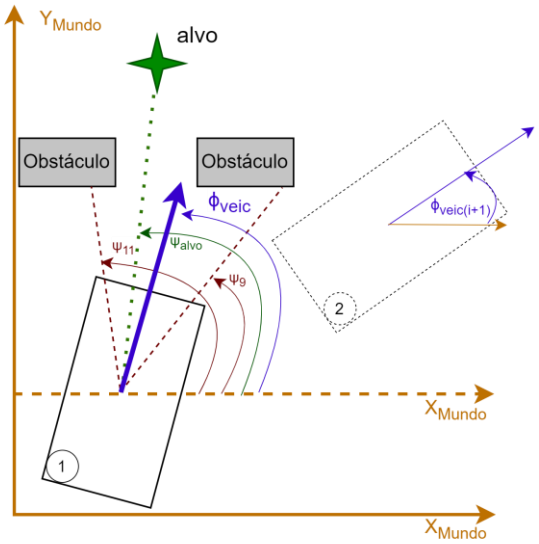


Figura 21: Distância entre obstáculos insuficiente à passagem do veículo entre obstáculos. Como tal, devido à influência da força repulsora, a dinâmica de navegação introduz na orientação do alvo um ponto repulsor. Ademais, são erigidos dois novos pontos fixos estáveis na orientação de fuga à trajetória de colisão.

Na condição apresentada na Figura 21, o sistema dinâmico que define o comportamento de seguir para o alvo, $f_{alvo}(\Phi_{veic})$, erige um ponto fixo estável na orientação Ψ_{alvo}. No entanto, a dinâmica de evitar obstáculos introduz, na mesma orientação, um ponto fixo repulsor, opondo-se ao comportamento de orientação ao alvo. A Figura 22 demonstra a dinâmica individual de cada comportamento, assim como a dinâmica resultante, referente à equação 9, considerando que a força de repulsão seja superior à de atração. Como é possível verificar, a dinâmica resultante introduz um ponto repulsor na orientação do alvo, evitando, deste modo, a colisão com os obstáculos detetados.

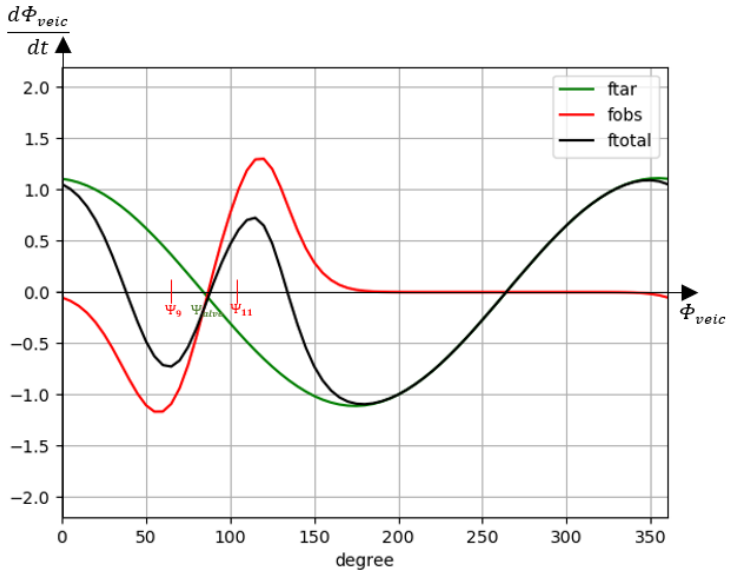


Figura 22: Representação gráfica dos campos vetoriais que definem a dinâmica de navegação na condição de a distância entre obstáculos ser insuficiente à passagem do veículo. Como verificável, a dinâmica resultante introduz na orientação do alvo um ponto fixo repulsor, anulando o efeito do comportamento de seguir para o alvo.

Durante a navegação num ambiente dinâmico pode ocorrer mudanças qualitativas no comportamento da plataforma omnidirecional, devido às alterações do número de pontos fixos assim como à estabilidade associada a estes. As alterações correspondem a bifurcações do sistema dinâmico, resultantes da informação sensorial recolhida. Dessa forma, é espetável que num ambiente complexo dinâmico possam ocorrer alterações de comportamento da plataforma móvel durante a execução da rota, no caso de serem detetados obstáculos. Como analisado em Bicho (1999), caso o ponto de bifurcação seja coincidente com a orientação de navegação do veículo, o sistema tende a estabilizar nesse ponto, podendo o mesmo ser do tipo instável. Uma possível solução consiste em introduzir na dinâmica de controlo de orientação um ruído branco gaussiano de variância unitária, ξ_n, com uma força estocástica de variância efetiva de Q, equação 10. Uma vez que o comportamento do sistema é gerado por soluções assimptoticamente estáveis, o sistema é robusto ao ruído introduzido bem como a eventuais ruídos de medição, de atuação e de ambiente.

$$f_{estoc} = \sqrt{Q}\xi_n \tag{10}$$

Desse modo, o sistema dinâmico que define o controlo da direção de navegação é descrito segundo 11, tendo por base as equações 9 e 10.

$$\frac{d\Phi_{veic}}{dt} = f_{alvo}(\Phi_{veic}) + f_{obs_{Total}}(\Phi_{veic}) + f_{estoc} \quad (11)$$

3.1.2. Dinâmica de Controlo da Velocidade Linear

A presença de obstáculos dinâmicos no ambiente, assim como a variação da informação sensorial, em resultado do deslocamento da plataforma móvel no espaço, redefinem os pontos fixos no sistema dinâmico definido na equação 11. Como enunciado em Bicho (1999), o sistema deve ser definido unicamente por soluções assintoticamente estáveis, isto é, deve estar próximo de um ponto fixo atrator. Para tal, o sistema deve ter a capacidade de convergir a orientação de navegação para os pontos fixos estáveis, à medida que este se desloque.

A taxa de deslocamento dos pontos fixos depende da velocidade dos obstáculos detetados, caso estes sejam dinâmicos, bem como da velocidade linear da plataforma móvel no ambiente. Efetivamente, deve ser implementado um método de controlo da velocidade que permita limitar a taxa de variação do deslocamento dos pontos fixos. Tendo por base os sistemas dinâmicos, a velocidade linear pode ser calculada segundo a equação 12.

$$\frac{dv_x}{dt} = -c_{alvo}.(v_x - V_{alvo}) - c_{obs}.(v_x - V_{obs}) \quad (12)$$

Tal como no sistema dinâmico que controla a direção de navegação, o controlo da velocidade linear, equação 12, é o resultado de dois comportamentos: o comportamento de se dirigir para o alvo e o comportamento de se desviar de obstáculos. Apenas um comportamento pode estar ativo ao mesmo tempo, estando no comportamento de seguir para o alvo se não existirem obstáculos na confluência da direção de navegação, ou então passa para o comportamento de desvio de obstáculos. Ambos comportamentos são controlados por um sistema dinâmico linear, que erige um atrator na velocidade desejada para cada um dos comportamentos, V_{alvo} ou V_{obs}. No comportamento de dirigir para o alvo, o valor de V_{alvo} deve convergir para o valor da velocidade de cruzeiro, ou caso esteja perto de uma posição de paragem (alvo final) reduzir a velocidade desejada em função da distância ao mesmo ($dist_{alvo}$). No comportamento de desvio de obstáculos, V_{obs}varia em função da distância ao obstáculo mais próximo, $d_{i,min}$, conforme equação 13.

$$V_{obs} = d_{i,min}.Max_{obs,dist} \quad (13)$$

O cálculo de V_{obs} depende ainda do parâmetro $Max_{obs,dist}$, que permite delinear a velocidade máxima de deslocamento para o comportamento de desvio de obstáculos. O parâmetro deve ser ajustado de modo que o sistema consiga "seguir de perto" o ponto fixo atrator. Ademais, deve-se considerar a possibilidade de V_{obs} ser superior a V_{alvo}. Esta condição pode ocorrer no caso de a velocidade de cruzeiro na rota ser baixa. Nesse caso, limita-se o valor máximo de V_{obs} como inferior a V_{alvo}.

Embora o sistema dinâmico expresso em 12 permita definir dois pontos fixos estáveis, em V_{alvo} e em V_{obs}, tal não é desejável. Assim, recorre-se aos parâmetros c_{alvo} e c_{obs} como mecanismos de seleção da componente ativa na dinâmica, para um determinado instante de tempo, em função da orientação do obstáculo detetado. Tendo em consideração a informação recolhida dos setores dinâmicos, é possível integrar uma função potencial do sistema dinâmico que permita aferir se a direção de navegação se encontra na zona repulsiva, isto é, se a orientação de navegação coincide com a orientação de um obstáculo.

$$U(\Phi_{veic}) = \sum_{i=1}^{N} \left[\lambda_{obs,i} ({\sigma_{obs,i}}^2) e^{-\frac{(\Phi_{veic} - \Psi_{obs,i})^2}{2.(\sigma_{obs,i})^2}} - \frac{\lambda_{obs,i}.{\sigma_{obs,i}}^2}{\sqrt{e}} \right] \quad (14)$$

Valores positivos de $U(\Phi_{veic})$, equação 14, indicam que a direção de navegação se encontra dentro de uma zona repulsiva, pelo que apenas o termo relativo aos obstáculos na equação 12 deve estar ativo. Como tal, $c_{alvo} = 0$ e $c_{obs} = c_{v,obs}$. Do mesmo modo, valores negativos assinalam que a rota se encontra desobstruída, pelo que $c_{obs} = 0$ e $c_{alvo} = c_{v,alvo}$. É, no entanto, desejável implementar um mecanismo que converta valores contínuos, resultados de $U(\Phi_{veic})$, em valores restritos entre $-\frac{1}{2}$ e $\frac{1}{2}$, facilitando posteriormente o cálculo do valor da intensidade da força de atração.

$$\alpha(\Phi_{veic}) = \tan^{-1}\left(\frac{c.U(\Phi_{veic})}{\pi}\right) \quad (15)$$

Deste modo, os parâmetros c_{alvo} e c_{obs}, relativos à equação 12, podem ser calculados segundo as equações 16 e 17. Os termos $c_{v,obs}$ e $c_{v,alvo}$ especificam as forças de atração na dinâmica de controlo da velocidade linear, no caso de estar numa zona de repulsão ou fora desta, respetivamente.

$$c_{obs} = c_{v,obs}(\frac{1}{2} + \alpha(\Phi_{veic})) \tag{16}$$

$$c_{alvo} = c_{v,alvo}(\frac{1}{2} - \alpha(\Phi_{veic})) \tag{17}$$

Como indicado em Bicho (1999), de modo que o comportamento de fuga aos obstáculos tenha precedência sobre o de "seguir o alvo", assim como se assegure a estabilidade do sistema, a seguinte hierarquia deve ser considerada aquando da parametrização dos termos dos sistemas dinâmicos (condições de parametrização 18):

$$\lambda_{alvo} \ll \lambda_{obs}; \qquad \lambda_{alvo} \ll c_{v,alvo}; \qquad \lambda_{obs} \ll c_{v,obs}; \tag{18}$$

3.2. Cinemática de Veículos

O termo movimento está naturalmente presente no nosso quotidiano, sendo um conceito fundamental que molda o conhecimento empírico dos conceitos associados aos movimentos naturais dos objetos ou corpos. A título de exemplo, o ser humano, através da observação, consegue intuitivamente prever a origem de um determinado corpo, deduzir a sua trajetória de movimento e antecipar a duração do movimento. Dada a importância da temática, as questões relacionadas com a modelação do movimento para os diversos tipos de objetos ou corpos foi uma das primeiras questões fundamentais da física (Goodman & Zavoro, 2009).

A cinemática, como o próprio nome adverte, relaciona-se com o estudo do movimento, sendo essencial à compreensão do funcionamento dos diversos sistemas mecânicos. A cinemática pode ser definida como um estudo matemático do movimento, desconsiderando a influência das forças e binários associados à locomoção (Spiers et al., 2016). Aplicando o conceito na robótica, a cinemática permite o estudo analítico dos movimentos de um determinado robô, estabelecendo modelos que permitem a mediação entre o sistema robótico modelado e o algoritmo de controlo. Considerando os requisitos de projeto, a cinemática revela ser essencial ao controlo do movimento do veículo.

Ao nível da robótica, a modelização do movimento pode ser definida em função das variáveis das juntas ou em função da atuação necessária para convergir à *pose* desejada, correspondendo à cinemática direta e inversa, respetivamente. A cinemática depende do número de eixos articulados do sistema robótico, correspondendo diretamente ao número de juntas ou graus de liberdade. A Figura 23 ilustra a relação entre a cinemática direta e inversa, evidenciando o facto de o conjunto do espaço das juntas ser dependente da manobrabilidade do sistema robótico (graus de liberdade), enquanto o conjunto no espaço ser limitado às variáveis que definem o movimento no espaço, tendo dimensão máxima de 6 (*pose*).

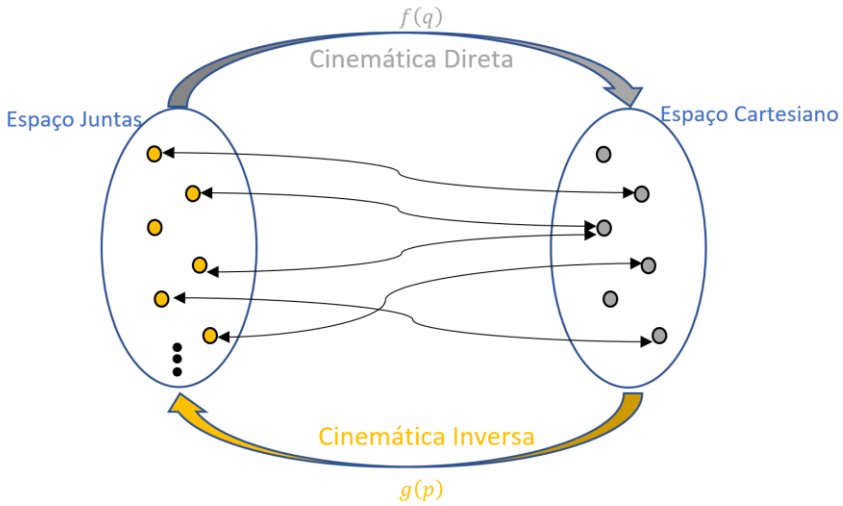


Figura 23: Cinemática direta e cinemática inversa, considerando um sistema robótico.

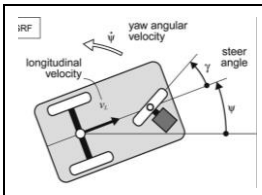
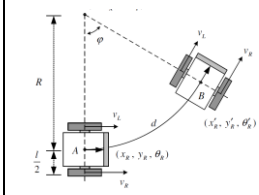
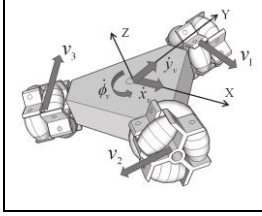
Tendo como referência a Figura 23, a expressão de cálculo da cinemática direta e inversa podem ser definidas pelas equações 19 e 20, respetivamente. Por notação, o conjunto no espaço das juntas define-se por $q = [\theta_1\ \theta_2\ \dots\ \theta_n]^T$, sendo n o número de graus de liberdade enquanto o espaço cartesiano caracteriza-se pela *pose*, p.

$$p = f(q) = \begin{bmatrix} x \\ y \\ z \\ \theta \\ \Psi \\ \Phi \end{bmatrix} \tag{19}$$

$$q = g(p) = f^{-1}\left(\begin{bmatrix} x \\ y \\ z \\ \theta \\ \Psi \\ \Phi \end{bmatrix}\right) \tag{20}$$

A Tabela 2 apresenta os tipos de robôs móveis mais comuns, referindo as variáveis que caracterizam o espaço das juntas e o espaço cartesiano do respetivo sistema robótico.

Tabela 2: Variáveis que caracterizam o espaço das juntas e o espaço cartesiano para diferentes tipos de robôs móveis.

	Tipo	Variáveis Espaço Juntas		Variáveis Espaço Cartesiano	
	Triciclo	V_L	Velocidade Longitudinal	(x, y)	Posição no Espaço
		Υ	Ângulo Direção	Φ	Orientação no Espaço
	Diferencial	V_L	Velocidade Roda Esquerda	(x, y)	Posição no Espaço
		V_R	Velocidade Roda Direita	Φ	Orientação no Espaço
	Omnidirecional (3 Rodas)	V_1	Velocidade Roda 1	(x, y)	Posição no Espaço
		V_2	Velocidade Roda 2	Φ	Orientação no Espaço
		V_3	Velocidade Roda 3		

Considerando o projeto de dissertação, o modelo do veículo selecionado assemelha-se ao omnidirecional, mas de 4 rodas mecânicas, tendo mais uma variável no espaço das juntas se comparado com o de 3 rodas apresentado na Tabela 2. O estudo da cinemática do veículo teve ainda em consideração os métodos de controlo previstos, com principal destaque para as variáveis de saída espectável do módulo de controlo do movimento. Desse modo, o conjunto de variáveis no espaço das juntas considerado define-se por $q = [\omega_1\ \omega_2\ \omega_3\ \omega_4]^T$, enquanto o do espaço cartesiano por $p = [V_{X_{Robo}}\ V_{Y_{Robo}}\ W_{Robo}]^T$. As variáveis relativas ao espaço cartesiano são equivalentes às apresentadas

na Tabela 2, estabelecendo uma dependência temporal entre si, isto é, p corresponde à variação temporal das variáveis que determinam a *pose* do veículo no espaço, $\frac{dx}{dt} = V_{X_{Robo}}$, $\frac{dy}{dt} = V_{Y_{Robo}}$ e $\frac{d\Phi}{dt} = W_{Robo}$. A cinemática considerada fundamenta-se de acordo com as equações 21 e 22.

$$p = \begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix} = f\left(\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}\right) \tag{21}$$

$$q = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = f^{-1}\left(\begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix}\right) \tag{22}$$

Visto que a estratégia de controlo do movimento adotada se define no espaço cartesiano, dá-se especial relevância ao estudo cinemática inversa do veículo, como apresentado no capítulo Cinemática do Veículo. Contudo, a cinemática direta do veículo também é estudada, podendo ser utilizada como recurso de validação do comportamento do veículo ou de localização no espaço, ficando dependente da informação sensorial complementar. Dado que o objetivo de um dos temas de trabalho, associado ao projeto de investigação, consiste em conceber um sistema de localização do veículo em espaço interior, esta problemática não será aprofundada nesta dissertação.

3.3. IDENTIFICAÇÃO DA *POSE* DE UM VEÍCULO NO ESPAÇO

A identificação de um objeto no espaço está dependente das suas características, especificamente aquelas associadas aos graus de liberdade. Dando o exemplo de um veículo automóvel, a sua localização no espaço pode corresponder às coordenadas GPS (Sistema Global de Posicionamento), identificando as respetivas coordenadas geográficas. No entanto, esta informação pode ser complementada adicionando, por exemplo, a altitude em relação ao nível do mar ou ainda a orientação no espaço em relação ao norte magnético.

Considerando um plano tridimensional, a localização de um objeto pode ser assumida por duas componentes, a posição cartesiana no plano, p_e, e a orientação no espaço, Υ_e. Deste modo, e atendendo à Figura 24, a identificação de um objeto considerando os ângulos de *Euler* é definível pelo vetor *pose*, sendo este constituído pela componente de posicionamento $p_e = (x_1, y_1, z_1)$ e pela componente relativa à orientação, $\Upsilon_e = \left(\Psi_{roll}, \theta_{pitch}, \Phi_{yaw}\right)$.

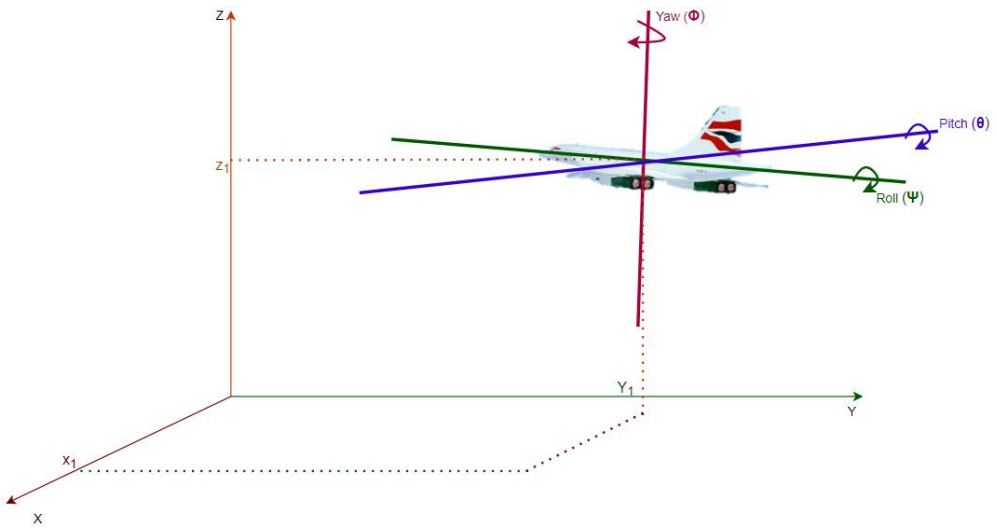


Figura 24: Componentes de localização de um objeto num plano tridimensional. Pose definida pelas componentes de posicionamento e de orientação.

No caso em estudo, a localização da plataforma móvel num espaço tridimensional pode ser descrita pelas componentes identificadas na Figura 25. O veículo apresenta três graus de liberdade, associados não só ao movimento longitudinal e lateral, estes no plano cartesiano bidirecional, como também ao movimento angular, segundo o eixo vertical. Desse modo, a localização pode ser descrita pelo vetor $p_e = (x_1, y_1, z_1)$ e pela orientação em relação ao referencial do mundo Φ_{yaw}, esta última relacionada pela velocidade angular sobre o seu centro de massa.

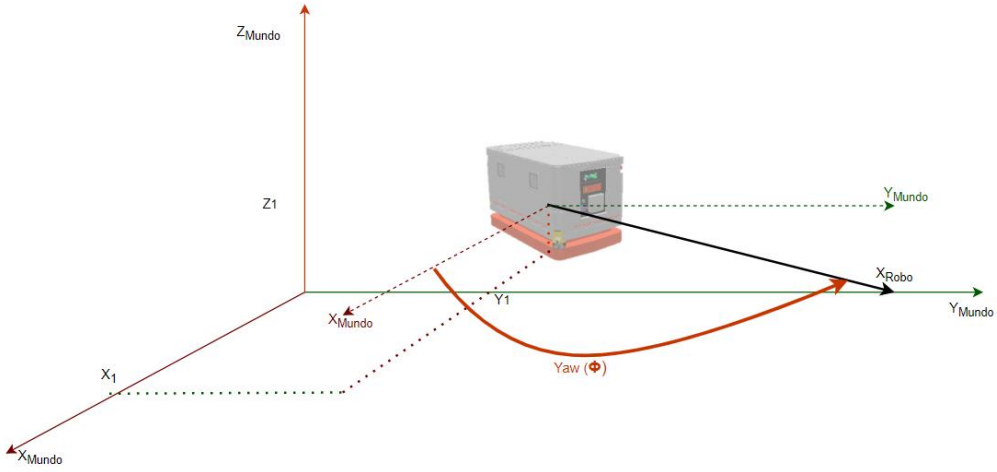


Figura 25: Componentes de localização de uma plataforma móvel num plano tridimensional.

As condições de operação para o caso de estudo desconsideram as variações de altitude durante a navegação do veículo em chão de fábrica. Assim, z_1 assumiu-se como sendo um valor constante e igual ao valor da altura do centro de massa em relação ao solo. Por conseguinte, a representação da plataforma móvel no espaço pode ser simplificada num plano cartesiano bidimensional, como constatado na Figura 26. No caso, a localização do veículo no espaço consistiu na posição cartesiana $p_e = (x_1, y_1)$ e na orientação do veículo, em relação ao referencial do mundo, Φ_{yaw}.

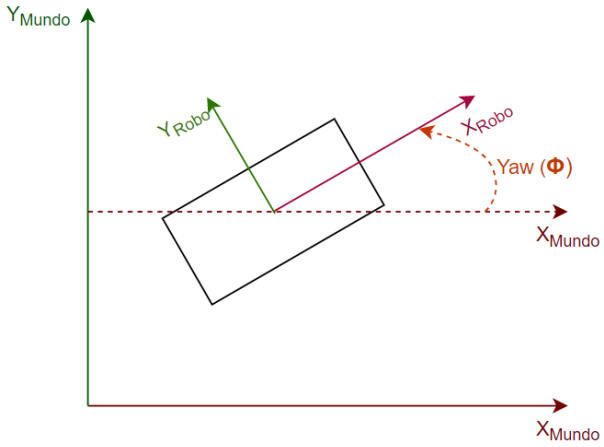


Figura 26: Localização do manipulador móvel considerando um plano bidimensional.

4. MANIPULADOR MÓVEL AUTÓNOMO

Este capítulo visa apresentar a composição do manipulador móvel, destacando a estrutura que permite a locomoção pelo espaço, isto é, a plataforma móvel. Dessa forma, são descritas as principais características da plataforma, tendo especial enfoque para a constituição das rodas, responsáveis pela definição dos movimentos holonómicos, e cinemática associada à locomoção, pelo meio de sensorização do ambiente circundante e formulação do *software* para o estudo e controlo da solução. Relativamente ao *software* é apresentado o *middleware* responsável pela mediação das ações entre o módulo de controlo e a interface, e a ferramenta de simulação utilizada como recurso ao desenvolvimento e validação do comportamento da solução proposta.

4.1. DESCRIÇÃO DO VEÍCULO

O KUKA *Mobile Robot Intelligent Industrial Work Assistant* (KMR iiwa) foi o manipulador móvel selecionado para realizar as tarefas de transporte e manipulação de objetos, em chão de fábrica dinâmico. O KMR iiwa, produzido pela *KUKA AG.*, é composto por um braço robótico, denominado de *leichtbauroboter* (LBR) iiwa 14 R820 e por uma plataforma móvel omnidirecional, o KUKA *Mobile Platform* (KMP) 200 *OmniMove* (Figura 27). A configuração apresentada pela *KUKA AG.* possibilita uma maior manobrabilidade, sendo que o KMP 200 *Omnimove* funcionou como meio de transporte não só do LBR iiwa 14 R820, como também dos objetos a manipular. Por outro lado, o mesmo braço robótico operou em diferentes zonas de trabalho, permitindo dessa forma uma otimização de recursos. No entanto, por questões de segurança, o manipulador apenas pode se movimentar caso a plataforma omnidirecional se encontre em estado considerado como estacionário. Deste modo, durante a tarefa de transporte, o braço robótico permanece imóvel, adotando uma *pose* adequada (Heggem & Wahl, 2020).

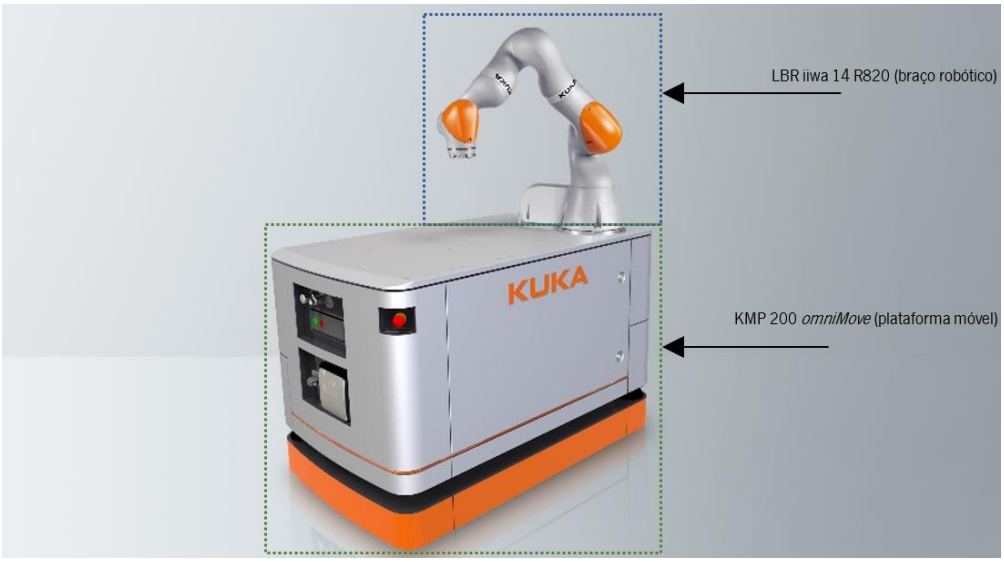


Figura 27: Composição do manipulador móvel Kuka KMR iiwa Adaptado de KUKA AG (2021).

4.1.1. PLATAFORMA OMNIDIRECIONAL

O controlador do movimento da plataforma móvel deve assegurar que as restrições, físicas ou dinâmicas, sejam respeitadas. Desse modo, a Tabela 3 expõe algumas das características principais a ter em consideração para o *design* da aplicação de transporte e manipulação de objetos.

Tabela 3: Principais características da plataforma móvel KMP 200 *omniMove*. Dados recolhidos de KUKA AG (2021).

Comprimento	1,13 m
Altura	0,70 m
Largura	0,63 m
Carga máxima	200 kg
Carga máxima (considerando braço robótico)	170 kg
Velocidade longitudinal *standard*	1,00 m/s
Velocidade lateral *standard*	0,56 m/s
Velocidade rotação *standard*	0,56 rad/s
Aceleração máxima	0,50 m/s^2
Retardo de frenagem máximo	1,00 m/s^2
Tempo operação *standard* (carga completa)	8 h

O tempo de operação indicado na Tabela 3 é um valor *standard*, estando dependente do ambiente de operação e das condições de operação. Caso a plataforma móvel opere segundo as condições *standard*, é possibilitada uma operação continua de oito horas, isto é, um turno de trabalho. De facto, o algoritmo responsável pela gestão de tarefas deve ter em consideração o tempo de operação contínuo da plataforma móvel, de forma a maximizar a eficiência de trabalho nas diversas estações presentes no plano industrial.

A plataforma móvel KMP encontra-se equipada com dois sensores *laser SICK* S300 *Expert*, posicionados diagonalmente e em oposição entre si. Ademais, a locomoção realiza-se pelo acionamento de quatro rodas omnidirecionais do tipo mecânico, independentes entre si. Como abordado no subcapítulo 4.2.1, cada roda é constituída por um conjunto de rolamentos livres, dispostos transversalmente entre si sobre o aro da roda, replicando uma roda de vinte e cinco centímetros de diâmetro. A disposição das rodas mecânicas da plataforma móvel encontra-se representada na Figura 28.

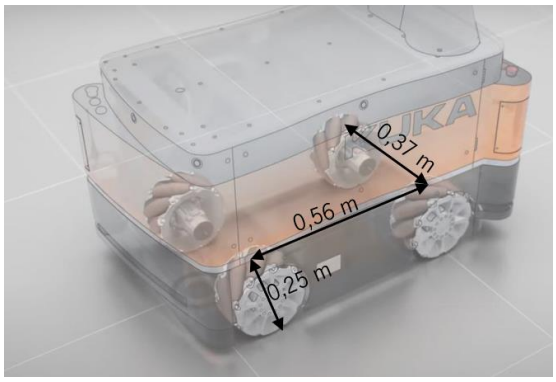


Figura 28: Ilustração da disposição do sistema de locomoção da plataforma móvel KMP 200 *OmniMove*. Adaptado de KUKA AG. (2021).

Na Figura 29 e Figura 30 estão reproduzidas as dimensões do KMP 200 *OmniMove*. As dimensões devem ser consideradas na dinâmica de controlo da navegação assim como na etapa de acostagem de forma a evitar colisões com obstáculos e estabelecer uma distância de segurança durante a navegação entre estações de trabalho. As dimensões indicadas não têm em consideração o braço manipulador.

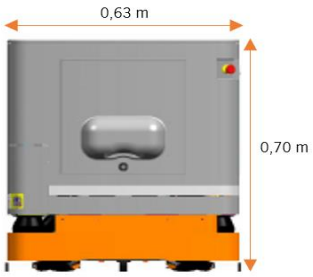


Figura 29: Representação frontal da plataforma móvel
Adaptado de KUKA AG (2021).

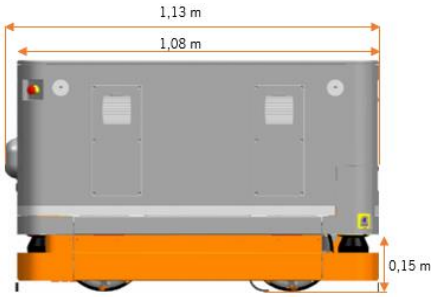


Figura 30: Representação lateral da plataforma móvel
Adaptado de KUKA AG (2021).

4.2. Movimento Omnidirecional

O conceito de movimento omnidirecional pode ser descrito como a capacidade de um veículo em se movimentar no espaço sobre um plano bidimensional, aferindo um maior grau de manobrabilidade e flexibilidade durante o movimento (Yifan Jia et al., 2013). Estes veículos, devido às suas características, possuem três graus de liberdade, permitindo que o movimento seja estabelecido por uma translação no espaço cartesiano, definido por um vetor cartesiano, bem como por um movimento rotacional. Em certas configurações, este pode ser estabelecido por ambos os casos, permitindo, deste modo, operar em ambientes mais exíguos (Ioan Doroftei et al., 2007).

Segundo Ioan Doroftei et al. (2007), os veículos omnidirecionais podem ser divididos em duas categorias: os veículos de "rodas de *design* convencional" e de "rodas de *design* especial", diferenciados pelo tipo de *design* das rodas. Os primeiros recorrem a rodas livres ou a rodas com direção assistida. Contudo, devido à sua natureza não-holonómica, a constante de tempo do processo de ajuste da orientação necessita de ser inferior à dinâmica de navegação. Por sua vez, os veículos de "rodas de *design* especial" baseiam-se no conceito de movimento ativo, relativo ao movimento de rotação da roda no solo, e ainda ao movimento passivo, numa outra direção, dependendo do tipo de *design* (usualmente especificado em 45° ou paralelamente), aumentando a sua manobrabilidade. Neste tipo de *design* destacam-se as rodas universais e as rodas mecânicas ou *Swedish*, ilustradas na Figura 31 e Figura 32, respetivamente.

Figura 31: Roda do tipo Universal. Retirado de Jahanian & Karimi (2006).

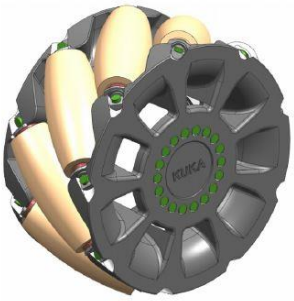


Figura 32: Roda Mecânica ou *Swedish*. Retirado de KUKA AG (2021).

4.2.1. RODAS MECÂNICAS (OU *SWEDISH*)

A roda do tipo mecânica destaca-se como um dos *designs* mais populares na solução de desenvolvimento de plataformas robóticas móveis omnidirecionais. Esta foi desenvolvida no ano de 1973 por Bengt Ilon, (Taheri et al., 2015).

A roda mecânica, considerada como convencional, é constituída por um conjunto de rolamentos, de eixo livre, dispostos transversalmente entre si, orientados num ângulo de 45°, sobre um aro, replicando um perfil de uma roda. Desse modo, o vetor de força pode ser decomposto pela força normal à direção de rotação da roda e pela componente da força no sentido de rotação. Por conseguinte, em função do sentido de rotação e velocidade angular de cada roda omnidirecional, é possível obter um vetor de força resultante na direção de movimento desejada. Devido às características descritas, as rodas *Swedish* são caracterizadas por três graus de liberdade, relativos à rotação da roda no eixo de rotação, w_{rot}, à rotação passiva dos rolamentos, w_{rol}, e ao deslizamento que ocorre sobre o eixo vertical, aquando do contacto dos rolamentos com o solo, w_{des} (Figura 33).

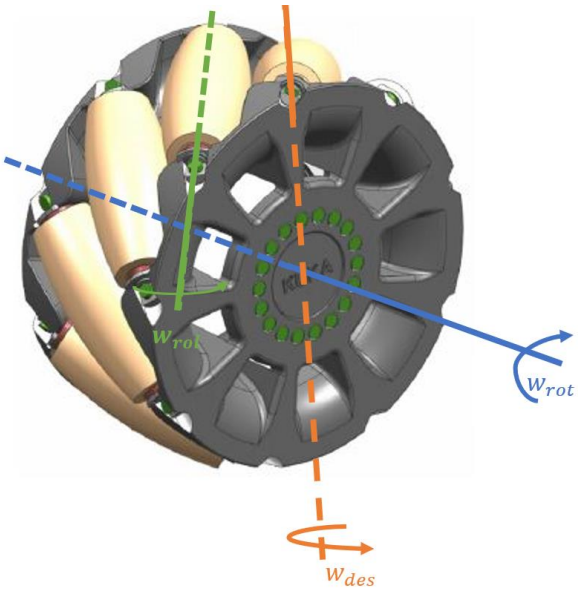


Figura 33: Disposição eixos de manobrabilidade associada à roda do tipo *Swedish*. Adaptado de KUKA AG (2021).

4.3. SENSORES

A plataforma omnidirecional encontra-se equipada com dois sensores *laser SICK* S300 *Expert* (Figura 34). O sensor *lidar* é um sistema de medição do tipo *laser* que permite recolher informação do ambiente circundante ao veículo, em duas dimensões.

Figura 34: Sensor laser *Sick* S300 *Expert*.

Estes sensores permitem medir distâncias de até trinta metros, segundo o princípio de *time-of-flight* (ToF), (Heggem & Wahl, 2020). Na Figura 35 encontra-se representado a disposição dos sensores *SICK* na plataforma KMP200 *Omnimove*.

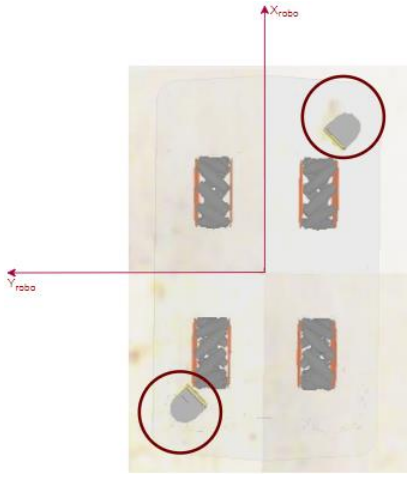


Figura 35: Representação da disposição dos sensores laser no manipulador móvel.

Cada sensor *SICK* S300 apresenta um ângulo de varredura de 270 ° com uma resolução de 0,5 ° e um alcance máximo de 30 m, possibilitando ainda a regulação do campo de proteção num alcance de até 3 m (AG SICK, 2021). Desse modo, e assumindo que o sensor é ideal, conclui-se que o sensor emite um total de 540 feixes *laser*. Na Figura 36 encontra-se representado os setores do sensor *laser SICK* S300, em função do ângulo de varredura.

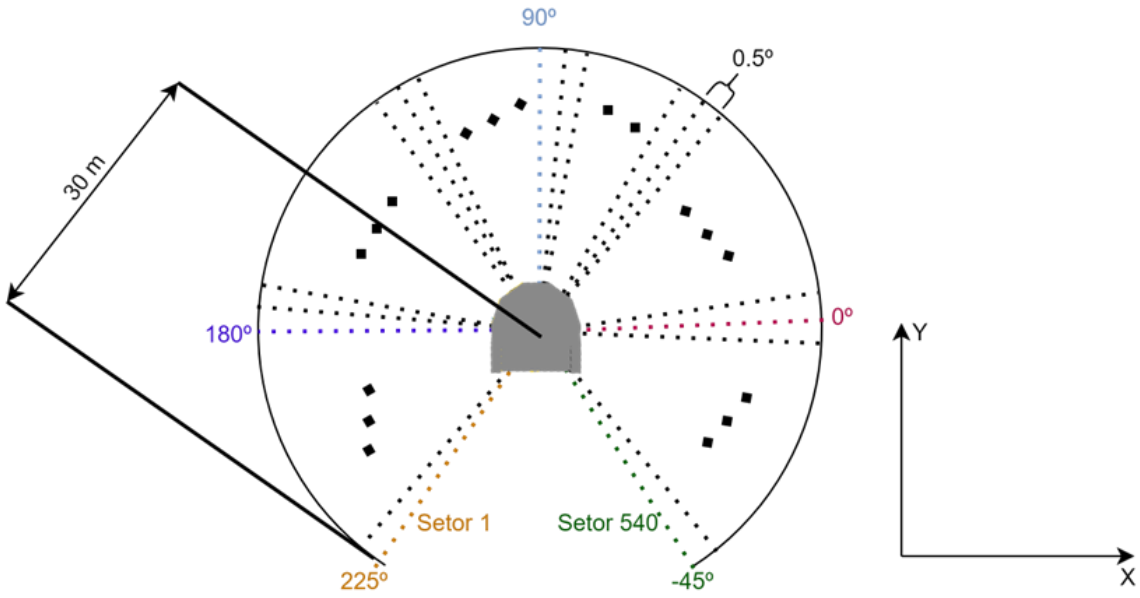


Figura 36: Ilustração da gama de varredura e resolução de medição.

O campo de proteção permite definir zonas de segurança que, quando violadas, obrigam a plataforma a ficar imóvel. Deste modo, e devido à disposição dos sensores na plataforma móvel, cada sensor S300

viabiliza a sensorização simultânea de uma secção lateral e longitudinal da plataforma móvel. A Figura 37 ilustra as zonas de sensorização em função de cada sensor *laser*.

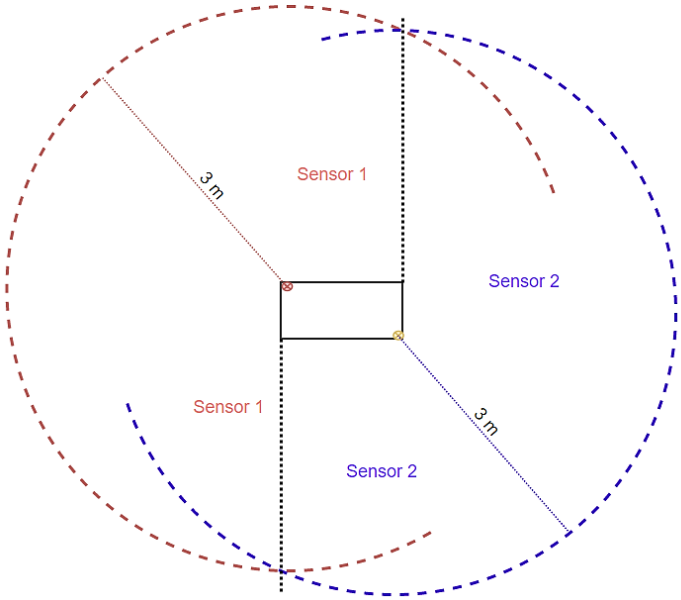


Figura 37: Representação do alcance máximo do campo de proteção de cada sensor. Por sua vez, o alcance máximo de sensorização corresponde a um raio de 30 m.

Uma vez que os sensores *SICK* S300 apenas possibilitam a sensorização num plano bidirecional, no caso o plano cartesiano XoY, a utilização do veículo prossupõe que a altura mínima à passagem do veículo seja verificada em todo o ambiente de navegação da plataforma móvel. Consequentemente, a condição idealizada na Figura 38 deve ser evitada.

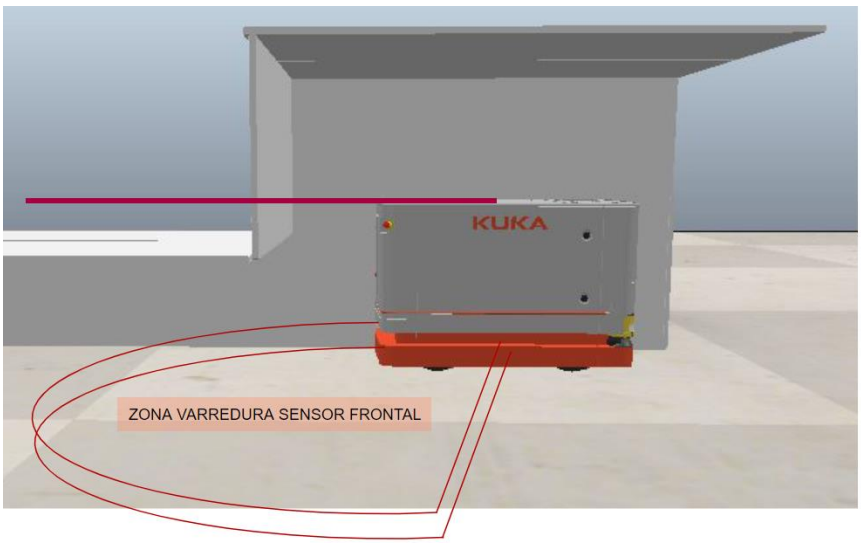


Figura 38: Volume de varredura do sensor laser frontal. Verifica-se que o plano vertical não é sensorizado.

4.4. Cinemática do Veículo

Considerando que o controlador do veículo determina as velocidades linear e angular desejadas para executar um determinado movimento, o interesse neste subcapítulo incide na conversão dos valores do controlador em valores de velocidade angular e sentido de rotação, a aplicar a cada roda omnidirecional da plataforma móvel KMP200.

A análise presente neste subcapítulo foi fundamentada no estudo presente em Taheri et al. (2015), sendo, contudo, adaptado à plataforma móvel anteriormente apresentada.

a) Determinar velocidade linear cartesiana dos rolamentos em relação ao referencial da roda

A velocidade linear de cada roda pode ser projetada nas suas componentes cartesianas, $V_{x_{roda,i}}$ e $V_{y_{roda,i}}$, tendo em consideração a velocidade angular da roda, ω_i, e a velocidade linear dos rolamentos, $V_{rolamento_{roda,i}}$, nas respetivas componentes cartesianas, equação 23 e equação 24. A definição do vetor da velocidade linear dos rolamentos no plano cartesiano depende do ângulo Υ, sendo constante e característico à roda omnidirecional (Figura 39). No caso particular das rodas mecânicas que compõem o KMP200, Υ toma o valor de +45 °.

$$V_{y_{roda,i}} = V_{rolamento_{roda,i}} * \sin(\Upsilon) \tag{23}$$

$$V_{x_{roda,i}} = V_{rolamento_{roda,i}} * \cos(\Upsilon) + \omega_i * Raio_{Roda} \tag{24}$$

A componente das ordenadas, responsável pelo deslizamento, apenas depende da velocidade linear dos rolamentos, ao passo que a componente das abcissas se caracteriza pela influência dos rolamentos assim como da rotação da roda, sobre o seu eixo.

A representação matricial pode ser definida segundo a equação 25.

$$\begin{bmatrix} V_{y_{roda,i}} \\ V_{x_{roda,i}} \end{bmatrix} = \begin{bmatrix} 0 & \sin(\Upsilon) \\ Raio_{Roda} & \cos(\Upsilon) \end{bmatrix} \begin{bmatrix} \omega_i \\ V_{rolamento_{roda,i}} \end{bmatrix} \tag{25}$$

Destaca-se a matriz de transformação ${}^{W_i}T_{P_i}$ que relaciona a velocidade cartesiana do eixo das rodas, $V_{x_{roda,i}}$ e $V_{y_{roda,i}}$, com a velocidade angular, ω_i, e linear, $V_{rolamento_{roda,i}}$, da respetiva roda (equação 26).

$$ {}^{W_i}T_{P_i} = \begin{bmatrix} 0 & \sin(\Upsilon) \\ Raio_{Roda} & \cos(\Upsilon) \end{bmatrix} \tag{26} $$

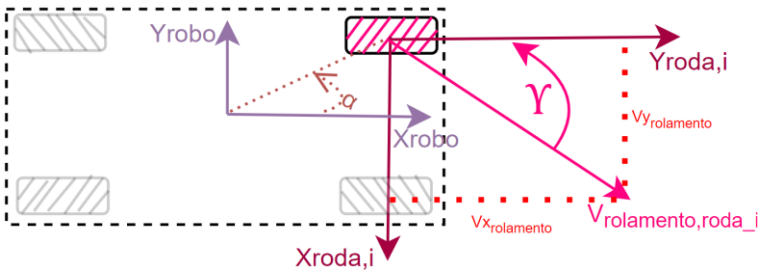


Figura 39: Definição do vetor velocidade linear dos rolamentos no referencial da roda i.

b) Transformação de referenciais

A transformação entre o centro da roda i e o sistema de coordenadas do veículo pode ser definida por uma rotação de β_i, correspondendo ao ângulo entre o eixo das ordenadas da roda i e o eixo das abcissas do sistema de coordenadas do veículo. A relação estabelece-se por uma matriz de rotação bidimensional, ${}^{Pi}T_R$ (equação 27). No caso em estudo, foi considerado uma rotação de $\beta_i$igual a -90˚, conforme a Figura 40.

$$ {}^{Pi}T_R = \begin{bmatrix} \cos\beta_i & -\sin\beta_i \\ \sin\beta_i & \cos\beta_i \end{bmatrix} \tag{27} $$

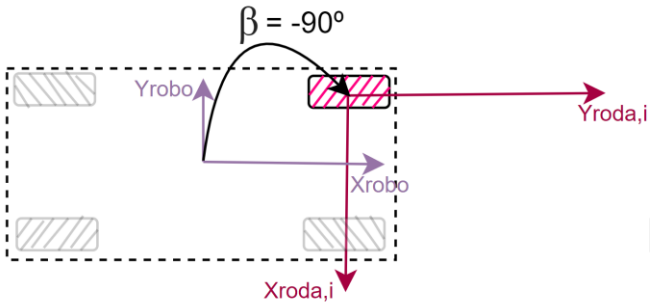


Figura 40: Relação de transformação entre referencial do manipulador e da roda i definido.

Tendo em consideração a matriz de rotação, a velocidade linear de cada roda, projetada no plano cartesiano em relação ao referencial do veículo, pode ser determinada segundo a equação 28.

$$\begin{bmatrix} V_{roda,i_{Xrobo}} \\ V_{roda,i_{Yrobo}} \end{bmatrix} = \begin{bmatrix} \cos\beta_i & -\sin\beta_i \\ \sin\beta_i & \cos\beta_i \end{bmatrix} \begin{bmatrix} V_{x_{roda,i}} \\ V_{y_{roda,i}} \end{bmatrix} \quad (28)$$

c) Velocidade linear das rodas em relação à velocidade linear e angular do veículo (Cinemática Direta)

A locomoção do veículo define-se segundo um plano bidimensional, recorrendo aos três graus de liberdade, estes definidos pela velocidade linear cartesiana bem como pela velocidade angular sobre o seu centro de massa. Desse modo, e considerando que o movimento é planar, a velocidade linear das rodas determina-se segundo as equações 29 e 30, sendo $V_{X_{Robo}}$ e $V_{Y_{Robo}}$ a projeção do vetor velocidade linear do veículo, W_{Robo} a sua velocidade angular sob o centro de massa e l_x e l_y as distâncias entre os referenciais do robô e da roda, conforme a Figura 41.

$$V_{roda,i_{Xrobo}} = V_{X_{Robo}} - W_{Robo} * l_y \quad (29)$$

$$V_{roda,i_{Yrobo}} = V_{Y_{Robo}} + W_{Robo} * l_x \quad (30)$$

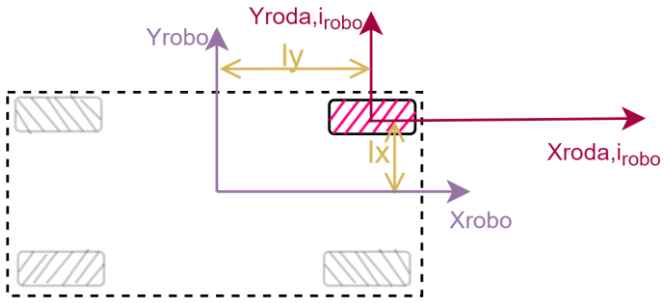


Figura 41: Representação da distância considerada entre eixos. A notação é referente ao referencial da roda i.

A representação matricial da cinemática direta do veículo adotada encontra-se definida pela equação 31, destacando-se a matriz transformação T', equação 32. Para o caso em estudo destaca-se o facto de o veículo ser constituído por quatro rodas omnidirecionais.

$$\begin{bmatrix} V_{roda,i_{Xrobo}} \\ V_{roda,i_{Yrobo}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -l_y \\ 0 & 1 & l_x \end{bmatrix} \begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix} \tag{31}$$

$$T' = \begin{bmatrix} 1 & 0 & -l_y \\ 0 & 1 & l_x \end{bmatrix} \tag{32}$$

d) Determinar cinemática inversa

Recorrendo às relações descritas, a velocidade linear de cada roda em relação ao referencial do veículo pode ser definida pela equação 25, que determina a velocidade linear da roda omnidirecional, no respetivo referencial, e pela matriz de transformação ${}^{Pi}T_R$, equação 28. Dessa forma, a velocidade de cada roda, em relação ao referencial do veículo, define-se segundo a equação 33.

$${}^{W_i}T_{P_i} \cdot {}^{Pi}T_R \cdot \begin{bmatrix} \omega_i \\ V_{rolamento_{roda,i}} \end{bmatrix} = \begin{bmatrix} V_{roda,i_{Xrobo}} \\ V_{roda,i_{Yrobo}} \end{bmatrix} \tag{33}$$

O modelo da cinemática inversa do veículo omnidirecional pode ser obtido pela paridade entre a equação 33 e a equação 31, resultando a equação 34.

$$ {}^{W_i}T_{P_i}.{}^{Pi}T_R.\begin{bmatrix} \omega_i \\ V_{rolamento_{roda,i}} \end{bmatrix} = T'.\begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix} = \begin{bmatrix} V_{roda,i_{Xrobo}} \\ V_{roda,i_{Yrobo}} \end{bmatrix} \quad (34) $$

Desse modo, obtém-se a relação entre a velocidade linear e a velocidade angular das rodas omnidirecionais, em relação ao referencial do veículo, conforme a equação 35.

$$ \begin{bmatrix} \omega_i \\ V_{rolamento_{roda,i}} \end{bmatrix} = {}^{W_i}T_{Pi}^{-1}.{}^{Pi_i}T_R^{-1}.T'.\begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix} \quad (35) $$

Através da equação 35 é possível determinar a velocidade angular de cada roda, ω_i, assim como a velocidade linear dos rolamentos da respetiva roda, $V_{rolamento_{roda,i}}$. De facto, ao isolar a variável ω_i do sistema de equações 35 obtém-se a equação 36.

$$ \omega_i = V_{X_{Robo}}.\frac{-\cos(\beta_i - \Upsilon)}{Raio_{Roda}.\sin(\Upsilon)} + V_{Y_{Robo}}.\frac{-\sin(\beta_i - \Upsilon)}{Raio_{Roda}.\sin(\Upsilon)} + W_{Robo}.\frac{-l.\sin(-\alpha + \beta_i - \Upsilon)}{Raio_{Roda}.\sin(\Upsilon)} \quad (36) $$

Substituindo as variáveis de acordo com as características do KMP200 e considerando o controlo das quatro rodas omnidirecionais, obtém-se o sistema de equações 37, relacionando a velocidade angular e linear da plataforma omnidirecional com a velocidade angular a aplicar a cada uma das rodas.

$$ \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \frac{1}{Raio_{Roda}}\begin{bmatrix} 1 & -1 & -(l_x + l_y) \\ 1 & 1 & (l_x + l_y) \\ 1 & 1 & -(l_x + l_y) \\ 1 & -1 & (l_x + l_y) \end{bmatrix}\begin{bmatrix} V_{X_{Robo}} \\ V_{Y_{Robo}} \\ W_{Robo} \end{bmatrix} \quad (37) $$

4.5. COMPONENTES DE *SOFTWARE*

Considerando os recursos e requisitos de projeto, este subcapítulo pretende apresentar os recursos *software* utilizados para a implementação da estratégia e análise do comportamento do sistema. Por conseguinte, numa primeira instância, é apresentado e justificado a escolha do *framework* usado por base no desenvolvimento da solução, referindo as suas funcionalidades e recursos. Por sua vez, numa

segunda instância, é realizado uma breve descrição dos simuladores mais populares, destacando e justificando a escolha do simulador entre os apresentados, dado os requisitos.

4.5.1. ROS - *ROBOT OPERATING SYSTEM*

O ROS (*Robot Operating System*) é um *framework* para aplicações robóticas que possibilita o desenvolvimento de sistemas robóticos de forma distribuída e interligada. A vasta comunidade ROS e a característica *open source* associada à coleção de *software* disponível, torna a ferramenta amplamente utilizada, tanto a nível amador como a nível académico (Zhengguang Ma et al., 2019) e profissional (Canonical L., 2021). Deste modo, a ferramenta permite que o foco de desenvolvimento de um sistema robótico se concentre na implementação de novas funcionalidades, reaproveitando os recursos presentes no ecossistema ROS. Apesar de não ser classificado como um sistema operativo, este disponibiliza diversos recursos, como mecanismos de comunicação entre processos, abstração da camada de *hardware*, entre outros exemplos (Vivas & Sabater, 2021).

O ecossistema ROS pode ser dividido pelo seu *core*, composto por ferramentas de comunicação, e por um conjunto de bibliotecas que facilitam a implementação de funcionalidades. O ambiente de desenvolvimento permite distribuir e organizar o *software* do sistema robótico por atributos ou funcionalidades, segundo *packages*. Os *packages* são compostos por uma ou mais unidades de processamentos, ou *nodes*, que utilizam os recursos ROS para estabelecerem comunicação entre si (Vivas & Sabater, 2021). Independentemente do tipo, a comunicação define-se por uma estrutura de dados que contém a informação a ser transmitida, denominada de *message.* Estas podem ser constituídas por tipo de mensagens genéricas ROS, bem como por tipos de mensagens personalizadas e específicas à aplicação em causa (Edouard R., 2022). Por sua vez, as mensagens podem ser transmitidas através de ROS *topics* (*publisher/subscriber*), ROS *services* ou *ActionLibs*.

A comunicação via ROS *topic* é baseada na arquitetura *publisher/subscriber* a um determinado tópico (Vivas & Sabater, 2021). A comunicação estabelecida é do tipo assíncrona e unidirecional, ou seja, o emissor não obtém confirmação, por parte do recetor, da receção da informação. No entanto, vários *nodes* podem publicar no mesmo tópico, assim como vários *nodes* podem subscrever o mesmo tópico. Por forma a que a comunicação seja estabelecida, todos os *publishers* e *subscribers* associados ao tópico devem usar o mesmo tipo de dados do respetivo tópico (Open Source Robotics Foundation, 2014). Este tipo de comunicação possibilita a transmissão contínua de informação entre *nodes*. A Figura 42 exemplifica, com algum grau de abstração associado, um possível esquemático de comunicação ROS

topic. No caso, a comunicação é estabelecia pelo tópico */exemplo*, definido pelo tipo de mensagem padrão *String*.

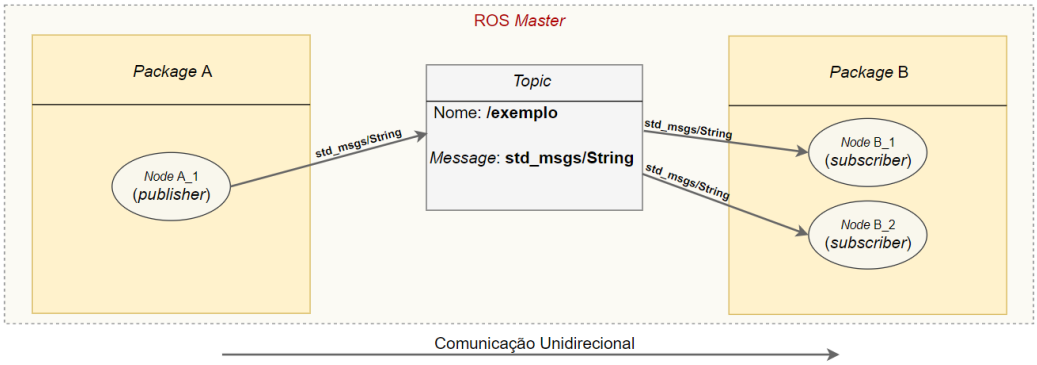


Figura 42: Representação simplificada de um esquemático de comunicação via ROS *topic*.

Por sua vez, o ROS *Service* estabelece comunicação do tipo síncrona e bidirecional, baseada na arquitetura cliente/servidor. A comunicação é realizada recorrendo a serviços, estes caracterizados por um nome único e por dois tipos de mensagem, relativos ao pedido por parte do *node* cliente e à resposta do *node* servidor. Os serviços são definidos em linguagem de descrição, no caso ficheiros "srv", nos *packages* presentes no ambiente ROS (Open Source Robotics Foundation, 2017). Em contraste com o ROS *topic*, o sistema apenas permite, para o mesmo serviço, um único *node* servidor. De facto, um serviço é criado aquando da execução do respetivo servidor (Open Source Robotics Foundation, 2019). Devido à natureza síncrona da comunicação, deve-se minimizar o tempo de resposta ao pedido solicitado, mitigando problemáticas relacionadas com o bloqueio do processo do *node* cliente. Na Figura 43 pode-se observar um possível esquemático de comunicação via ROS *Service* entre dois *nodes*. É de realçar que o sistema permite definir, para cada sentido de comunicação, um tipo de mensagem distinto.

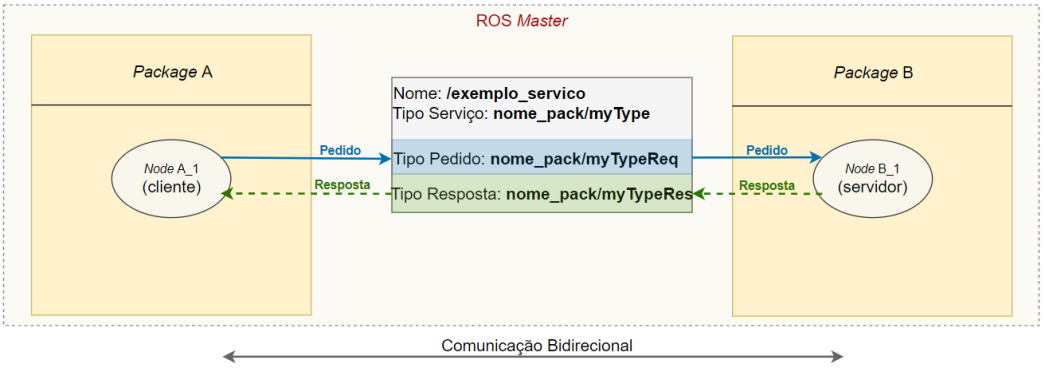


Figura 43: Representação simplificada de um esquemático de comunicação via ROS *Server* entre dois nodes.

Em alternativa ao recurso anterior, o ROS *Action Protocol* implementa um mecanismo de comunicação baseada na arquitetura cliente/servidor, mas do tipo assíncrono. A comunicação é estabelecida entre o *node ActionServer*, que executa um determinado serviço ou funcionalidade, e o *node ActionClient*, que efetua o pedido de execução da tarefa. Tendo como referência o *node* cliente, este pode transmitir dois tipos de mensagem: o *goal,* responsável por enviar novos pedidos ao servidor, e o *cancel*, que solicita ao servidor o cancelamento da ação. Por sua vez, a interface *action* do servidor possibilita o envio de três tipos de mensagens ao cliente: o *feedback*, para enviar informação periódica da ação, o *status*, para informar o estado do processo solicitado, e o *result*, este de envio único aquando da conclusão do *goal* (Open Source Robotics Foundation, 2018). A Figura 44 exemplifica o esquema de comunicação relativo à interface *action* entre o cliente *Action* e o servidor *Action*. Este modelo de comunicação permite a definição dos tipos de mensagem associados ao *goal*, *feedback* e *result*, especificando, deste modo, as mensagens de acordo com os requisitos do projeto.

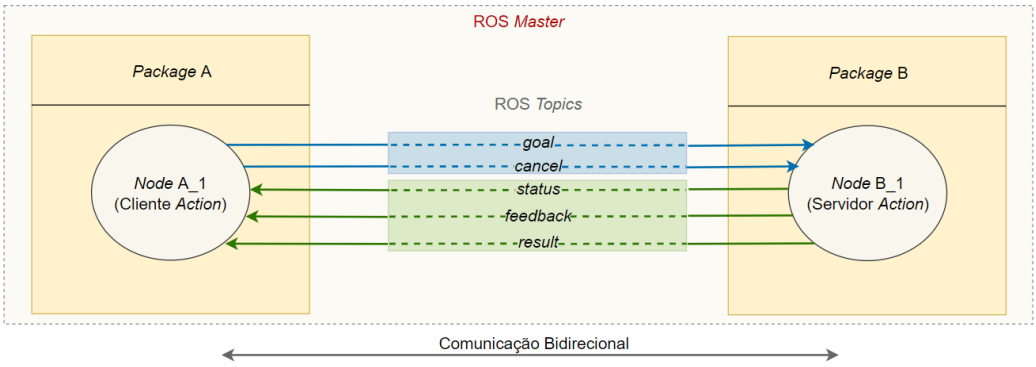


Figura 44: Representação simplificada de um esquemático de comunicação via ROS *ActionLib* entre dois nodes.

Em virtude das caraterísticas e funcionalidades indicadas, o ROS é usado como plataforma principal para a implementação do *software* do sistema proposto.

4.5.2. *SOFTWARE* PARA SIMULAÇÃO: *COPPELIASIM*

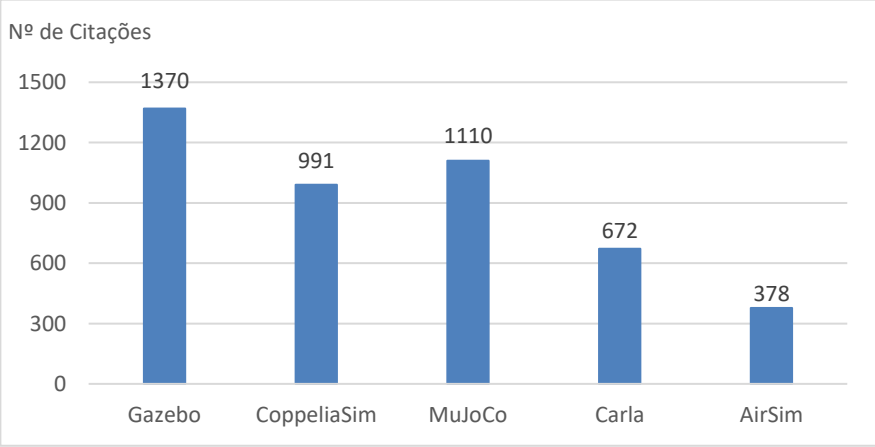
A necessidade da compreensão da realidade circundante e dos seus fenómenos originou o desenvolvimento de ferramentas *software* que simulem os sistemas ou modelos físicos, sendo reconhecida a sua importância desde o início do século 20 (Kumar & Narayan, 2011). A simulação define-se como o processo de *design* e execução de um dado sistema físico real ou teórico, permitindo

o seu estudo e compreensão do comportamento obtido. Ademais, as ferramentas de simulação permitem a deteção de possíveis falhas nas soluções projetadas, mitigando a posterior ocorrência de anomalias e possíveis prejuízos (Kumar & Narayan, 2011).

O crescente aumento das capacidades computacionais e gráficas, resultado da constante evolução tecnológica, proporciona que, atualmente, a simulação de cenários 3D compostos por modelos e ambientes complexos seja uma realidade. De modo geral, um *software* de simulação é constituído por um conjunto de bibliotecas que implementam a cinemática, motor de física e interface gráfica. O modo como estes elementos interagem entre si influencia o desempenho e precisão do *software* de simulação, sendo, por esse motivo, aspetos cruciais na arquitetura e metodologias de controlo que constitui o simulador (Eric Rohmer et al., 2013).

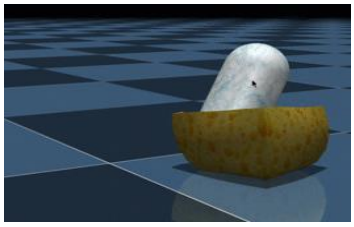
As ferramentas de simulação computacional são, atualmente, recursos essenciais para o desenvolvimento dos diversos tipos de projetos robóticos. Esta premissa fundamenta-se na constante verificação de citações de projetos de investigação onde inclui a utilização de pelo menos um *software* de simulação como ferramenta de auxílio à análise e validação de resultados do projeto. Dentro dos mais variados simuladores de robótica, destacam-se os que são atualmente mais utilizados que, por norma, são aqueles que além de disponibilizar atualizações regulares, também incluem uma vasta gama de recursos e funcionalidades, destacando-se os apresentados na Tabela 4 (Collins et al., 2021):

Tabela 4:Número de citações dos simuladores mais populares (contabilizados entre 2016 e 2020). Informação recolhida de Collins et al. (2021).



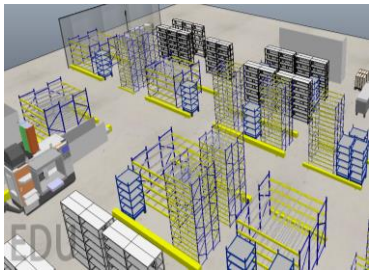
Os simuladores enunciados representam o estado da arte de ferramentas computacionais de auxílio ao desenvolvimento de projetos robóticos, divergindo, alguns dos indicados, do enquadramento da área de investigação. O *AirSim* (Figura 45 b)) é apresentado como sendo um simulador de robótica aérea,

dispondo de recursos relacionados com a temática. Deste modo, devido à sua natureza, este simulador é excluído como possível ferramenta computacional para o projeto de dissertação. Por sua vez, sendo o *Carla* (Figura 45 e)) projetado como um simulador de condução autónoma direcionado para veículos de direção diferencial em ambientes externos, não se ajusta aos requisitos de projeto Os restantes simuladores indicados na Tabela 4 apresentam funcionalidades semelhantes, diferindo, contudo, no foco de aplicações especializados e, consequentemente, nos recursos associados às respetivas áreas de especialização. O *MuJoCo* (*Multi-Joint dynamics with Contact*) é um simulador com motor de física de uso geral especializado na dinâmica de contacto com os objetos (Figura 45 a)), sendo este um dos motivos da sua popularidade entre a comunidade. Contudo, o *software* não apresenta suporte com o *middleware* ROS, recurso essencial neste projeto de dissertação. A popularidade do *Gazebo* (Figura 45 d)) deve-se à vasta gama de aplicações e recursos disponibilizados e à sua integração com o ROS, tornando-se amplamente utilizada na comunidade ROS (Collins et al., 2021). Em paridade, o *CoppeliaSim* (Figura 45 c)) apresenta-se como um simulador multiplataforma de uso geral, destacando-se pela sua versatilidade e facilidade de customização dos ambientes de simulação. Ambos os simuladores são compostos por motores de física, possibilitando uma maior aproximação ao mundo real.



(a) Ambiente Simulação *software MuJoCo*

(b) Ambiente Simulação *software AirSim*



(c) Ambiente Simulação *software CoppeliaSim*

(d) Ambiente Simulação *software Gazebo*

(e) Ambiente Simulação *software Carla*

Figura 45: Ambiente de cenário das ferramentas de simulação analisadas.

Considerando o projeto de dissertação, o comportamento do sistema deve ser verificado e analisado num simulador robótico 3D versátil, o qual seja composto por uma vasta biblioteca de objetos virtuais tridimensionais e que permita o controlo e comunicação do sistema robótico concebido por meios

externos ao *software*. A escolha do simulador *CoppeliaSim* fundamenta-se, dentro de outros aspetos, no facto de cumprir com os requisitos anteriormente enunciados, bem como possibilitar a importação de modelos CAD e monitorização de variáveis na caixa de diálogo, esta presente na interface gráfica do *software*, além de permitir estabelecer comunicação via ROS ou bibliotecas *RemoteApi*. Ademais, o *software* pode ser executado nos três sistemas operativos mais populares.

O *CoppeliaSim* é um simulador robótico de propósito geral que auxilia o desenvolvimento e validação de algoritmos e sistemas robóticos, de forma versátil e escalável, num dado ambiente de simulação (Bogaerts et al., 2020). O simulador permite o controlo de modelos num registo distribuído, possibilitando a conceção de cenários complexos compostos por múltiplos modelos dinâmicos a controlar (Figura 46). No exemplo multi-robô, estes podem ser controlados num outro processo ou *thread* na mesma unidade computacional, balanceando a exigência de cálculo entre os *cores* do CPU, ou numa unidade computacional dedicada, como noutro computador ou o próprio robô. Esta última possibilidade, além de reduzir a carga computacional no computador que executa a simulação, permite a representação virtual do objeto físico, no caso o *digital-twin* (Eric Rohmer et al., 2013).

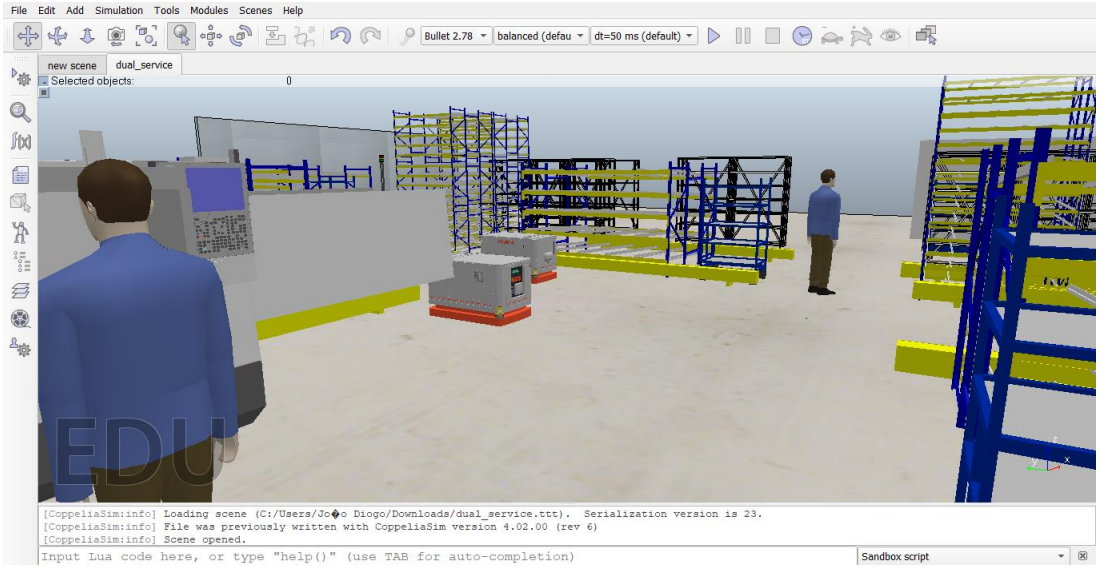


Figura 46: Exemplo cenário complexo, constituído por múltiplos modelos dinâmicos.

A arquitetura do simulador caracteriza a versatilidade e escalabilidade do *software*, dispondo de múltiplas técnicas de programação, alargando o espetro de aplicações e recursos. Genericamente, o simulador é constituído por três elementos: os objetos de cenário, os módulos de cálculo e as técnicas ou mecanismos de controlo (Shamshiri et al., 2018). Dada a relevância dos elementos na arquitetura do

simulador, estes são analisados individualmente e destacados os utilizados como recurso ao projeto de dissertação.

a) Mecanismos de Controlo: Técnicas de Entidade de Controlo:

- ***Embedded Script***: Mecanismo característico do simulador *CoppeliaSim*. Permite a customização de um cenário ou modelo via *scripts*, estas escritas em linguagem própria, *Lua*, ou em *Python*, a partir da versão 4.3. Um cenário, por defeito, necessita de uma *script* principal, responsável pelo funcionamento do simulador. De modo a garantir a compatibilidade e execução dos modelos entre cenários, esta *script* não deve ser modificada. A *script* principal é também responsável por invocar as *scripts* filho, respeitando a hierarquia de simulação. As *scripts* filhos, ao contrário da principal, são associadas a um objeto ou modelo. Tendo como exemplo o projeto em estudo, estas executam o controlo de baixo nível dos modelos ou objetos em função da informação sensorial recolhida, no caso dos sensores *lidar*, ou da informação recebida do controlador, no caso da atuação dos motores.
- ***Add-ons*** ou ***sandbox***: Método implementado via *scripts*, escritas em linguagem *Lua* ou *Python*, que permite uma rápida customização do simulador. Este recurso não deve ser associado a um cenário específico, mas a funcionalidades genéricas do simulador.
- ***Plug-ins***: Método de customização do cenário via *plugins*. Este é considerado um recurso polivalente, permitindo desde o registo de comandos *API* próprios ao projeto, neste caso considerado como mecanismo auxiliar ao *embedded script*, até a implementação de funcionalidades adicionais ao simulador, como por exemplo, estabelecer interfaces de comunicação externas, como a interface *ROS* ou o *remote API*.
- Cliente ***remote API***: A interface *remote API* possibilita a conexão de aplicações externas ao simulador, via comunicação *socket*. A comunicação é definida por serviços, concebidos pelo servidor *remote API*, e por clientes *remote API*. Este recurso permite a comunicação multiplataforma, admitindo por exemplo, estabelecer comunicação com uma aplicação pertencente a um sistema robótico real.
- ***Node*** ROS: Método que permite estabelecer comunicação externa multiplataforma. Esta comunicação pode ser estabelecida via serviços ROS, que permite executar comandos *CoppeliaSim*, ou via ROS *publisher/subscriber*, esta adequada para a troca de informação. O *CoppeliaSim* implementa a comunicação ROS via *plugin*. Considerando o atual projeto, devido à versatilidade do ROS, o fluxo de informação entre o simulador e os algoritmos de controlo é estabelecido via ROS *node*.
- ***ZeroMQ node***: Método que permite estabelecer comunicação com aplicações externas, via funções *API*.

Os mecanismos e técnicas de controlo suportados pelo simulador encontram-se descritos, concisamente, na Figura 47, sendo destacado os característicos ao projeto.

	Embedded script	Add-on / sandbox script	Plugin	Remote API client	ROS / ROS2 node	ZeroMQ node
Control entity is external (i.e. can be located on a robot, different machine, etc.)	No	No	No	Yes	Yes	Yes
Difficulty to implement	Easiest	Easiest	Relatively easy	Easy	Relatively easy	Easy
Supported programming language	Lua, Python	Lua, Python	C/C++	C/C++, Python, Java, JavaScript, Matlab, Octave	Any [1]	Any
Code execution speed	Relativ. fast [2]	Relativ. fast [2]	Fast	Depends on programming language	Depends on programming language	Depends on programming language
Communication lag	None [3]	None [3]	None	Yes	Yes	Yes
Control entity can be fully contained in a scene or model, and is highly portable	Yes	No	No	No	No	No
Control entity relies on	CoppeliaSim	CoppeliaSim	CoppeliaSim	Sockets, ZeroMQ or WebSockets	ROS / ROS2 framework	ZeroMQ
Stepped operation [4]	Yes, inherent	Yes, inherent	Yes, inherent	Yes	Yes	Yes
Non-stepped operation [4]	Yes, via threads	Yes, via threads	No (threads available, but API access forbidden)	Yes	Yes	Yes

Figura 47: Mecanismos e técnicas de controlo suportadas pelo *CoppeliaSim*. Destaque dos métodos utilizados no projeto. Imagem adaptada de Coppelia Robotics L. (2022).

b) De Objetos de Cenário a Modelos de Cenário:

Em *CoppeliaSim*, os objetos de cenário são elementos primários que permitem o desenvolvimento de cenários ou modelos de simulação. Atualmente, o simulador apresenta uma vasta gama de objetos *standard*, distribuídos por 12 coleções, conforme ilustrado na Figura 48. Os objetos que compõem o cenário ou modelo podem ser identificados na hierarquia de cenário ou, tridimensionalmente, em ambiente de simulação.

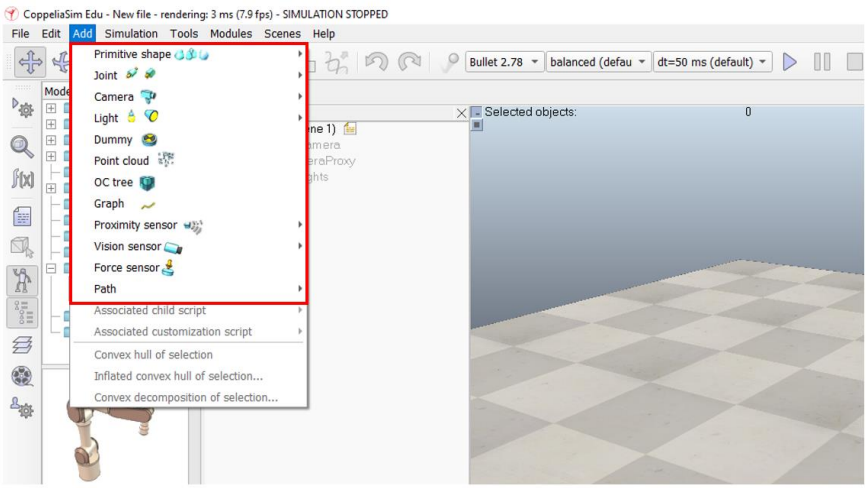


Figura 48: Biblioteca de objetos suportados distribuída por 12 coleções distintas. Cada coleção é identificada por uma simbologia *CoppeliaSim*, conforme ilustrado na figura.

O *software* permite o ajuste dos objetos de cenário conforme as necessidades e características de projeto, oferecendo uma panóplia de propriedades que podem ser ajustadas, destacando-se as 'propriedades especiais'. Estas permitem definir características de interação entre objetos e, consequentemente, entre modelos. Conforme ilustrado na Figura 49, alguns dos objetos podem ser caracterizados como sendo responsivos a colisões, medíveis e detetáveis, permitindo a ocorrência de colisões entre objetos, o cálculo de distâncias entre objetos e caracterizar se estes são detetáveis pelos sensores de proximidade, respetivamente.

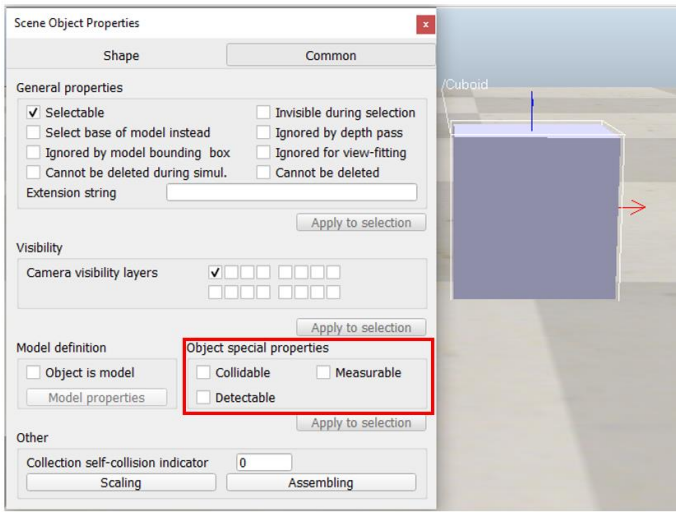


Figura 49: Propriedades especiais associadas aos objetos *CoppeliaSim*. Dependendo do tipo do objeto, alguma das propriedades pode não estar disponíveis.

Por definição, os modelos correspondem a uma seleção de objetos de cenário compostos na mesma estrutura hierárquica. Um cenário pode ser composto por múltiplos modelos, sendo estes assinalados como base de modelo na hierarquia de cenário, conforme ilustrado na Figura 50. É de destacar que o comportamento de um modelo não pode ser verificado de forma isolada, mas num cenário de simulação. Na Figura 50 encontra-se representado a estrutura do modelo de simulação correspondente à plataforma móvel *KMP OmniMove* 200.

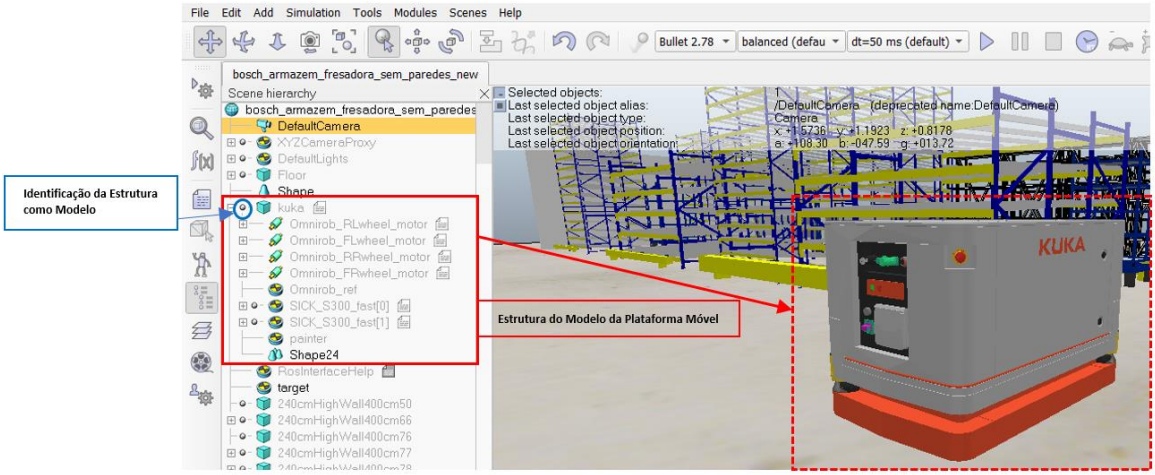


Figura 50: Identificação do modelo de simulação da plataforma móvel. Na hierarquia é possível identificar os objetos que modelam a estrutura.

c) Módulos de Cálculo:

Nativamente, o *CoppeliaSim* disponibiliza um conjunto de módulos de cálculo gerais, subjacente aos cenários e modelos de simulação. Como Eric Rohmer et al., (2013) refere, a integração dos módulos no *software* permite que os modelos e cenários desenvolvidos sejam versáteis e compatíveis entre ambientes de simulação e plataformas distintas, diminuindo a dependência de bibliotecas externas que implementem as funcionalidades.

Os módulos de cálculo implementados no simulador operam diretamente nos objetos de cenário, individualmente ou em interação múltipla, conforme a funcionalidade. Os cálculos são dependentes da formulação dos objetos, isto é, das malhas triangulares utilizadas para a visualização e simulação dos corpos rígidos (Eric Rohmer et al., 2013). Como ilustrado na Figura 51, um modelo pode ser constituído por um conjunto de objetos que, por sua vez, são formulados por malhas triangulares, designada de *shapes*, se analisado ao nível do *wireframe*.

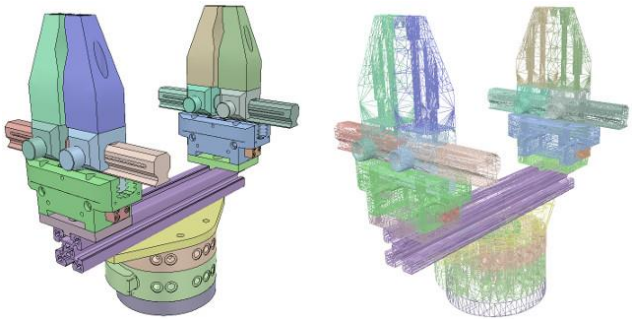


Figura 51: Representação do modelo de uma garra em formato sólido e respetiva representação *wireframe*. Imagem retirada de Coppelia Robotics, L (2022).

Entre as funcionalidades implementadas, destacam-se as seguintes (Eric Rohmer et al., 2013):

- Cálculo das cinemáticas direta e inversa;
- Cálculo da dinâmica de objetos e interação entre si, via motor de física;
- Verificação independente de colisões entre *shapes* (independente da dinâmica de objetos);
- Cálculo de distâncias mínimas entre *shapes*;
- Planeamento de movimentos, para modelos holonómicos e não-holonómicos;

5. ARQUITETURA DO SISTEMA

A arquitetura do sistema encontra-se definida em três módulos gerais, incorporando as tarefas previstas tendo em consideração as restrições e condições de navegação em espaços dinâmicos, partilhados com operadores humanos. O *Service Manager* realiza a função de interface de comunicação entre o sistema robótico (manipulador móvel) e o mundo exterior. O módulo recebe um serviço completo a ser realizado pelo sistema, dividindo-o num conjunto de *tasks*. Posteriormente, o *Service Manager* solicita a execução de cada *task* ao *Movement Controller*. Em função da *task* requerida, o *task manager*, uma das componentes do *Movement Controller*, gere os módulos associados ao movimento, orquestrando-os conforme a condição atual do sistema. O módulo de gestão de movimentos requer informação do posicionamento no espaço, assim como do espaço circundante ao manipulador. Esta informação é fornecida pelo módulo *Environment Perception*. O controlador estabelece comunicação com o módulo *interface*, abstraindo-se do mesmo devido à mediação de um *middleware*. A Figura 52 ilustra os módulos e respetivos componentes que formulam a arquitetura do sistema.

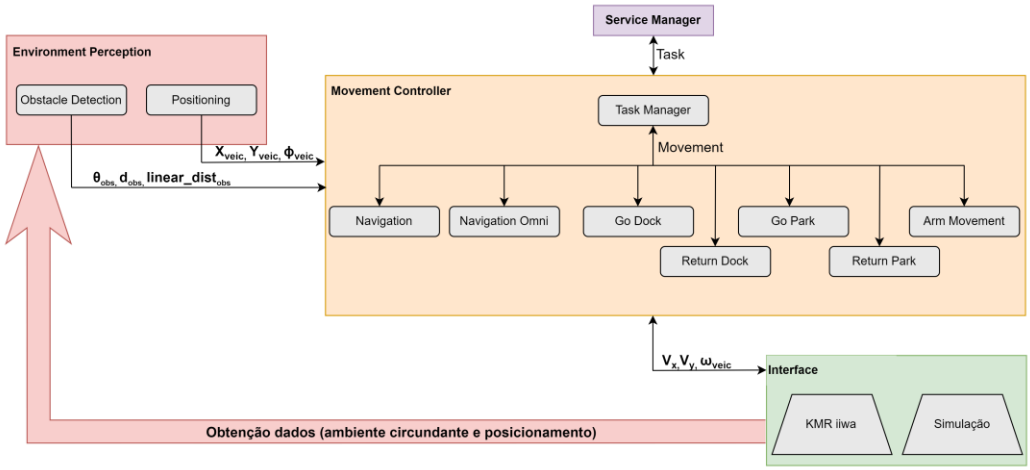


Figura 52: Arquitetura do sistema definida em três módulos gerais, permitindo o controlo e gestão dos comportamentos previstos. O *middleware* ROS permite a comunicação entre o sistema de controlo e a interface considerada.

5.1. *SERVICE MANAGER*

O módulo *Service Manager* permite que seja possível estabelecer comunicação entre um servidor externo e o sistema robótico proposto. Este recebe um serviço completo a ser executado pelo manipulador móvel. De facto, o módulo divide o serviço recebido num conjunto de *tasks* individuais a serem realizadas pelo sistema.

Cada *task* encontra-se definida por um *action goal*, caracterizando o método de acostagem a realizar, a rota a ser percorrida pela plataforma omnidirecional e a orientação final de acostagem. O *Service Manager* estabelece comunicação via *ActionLib* (Figura 53), "*task.action*", com o *task manager*, permitindo, deste modo, obter *feedback* e estado final de cada *task*.

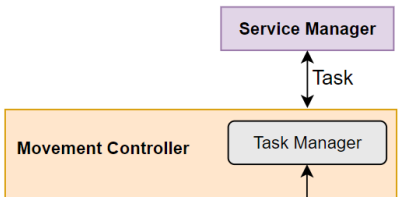


Figura 53: Comunicação via *ActionLib* entre módulos *Service Manager* e o *Movement Controller*.

A Tabela 5 expõe a estrutura de mensagem relativa à comunicação bidirecional entre o *Service Manager* e o *task manager*.

Tabela 5: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *Service Manager* e o *task manager*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	action_goal	string	Tipo de acostagem (*dock* ou *park*)
	route	miar_msgs/GraphNode	Definição da rota a ser percorrida
	action_orientation	float	Orientação final de acostagem
Result	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado final da tarefa
Feedback	route_performed	miar_msgs/GraphNode	Rota já percorrida

5.2. *MOVEMENT CONTROLLER*

O *Movement Controller* é um módulo que contém as componentes responsáveis por definir o movimento da plataforma móvel, bem como do braço manipulador. A Figura 54 ilustra os seis tipos de comportamentos que podem definir o movimento do veículo, assim como o de geração de trajetórias para o braço manipulador. Como verificável, todos os módulos estabelecem comunicação com o *task manager*, responsável pela coordenação, gestão e controlo do tipo de comportamento ativo.

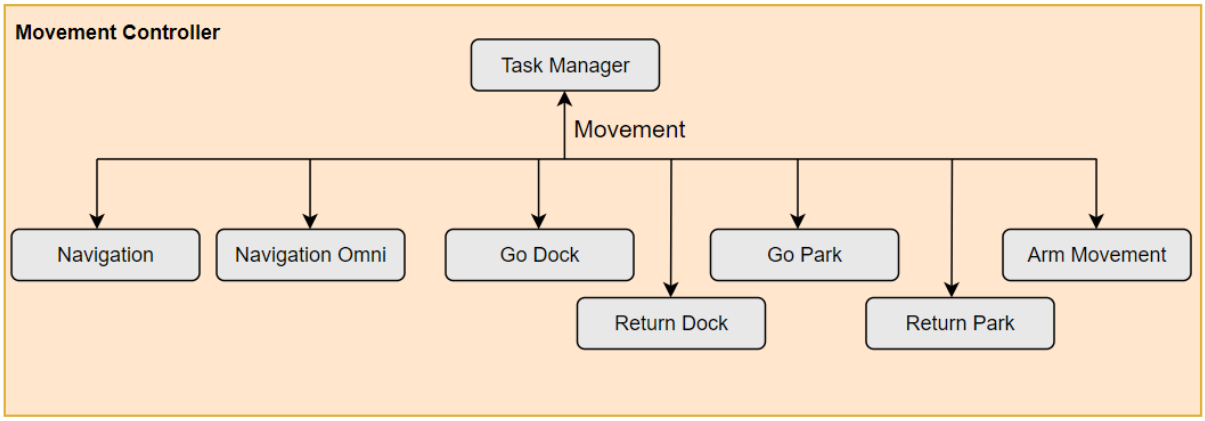


Figura 54: Representação dos 6 tipos de comportamentos que podem definem o movimento do veículo. O *arm movement* é responsável pelo controlo do movimento do braço robótico da plataforma móvel. Todos os comportamentos estabelecem comunicação bidirecional com o *task manager*.

O movimento da plataforma móvel pode ser definido por uma das seis componentes, nomeadamente o de navegação com movimentos legíveis ao operador humano, de navegação recorrendo às características holonómicas e dos processos de aproximação e afastamento aos locais de acostagem. Estes últimos encontram-se subdivididos em duas categorias: o *dock* e o *park*. O *dock* é composto pelos processos de aproximação e afastamento aos locais de trabalho, enquanto o *park* é relativo ao local de estacionamento e carregamento do manipulador móvel. A geração de movimentos do braço colaborativo encontra-se definido pela componente *arm movement*. Como o foco de estudo incide na navegação autónoma de um veículo omnidirecional, dá-se especial relevância às componentes relacionadas com a geração de movimento da plataforma móvel. O *task manager* é a componente que recebe as tarefas do *Service Manager*, simplificando-as em operações elementares, interpretáveis pelas componentes de geração de movimento. Conforme a tarefa recebida, o *task manager* coordena a ativação das respetivas componentes do *Movement Controller*.

5.2.1. *TASK MANAGER*

O *task manager* implementa o algoritmo de gestão de tarefas e monitorização da execução de cada uma destas. As tarefas, recebidas sequencialmente do *Service Manager*, são executadas pela gestão dos recursos presentes no modulo *Movement Controller*. Dessa forma, o *task manager* é responsável pela receção das tarefas a realizar, pela orquestração dos diversos comportamentos, pela monitorização do estado de execução de cada tarefa e, ainda, pelo retorno do estado do processo ao *Service Manager*.

A seleção do comportamento é dependente do tipo de tarefa recebida, assim como do estado atual da plataforma móvel. Como tal, numa primeira fase, o módulo verifica qual a última tarefa realizada, podendo ser de *dock* ou de *park*. Conforme o *goal* recebido, o algoritmo seleciona o comportamento de *return_dock* ou de *return_park*, respetivamente. Na primeira iteração, assume-se que o sistema se encontre em *park*, realizando o *return_park*. Caso o último *goal* seja inválido (sistema contido num estado diferente de *dock* ou *park*), o veículo permanece imóvel, retornando uma mensagem de erro ao *Service Manager*. Uma vez terminado com sucesso, o *task manager* seleciona o comportamento seguinte em função da informação da rota recebida na mensagem associada à tarefa. Se a variável *narrow_area* se encontrar ativa, o veículo deve navegar em modo omnidirecional, caso contrário deve-se movimentar de forma legível aos operadores humanos. Após a validação do resultado da operação, o *task manager* seleciona o tipo de manobra de acostagem a realizar, no caso de corresponder ao último *via point* da tarefa. Se esta condição não ocorrer, é realizada uma nova verificação da variável *narrow_area*. O tipo de manobra de acostagem a definir é selecionado em função do *goal* da tarefa ativa, podendo se tratar de um local de trabalho ou de estacionamento, ativando o comportamento *go_dock* ou *go_park*, respetivamente.

Na Figura 55 encontra-se representado o algoritmo anteriormente descrito, em formato de fluxograma. Este algoritmo é executado cada vez que o *task manager* recebe uma nova tarefa válida do *Service Manager* e se a última tarefa estiver finalizada.

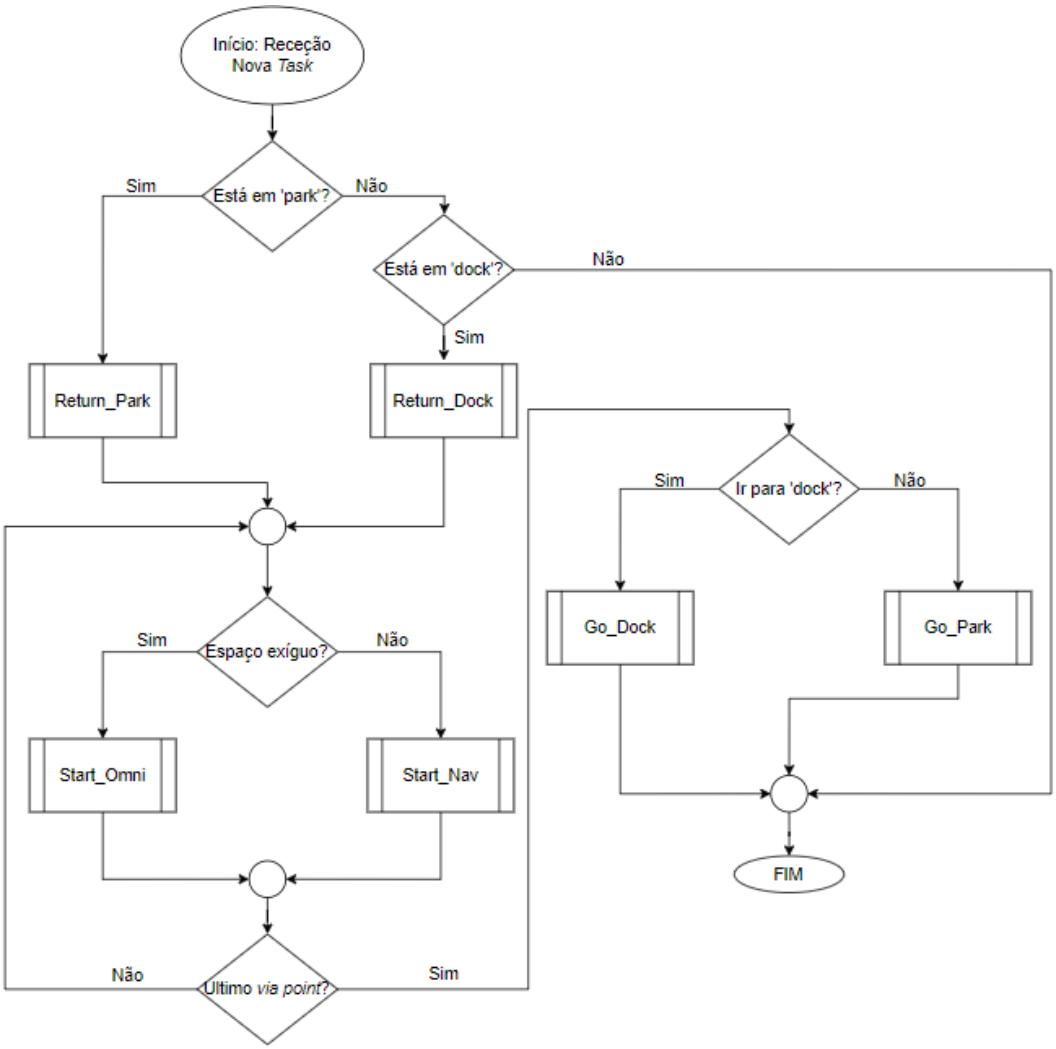


Figura 55: Fluxograma do algoritmo de gestão e seleção de comportamentos executado no módulo *task manager*.

As componentes do módulo *Movement Controller* comunicam via ROS *action server*, ou *ActionLibs*, com o *task manager*, estabelecendo uma comunicação bidirecional do tipo assíncrona. Na solução proposta, para cada comportamento, o *task manager* encontra-se definido como cliente, permitindo requisitar serviços aos módulos, obter periodicamente o estado da tarefa ativa e ainda, ter a possibilidade de requisitar o cancelamento da sua execução.

5.2.2. *NAVIGATION*

A componente *navigation* implementa o comportamento de navegação autónoma entre dois pontos previamente definidos, em ambientes complexos dinâmicos, de forma consciente e segura, considerando a partilha de chão de fábrica com operadores humanos. Neste regime, a plataforma omnidirecional deve descrever movimentos legíveis aos operadores humanos, facilitando, deste modo, a perceção da intenção

de manobra do manipulador móvel. Ao nível de gestão de comportamentos, a componente estabelece comunicação com o *task manager*, via *ActionLib*, aferindo o recurso de *server* na comunicação. Dada a relevância e o destaque do comportamento para a solução em estudo da dissertação, as características e detalhes de implementação encontra-se descritos de forma mais pormenorizada no capítulo Implementação.

5.2.3. *NAVIGATION OMNIDIRECTIONAL*

A componente *navigation omnidirectional*, identicamente ao verificado no comportamento *navigation*, implementa a capacidade de navegar autonomamente, num ambiente complexo dinâmico, considerando ainda a partilha do espaço com operadores humanos. Contudo, este comportamento distingue-se do anterior no facto de poder estabelecer movimentos de características holonómicas, isto é, não intuitivas aos seres humanos. Esta faculdade permite a operação do sistema em espaços mais exíguos e, consequentemente, mais exigentes. Considerando os requisitos de projeto, este regime de funcionamento adequa-se, por exemplo, durante a navegação nos corredores de acesso aos locais de trabalho. A componente estabelece comunicação com o *task manager* via *ActionLib*, desempenhando a função de *server* da comunicação. As características e detalhes de implementação encontram-se analisadas com maior detalhe no capítulo Implementação.

5.2.4. *GO TO DOCK* E *GO TO PARK*

As componentes *go to dock* e *go to park* implementam os métodos que capacitam o sistema de realizar os processos de acostagem aos locais de acostagem previamente definidos. As componentes fundamentam-se no mesmo tipo de estratégia de controlo, beneficiando da maior flexibilidade e manobrabilidade apresentada pela plataforma omnidirecional. A distinção entre as componentes deve-se às restrições de movimento no momento de aproximação, sendo estas aspetos fulcrais no método implementado no *go to park*, conforme indicado no capítulo Implementação. Ambas as componentes estabelecem comunicação independente com o gestor *task manager*, via ROS *ActionLib*, incumbindo-se da função de *server* das comunicações. A estratégia de controlo e a diferenciação entre as componentes são descritas com maior detalhe no capítulo Implementação.

5.2.5. *Return from Dock* e *Return from Park*

As componentes *return from dock* e *return from park* implementam os métodos que possibilitam ao sistema realizar as operações de afastamento aos locais de acostagem. Considerando as aplicações em estudo nesta dissertação, ambas as componentes partilham as estratégias de geração e controlo de movimentos, particularizando-se dos métodos de aproximação na estratégia de variação do módulo da velocidade. A distinção das estratégias de afastamento deve-se à diferenciação do tipo de tarefa a realizar, por parte do *task manager*, bem como à possibilidade de serem implementados mecanismos específicos a determinada componente. Identicamente às operações de aproximação, as componentes estabelecem a propriedade de *server* na comunicação com o *task manager*, via ROS *ActionLib*. Cada componente estabelece uma comunicação independente com o gestor de tarefas. Os fundamentos e características de controlo e implementação são descritas com maior detalhe no capítulo Implementação.

5.3. *Environment Perception*

O *Environment Perception* é um módulo que possibilita a recolha, tratamento e processamento de informação sensorial relativa ao espaço circundante ao veículo, assim como da sua localização no ambiente de navegação. Para o efeito, o módulo encontra-se definido pelas componentes de deteção de obstáculos e de posicionamento no espaço. Cada componente estabelece comunicação unidirecional (Figura 56) com os comportamentos que definem o tipo de movimento, via ROS *topics*, enviando os dados segundo os formatos adequados.

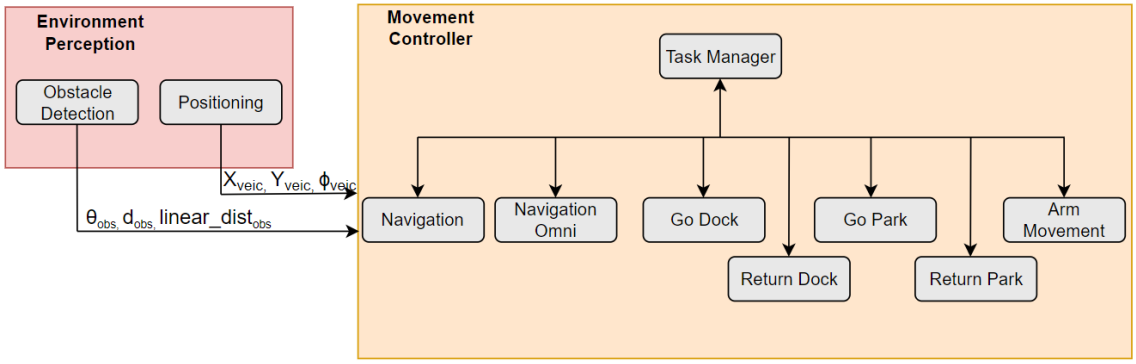


Figura 56: Esquemático de comunicação entre os módulos *Environment Perception* e *Movement Controller* e respetivas componentes.

Na Figura 57 encontra-se ilustrado um possível esquema de comunicação entre os módulos *Movement Controller* e *Environment Perception*. De forma genérica, é verificável que a componente

obstacle detection publica dois tipos de mensagens distintas, estes relativos aos setores da dinâmica e distâncias lineares de segurança. Por sua vez, a componente *positioning* publica a informação de posicionamento no espaço de navegação. Ambas as componentes partilham os dados com os módulos responsáveis pela geração de movimentos, em intervalos periódicos pré-definidos.

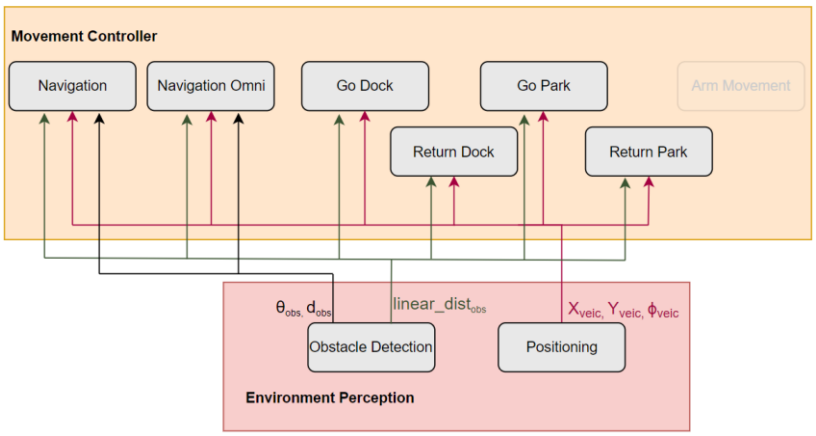


Figura 57: Esquemático de comunicação pormenorizado entre as componentes que compõem os módulos *Environment Perception* e *Movement Controller*. Como se verifica, as componentes *navigation* e *navigation omni* necessitam ainda da informação dos setores de interesse à dinâmica de navegação, enviados pela componente *obstacle detection* via ROS *topic*.

5.3.1. *OBSTACLE DETECTION*

A componente *obstacle detection* está incumbida dos métodos de recolha, processamento e manipulação dos dados recolhidos dos sensores *laser lidar*. A componente, com base na informação dos dois sensores *Sick* S300, realiza dois métodos de processamento distintos, um relativo à conversão dos dados em informação organizada em setores de interesse à dinâmica de navegação, holonómica e não-holonómica, e outro referente ao cálculo das distâncias de segurança lineares, entre o corpo do veículo e os obstáculos. Ademais, dada a disposição dos sensores na plataforma móvel, os dados recolhidos são moldados por uma transformação de referenciais, relativamente ao centro de massa do manipulador móvel. Apesar de a componente não comunicar diretamente com o *task manager*, esta estabelece comunicação, via ROS *topic*, com todas as componentes comportamentais que definem o módulo *Movement Controller*, excluindo o *arm movement*, publicando a informação do espaço circundante ao sistema a cada ciclo de computação. As estratégias e fluxo de controlo são descritas com maior detalhe no capítulo Implementação.

5.3.2. *POSITIONING*

A componente *positioning* é responsável por determinar a posição e orientação do veículo no ambiente de navegação. Esta componente é crítica aos módulos de geração de movimentos e, como tal, deve ser precisa e robusta a ruídos e interferências internas e externas ao sistema.

A conceção de um sistema de localização em espaços interiores enquadra-se como objetivo primário de um outro trabalho de investigação pertencente ao projeto. Contudo, para o trabalho de desenvolvimento em causa, os dados podem, nesta primeira fase, serem obtidos a partir do ambiente de simulação computacional. Desta forma, é possível verificar e validar o comportamento do sistema para as diversas condições de testes, considerando a coordenação dos módulos de geração de movimentos.

A componente concebida recebe os dados de localização do ambiente de simulação através do tópico */pose_vehicle_world*, obtendo a posição cartesiana no espaço, bem como a orientação em quaternião. Apesar da posição cartesiana ser diretamente publicada para os módulos, tal não se verifica para a orientação. Os dados de orientação espacial são convertidos em ângulos de *Euler*, extraindo o ângulo *yaw*, correspondente à orientação do veículo, como analisado no capítulo Identificação da *Pose* de um Veículo no Espaço. Conforme o ilustrado na Figura 58, os dados tratados são publicados para os módulos de geração de movimentos, pelo tópico */vehicle_pose_phirobot*.

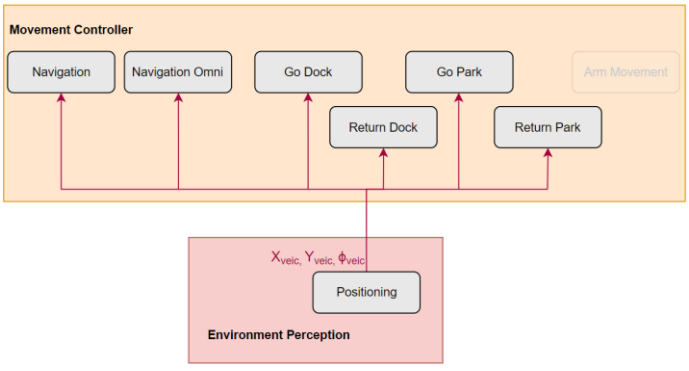


Figura 58: Comunicação via ROS *topic* entre a componente *positioning* e todas as componentes do módulo *Movement Controller* que subscrevem ao tópico */vehicle_pose_phirobot*.

Dada a criticidade do módulo, a componente apenas publica os dados de localização na condição de estes serem os mais recentes. O tópico é definido pela variável *geometry_msgs::Pose2D*, já definida no ambiente ROS, contendo a posição cartesiana, (x_i, y_i), e orientação do veículo em relação ao referencial do mundo, Φ_{yaw}.

6. NAVEGAÇÃO OMNIDIRECIONAL

O módulo de navegação omnidirecional implementa o comportamento de navegação holonómica em espaços dinâmicos complexos, possibilitando a navegação em espaços mais exíguos e exigentes ao nível da manobrabilidade. Na estratégia adotada, a navegação holonómica traduz-se no controlo individual dos três graus de liberdade do veículo, formulando movimentos caracterizados pelo módulo e sentido do movimento linear, como também pela velocidade angular do mesmo. O método de controlo fundamenta-se na dinâmica de navegação formulada pelos sistemas dinâmicos, lineares e não-lineares, partilhando os comportamentos de seguir em direção a um alvo e de evitar obstáculos.

6.1. DINÂMICA CONTROLO MOVIMENTO OMNIDIRECIONAL

A maior flexibilidade e manobrabilidade apresentada pelos veículos omnidirecionais proporcionam uma maior liberdade de movimento, podendo ser definido pelo vetor velocidade linear, projetado nas respetivas componentes cartesianas, bem como pelo movimento angular, centrado sobre o centro de massa.

A estratégia de controlo da direção de movimento divide-se em dois métodos de controlo complementares, estes referentes à amplitude e orientação do vetor velocidade linear, Figura 59, e à orientação do veículo no espaço, Figura 60.

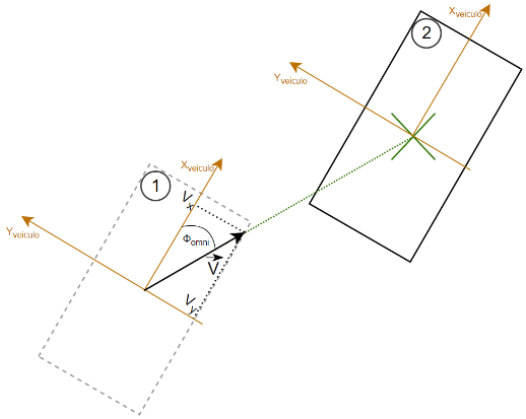


Figura 59: Ilustração do método de controlo da amplitude e orientação do vetor velocidade linear.

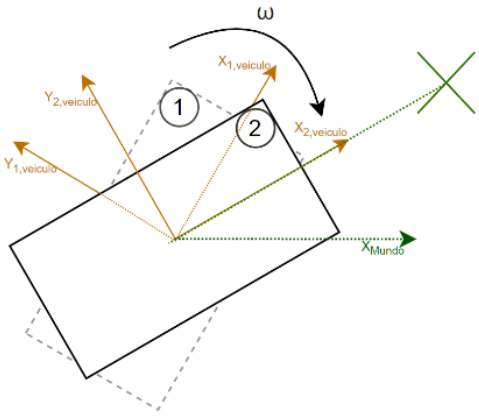


Figura 60: Ilustração do método de controlo da orientação do veículo no espaço.

6.1.1. Controlo da Direção do Vetor Velocidade Linear

O algoritmo de controlo da direção do vetor velocidade define a orientação do movimento linear do veículo omnidirecional, Φ_{omni}, em função do espaço envolvente e da orientação do alvo. Similarmente ao método de navegação autónoma não-holonómica, o comportamento define-se por um sistema dinâmico composto pelos comportamentos de convergência ao alvo, $f_{alvo,omni}$, e de fuga aos obstáculos detetados, $f_{obs\_total,omni}$. O método deve assegurar a convergência do sistema para soluções assintoticamente estáveis, independentemente da orientação de navegação, Φ_{nav}.

Tendo em consideração as características holonómicas associadas aos veículos omnidirecionais, o método deve considerar a hipótese de a orientação de movimento linear não coincidir com a orientação do veículo, isto é $\Phi_{omni} \neq \Phi_{veic}$. Esta particularidade redefine a notação de direção de navegação do veículo, Φ_{nav} ,se comparado com a navegação não-holonómica.

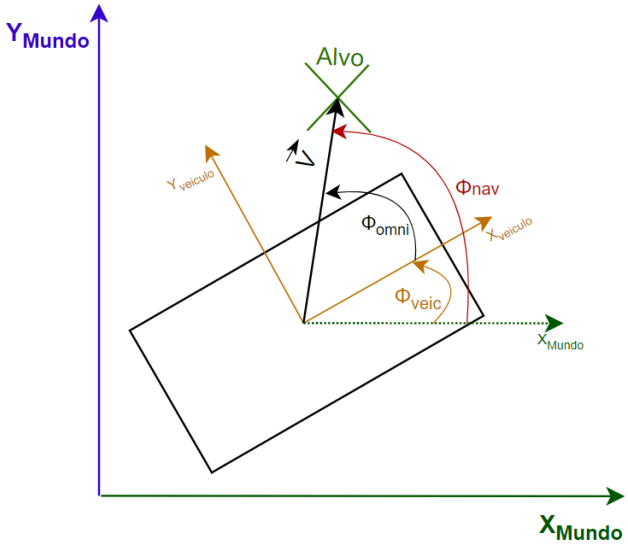


Figura 61: Definição das variáveis de controlo para o movimento holonómico. Φ_{veic} representa a orientação do veículo no espaço, Φ_{omni} a orientação do movimento linear e Φ_{nav} a orientação de navegação em relação ao espaço.

Por conseguinte, e atendendo à Figura 61, Φ_{nav} determina-se em função da orientação do veículo no espaço, sendo esta compensada pela componente Φ_{omni} , equação 38. Destaca-se a particularidade de Φ_{omni} ser determinado em relação à orientação frontal do veículo.

$$\Phi_{nav} = \Phi_{veic} + \Phi_{omni} \tag{38}$$

a) Comportamento de Seguir para o Alvo

Tendo como referência a estratégia enunciada em Bicho, (1999), o comportamento de navegar em direção ao alvo molda-se segundo uma função circular sinusoidal, estabelecendo a sua orientação como um ponto fixo atrator. Para o efeito, determina-se a orientação do alvo, Ψ_{alvo}, em relação à posição, equação 39.

$$\Psi_{alvo} = \tan^{-1}\left(\frac{Y_{alvo,i} - Y_{veic,i}}{X_{alvo,i} - X_{veic,i}}\right) \tag{39}$$

Durante a navegação, pretende-se que a orientação do alvo i seja independente da orientação do veículo, Φ_{veic}, devendo esta ser determinada em função do referencial estático associado ao espaço/"mundo", Figura 62.

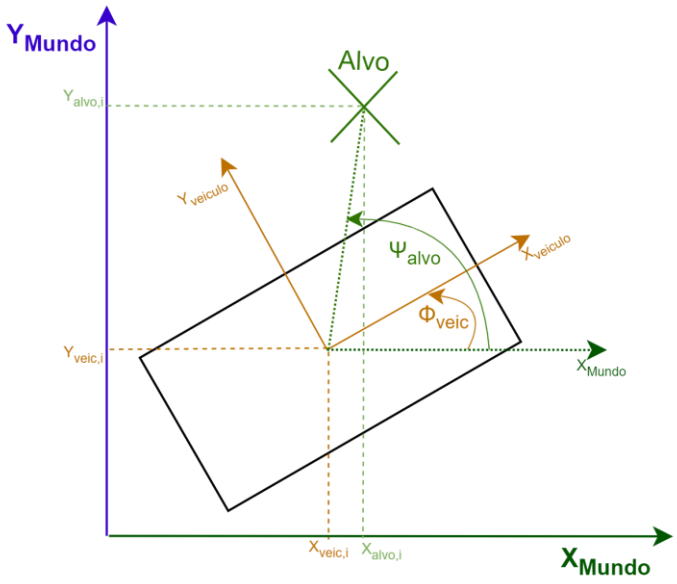


Figura 62: Orientação do alvo em relação ao espaço. O comportamento de convergir para o alvo erige um ponto fixo atrator em Ψ_{alvo}.

Atendendo às considerações enunciadas, o sistema dinâmico que modela o comportamento de convergência da orientação do movimento linear para o alvo é descrito segundo a equação 40. O sistema dinâmico introduz um ponto fixo atrator na direção do alvo, Ψ_{alvo}, com uma taxa de relaxamento de

$\lambda_{alvo_{omni}}$ (> 0). A gama de atração influencia a dinâmica em todo o domínio circular, excluindo a condição de $\Phi_{nav} = -\Psi_{alvo}$.

$$f_{alvo\_omni}(\Phi_{nav}) = -\lambda_{alvo_{omni}}.\sin(\Phi_{nav} - \Psi_{alvo}) \quad (40)$$

b) Comportamento de Evitar Colisões com Obstáculos

A componente de evitar colisões com obstáculos fundamenta-se nos conceitos enunciados em Bicho, (1999), formulando um ponto fixo repulsor coincidente com a direção do setor da dinâmica, $\Psi_{obs,i}$. Deste modo, cada setor dinâmico, descrito pela equação da dinâmica 41, introduz uma força repulsiva variável dependente da distância ao obstáculo virtual detetado. Comparativamente ao método proposto em Bicho, (1999), devido às características holonómicas dos veículos omnidirecionais, o sistema dinâmico apresentado distingue-se pela variável comportamental selecionada, Φ_{nav}, sendo as restantes componentes obtidas de forma idêntica para a condição de navegação não-holonómica.

$$f_{obs_{omni},i}(\Phi_{nav}) = \lambda_{obs_{omni},i}.\left(\Phi_{nav} - \Psi_{obs,i}\right).e^{-\frac{(\Phi_{nav}-\Psi_{obs,i})^2}{2.(\sigma_{obs,i})^2}} \quad (41)$$

Identicamente ao método de navegação não-holonómica, o comportamento de evitar colisões com obstáculos estáticos e dinâmicos define-se pelo contributo de cada setor dinâmico, traduzindo-se num somatório das forças repulsivas individuais, equação 42.

$$f_{obs\_omni_{Total}}(\Phi_{nav}) = \sum_{i=1}^{n} f_{obs_{omni},i}(\Phi_{nav}) \quad (42)$$

A título de exemplo, para a condição verificada na Figura 63, a dinâmica resultante da equação 42 introduz um ponto fixo repulsor na orientação média dos obstáculos detetados, no caso por Ψ_n e Ψ_{n-1}, divergindo a orientação de movimento linear dessa direção. Considerando unicamente a influência do comportamento de evitar obstáculos, para a condição ilustrada, o sistema tende a manter a direção de movimento, uma vez que está fora da gama de repulsão do ponto fixo repulsor, ou seja, dada as circunstâncias apresentadas, a orientação do obstáculo não introduz risco de colisão para o sistema.

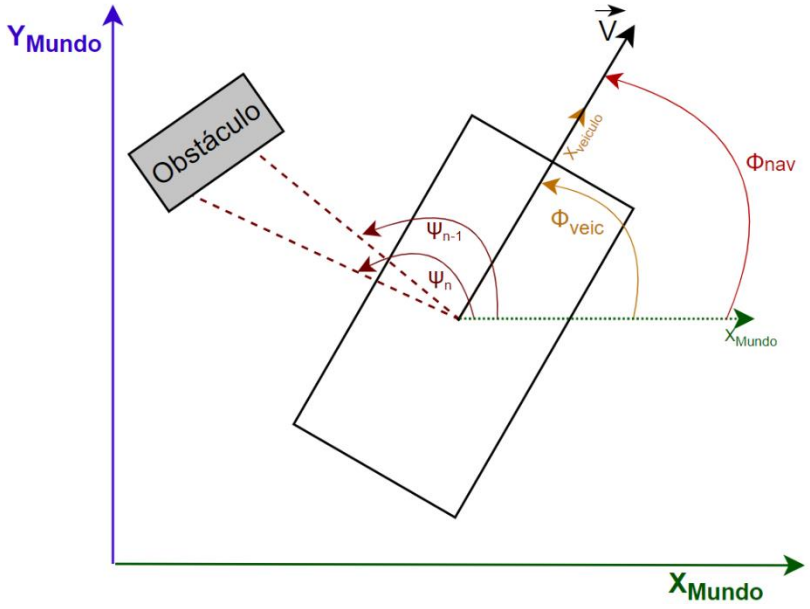


Figura 63: Representação da influência do comportamento de evitar obstáculos. A condição ilustrada considera apenas a ação do método de controlo da orientação do vetor velocidade linear, mantendo, desse modo, a orientação do veículo inalterável.

c) Integração dos Comportamentos Considerados

O controlo da direção do movimento linear define-se pela influência de ambos os comportamentos considerados. Desse modo, o sistema dinâmico que define a estratégia corresponde ao somatório da força atratora, $f_{alvo\_omni}(\Phi_{nav})$, e da força repulsora resultante, $f_{obs\_omni_{Total}}(\Phi_{nav})$.

Por forma a evitar o relaxamento do sistema em pontos fixos instáveis, no caso de a direção ser coincidente com o ponto repulsor, é adicionada à dinâmica uma força estocástica, f_{estoc}. Esta força é constituída por um ruído branco do tipo gaussiano, idêntico ao apresentado na condição de navegação não-holonómica.

$$\frac{d\Phi_{nav}}{dt} = f_{alvo\_omni}(\Phi_{nav}) + f_{obs\_Total\_omni}(\Phi_{nav}) + f_{estoc} \quad (43)$$

d) Implementação do Método na Dinâmica do Veículo

A estratégia proposta baseia-se na projeção vetorial do vetor velocidade linear nas respetivas componentes cartesianas, esta em função da variação da direção de navegação resultante da dinâmica descrita pela equação 43. Para o efeito, o algoritmo, com base na sua variação temporal, determina o

ângulo de direção de navegação, Φ_{nav}. A conversão enunciada baseia-se numa aproximação da equação diferencial 43 numa equação algébrica recursiva, admitindo que o tempo de computação seja conhecido, Δt. Para o efeito, recorre-se ao método de *Euler*, determinando a derivada de $\frac{d\Phi_{nav}}{dt}$, conforme a equação 44.

$$\frac{d\Phi_{nav}(t)}{dt} = \lim_{\Delta t \to 0} \frac{\Phi_{nav}(t + \Delta t) - \Phi_{nav}(t)}{\Delta t} \tag{44}$$

Admitindo o passo de *Euler* como o tempo de um ciclo de computação, realiza-se a aproximação à equação 44 através do método de *Euler* progressivo, equação 45. O método de *Euler* regressivo não pode ser aplicado devido a ser não causal.

$$\frac{d\Phi_{nav}(t)}{dt} \approx \frac{\Phi_{nav}(t + \Delta t) - \Phi_{nav}(t)}{\Delta t} \tag{45}$$

Rearranjando os termos, retira-se o valor $\Phi_{nav}(t + \Delta t)$ em função da orientação de navegação atual,$\Phi_{nav}(t)$, e do valor obtido da dinâmica descrita pela equação 43, conforme 46.

$$\Phi_{nav}(t + \Delta t) = \Phi_{nav}(t) + \Delta t . \frac{d\Phi_{nav}(t)}{dt} \tag{46}$$

Por último, uma vez determinado o valor de orientação de navegação, deduz-se a amplitude de orientação relativa ao movimento linear, tendo em consideração a relação descrita pela equação 38, obtendo a equação 47.

$$\Phi_{omni} = \Phi_{nav} - \Phi_{veic} \tag{47}$$

6.1.2. Controlo da Orientação do Veículo

O controlo de orientação do veículo no espaço é um dos mecanismos complementares do controlo do comportamento de navegação omnidirecional. Este mecanismo implementa um método de ajuste da orientação do veículo em função das considerações de navegação verificadas. O método assemelha-se

na estratégia de ajuste da orientação de navegação para o comportamento de navegação não-holonómica. Com o efeito, a estratégia define-se pela influência dos comportamentos de evitar colisões com obstáculos e de convergência da orientação do veículo em direção ao alvo (Figura 64).

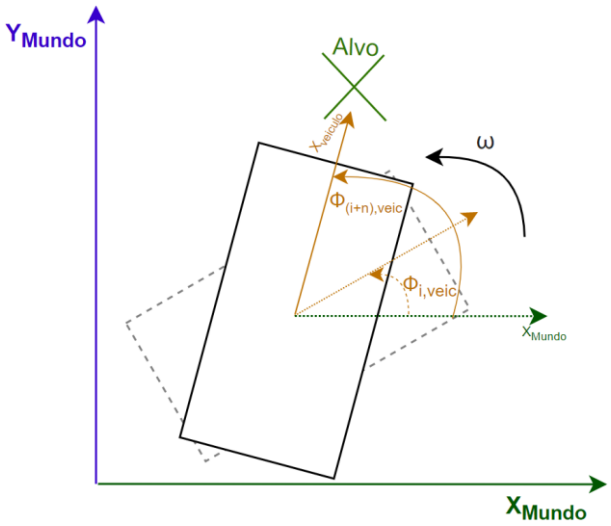


Figura 64: Ajuste da orientação do veículo, considerando apenas a influência do alvo. A condição ilustrada considera apenas a ação do método de controlo da orientação do veículo, desconsiderando a definição do vetor velocidade linear.

a) Seguir para o Alvo

Assumindo que as posições do alvo e do veículo no espaço sejam conhecidas, o campo dinâmico define-se segundo a equação da dinâmica 48, sendo Ψ_{alvo} a orientação do alvo relativamente ao veículo, Φ_{veic} a orientação do veículo e $\lambda_{alvo_{ori}}$ (> 0) a taxa de relaxamento.

$$f_{alvo_{ori}}(\Phi_{veic}) = -\lambda_{alvo_{ori}}.\sin(\Phi_{veic} - \Psi_{alvo}) \quad (48)$$

b) Comportamento de Divergir orientação do Veículo dos Obstáculos

Semelhantemente ao comportamento anterior, o método de divergir a orientação do veículo da orientação dos obstáculos segue os fundamentos da estratégia de navegação não-holonómica. O sistema dinâmico que formula a dinâmica individual, equação 49, depende da orientação de cada obstáculo detetado, $\Psi_{obs,i}$, da gama angular da força repulsora, $\sigma_{obs,i}$, e da força de repulsão, $\lambda_{obs_{i,ori}}$ (> 0).

$$f_{obs_i,ori}(\Phi_{veic}) = \lambda_{obs_{i,ori}}.\left(\Phi_{veic} - \Psi_{obs,i}\right).e^{-\frac{(\Phi_{veic}-\Psi_{obs,i})^2}{2.(\sigma_{obs,i})^2}} \quad (49)$$

A dinâmica resultante formula-se pela influência individual dos setores dinâmicos do sistema, sendo esta definida por um somatório da ação de cada setor dinâmico, conforme a equação 50.

$$f_{obs_{ori_{Total}}}(\Phi_{veic}) = \sum_{i=1}^{n} f_{obs_i,ori}(\Phi_{veic}) \quad (50)$$

c) Integração dos Comportamentos Considerados

O controlo da orientação do veículo delineia-se pela ação simultânea dos comportamentos de orientação em direção ao alvo, $f_{alvo_{ori}}(\Phi_{veic})$, e de evitar que a orientação do veículo coincida com a dos obstáculos, $f_{obs_{ori_{Total}}}(\Phi_{veic})$. Ademais, conforme os métodos de controlo dinâmico enunciados, é adicionado à dinâmica a ação de uma força estocástica, resultando na equação 51.

$$\frac{d\Phi_{veic}}{dt} = f_{alvo_{ori}}(\Phi_{veic}) + f_{obs_{ori_{Total}}}(\Phi_{veic}) + f_{estoc} \quad (51)$$

d) Implementação do Método na Dinâmica do Veículo

Dado o controlo da orientação do veículo ser realizado pela manipulação da variável ω_{veic}, atributo da cinemática inversa do veículo, a ação de controlo gerada do campo dinâmico 51 é diretamente utilizada como variável de atuação na dinâmica do veículo. Esta característica é verificável pela associação descrita pela equação 52, atribuindo a variação temporal da orientação do veículo à velocidade angular do mesmo.

$$\omega_{veic} = \frac{d\Phi_{veic}}{dt} = f_{alvo_{ori}}(\Phi_{veic}) + f_{obs_{ori_{Total}}}(\Phi_{veic}) + f_{estoc} \quad (52)$$

O enfoque pela natureza holonómica traduz-se na prevalência da atuação do método de ajuste de amplitude de orientação do movimento linear, Φ_{omni} , sobre o método da dinâmica de orientação do

veículo no espaço, Φ_{veic}. Ao nível de parametrização dos termos de controlo, o predomínio dos métodos impõe que o sistema seja mais reativo para as variações da dinâmica relativa ao controlo de Φ_{omni}, estabelecendo a seguinte hierarquia:

$$\lambda_{alvo_{omni}} > \lambda_{alvo_{ori}} \quad \left| \quad \lambda_{obs_{omni},i} > \lambda_{obs_{i,ori}} \right| \quad \lambda_{obs_{omni},i} > \lambda_{alvo_{omni}} \quad \left| \quad \lambda_{obs_i,ori} > \lambda_{alvo_{ori}} \right.$$

6.1.3. Controlo Módulo da Velocidade Linear

O módulo da velocidade linear é controlado por um sistema dinâmico linear, idêntico ao enunciado na dinâmica de controlo da velocidade para a navegação não-holonómica. O sistema dinâmico resultante é formulado pelos comportamentos de se dirigir para o alvo e de desvio de obstáculos. No entanto, contrariamente aos métodos de controlo da variação angular, a dinâmica deve ser definida pela ação de apenas um dos comportamentos, selecionando o comportamento de seguir para o alvo caso não sejam detetados obstáculos na proximidade do veículo. Caso contrário, o método rege-se pelo comportamento de evitar obstáculos, adaptando a velocidade linear em função da distância ao obstáculo mais próximo detetado.

a) Comportamento de Seguir para o Alvo

Na condição de navegação sob a influência do comportamento de seguir para o alvo, o sistema dinâmico linear erige um ponto fixo atrator coincidente com a velocidade máxima admitida para o traçado de navegação, estabelecendo $V_{alvo,omni} = Vel_{max,i}$, sendo $Vel_{max,i}$ a velocidade máxima permitida no segmento de rota estabelecido pelo alvo i. Caso o alvo seja o último da rota, a velocidade linear deve diminuir à medida que o veículo se aproxima do mesmo. Esta informação é obtida da mensagem recebida do *Service Manager* aquando da receção do serviço. Como verificável na Figura 65, o sistema determina as distâncias aos obstáculos detetados pelos setores da dinâmica, contudo, no cálculo, é desconsiderado a distância do corpo do próprio veículo. Na condição retratada, dada a validação das distâncias de segurança, a velocidade de navegação converge para $V_{alvo,omni}$.

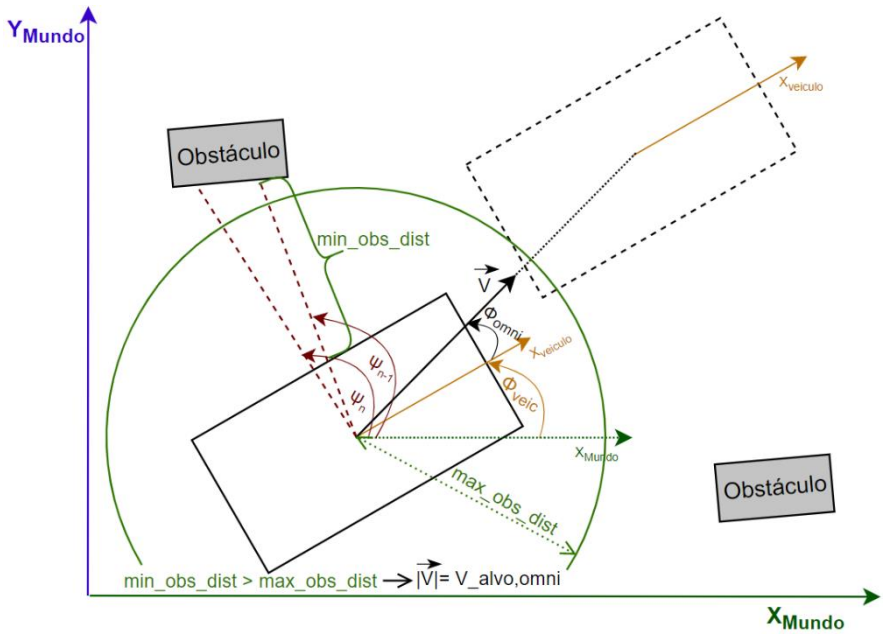


Figura 65: Dinâmica linear definida pela ação do comportamento de convergência para o alvo, convergindo o módulo da velocidade linear para $V_{alvo,omni}$.

Considerando apenas a dinâmica de seguir um alvo, a dinâmica linear define-se pela expressão descrita na equação 53, convergindo a velocidade linear para $V_{alvo,omni}$ com uma taxa de relaxamento definida de $c_{alvo,omni}$ (> 0).

$$\frac{dv_{linear,omni}}{dt} = -c_{alvo,omni}.\left(v_{linear,omni} - V_{alvo,omni}\right) \tag{53}$$

b) Comportamento de Evitar Obstáculos

Por sua vez, o comportamento de evitar obstáculo implementa o controlo de velocidade em função da menor distância detetada pelos setores da dinâmica, $min_{obs,dist}$. Este comportamento permite que o sistema consiga corrigir a *pose*, num espaço temporal útil, por forma a evitar colisões com os obstáculos detetados. A velocidade linear desejada na presença de obstáculos, $V_{obs,omni}$, é determinada por uma função do tipo cúbica dependente da distância mínima detetada, $min_{obs,dist}$, do tipo descrito pela equação 54.

$$V_{obs,omni} = a.\left(min_{obs,dist}\right)^3 + b.\left(min_{obs,dist}\right)^2 + c.\left(min_{obs,dist}\right) + d \tag{54}$$

A taxa de variação do módulo da velocidade linear é ajustada pelos termos que compõem a função 54, especificando três zonas de variação possíveis, em função da distância ao obstáculo mais próximo (Figura 66). A ação da dinâmica compreende-se entre a distância máxima que caracteriza o comportamento de evitar obstáculos, $max_{obs,dist}$, e a distância mínima de paragem do movimento linear. Por motivos de segurança, o valor da distância mínima de paragem estabelece-se como superior ao imposto no mecanismo de paragem de segurança do veículo. Destaca-se que o mecanismo de paragem de segurança anula todas as componentes de movimento do veículo omnidirecional, isto é, os movimentos lineares e angulares.

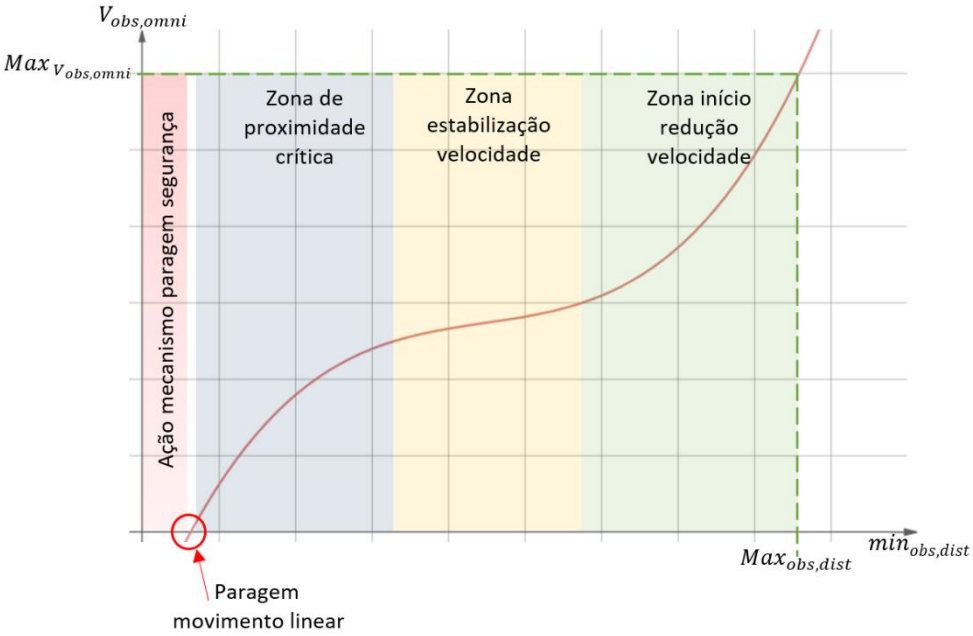


Figura 66: Representação gráfica da variação do módulo da velocidade linear em função da menor distância aos obstáculos detetados. Os termos devem ser ajustados de modo a garantir a estabilidade do sistema.

Considerando que o método de controlo de velocidade linear é governado pela ação do comportamento de evitar obstáculos, o sistema dinâmico que define a dinâmica linear simplifica-se de acordo com a equação 55. A dinâmica de controlo converge a velocidade linear do veículo, $v_{linear,omni}$, para o valor desejado, $V_{obs,omni}$, com uma taxa de relaxamento de $c_{obs,omni}$ (> 0).

$$\frac{dv_{linear,omni}}{dt} = -c_{obs,omni}.\left(v_{linear,omni} - V_{obs,omni}\right) \tag{55}$$

A Figura 67 exemplifica uma possível condição de navegação no qual é verificado a presença de um obstáculo dentro da gama de atuação do comportamento de evitar obstáculos. Para o efeito, o sistema ajusta a velocidade linear em função do menor valor determinado pelos setores dinâmicos, $min_{obs,dist}$, de acordo com a equação 54.

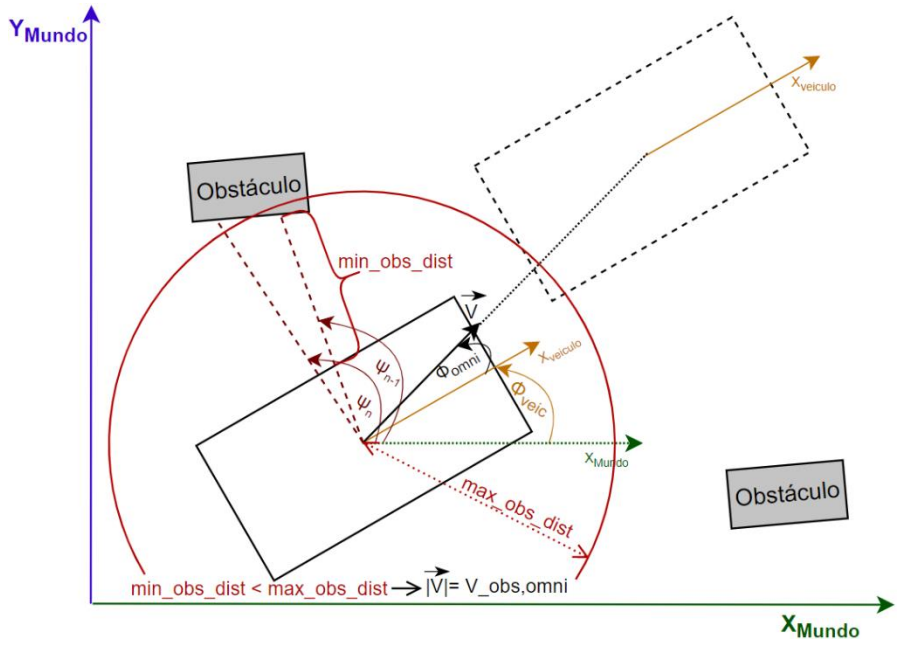


Figura 67: Dinâmica linear definida pela ação do comportamento de evitar colisões, convergindo o módulo da velocidade linear para $V_{obs,omni}$.

c) Mecanismo de Seleção de Comportamentos

A dinâmica resultante é formulada pela ação de um dos comportamentos indicados, no mesmo espaço temporal. Para o efeito, na dinâmica linear é integrado um mecanismo de seleção de comportamentos, gerido pelos termos $c_{alvo,omni}$ e $c_{obs,omni}$. Os termos, além de assimilar a estratégia de seleção do comportamento desejado, também definem a taxa de relaxamento do sistema dinâmico. De modo a garantir que o sistema consiga convergir para as soluções assintoticamente estáveis, a seguinte hierarquia deve ser garantida, sendo um complemento à apresentada no capítulo Controlo da Orientação do Veículo:

$$c_{v,obs,omni} > c_{v,alvo,omni} \quad \left| \quad c_{v,obs,omni} > \lambda_{obs_{omni},i} \quad \right| \quad c_{v,alvo,omni} > \lambda_{alvo_{omni}}$$

A seleção dos comportamentos obedece ao seguinte critério:

Obstáculos fora da gama de atuação do comportamento evitar obstáculos:	$min_{obs,dist} > max_{alcance,obs}$	$c_{alvo,omni} = c_{v,alvo,omni}$ $c_{obs,omni} = 0$
Obstáculos dentro da gama de atuação comportamento evitar obstáculos:	$min_{obs,dist} < max_{alcance,obs}$	$c_{alvo,omni} = 0$ $c_{obs,omni} = c_{v,obs,omni}$

Considerando a integração do mecanismo de seleção de comportamentos, a dinâmica de controlo resultante define-se conforme a equação 56.

$$\frac{dv_{linear,omni}}{dt} = -c_{alvo,omni}.\left(v_{linear,omni} - V_{alvo,omni}\right) - c_{obs,omni}.\left(v_{linear,omni} - V_{obs,omni}\right) \quad (56)$$

Na estratégia adotada, o controlo do movimento do veículo é definido pelos valores instantâneos das velocidades lineares, projetados no plano vetorial bidimensional, e angulares desejados. Dado que a dinâmica de controlo considerada estipula o módulo da variação da velocidade linear, é necessário converter os dados em valores instantâneos de velocidade. O procedimento de conversão é similar ao descrito pela equação 45, diferindo a variável de controlo, conforme a equação 57.

$$\frac{dv_{linear,omni}(t)}{dt} \approx \frac{v_{linear,omni}(t + \Delta t) - v_{linear,omni}(t)}{\Delta t} \quad (57)$$

6.1.4. Formulação do Vetor Velocidade Linear nas Componentes Cartesianas

A integração dos dados obtidos das dinâmicas de controlo 56 e 43 pode ser condensado num vetor de velocidade, projetado num plano bidimensional, conforme representado na Figura 68. O vetor caracteriza-se pelo módulo da velocidade linear desejada e pela amplitude angular em relação à orientação frontal do veículo, informações obtidas das equações 57 e 47, respetivamente.

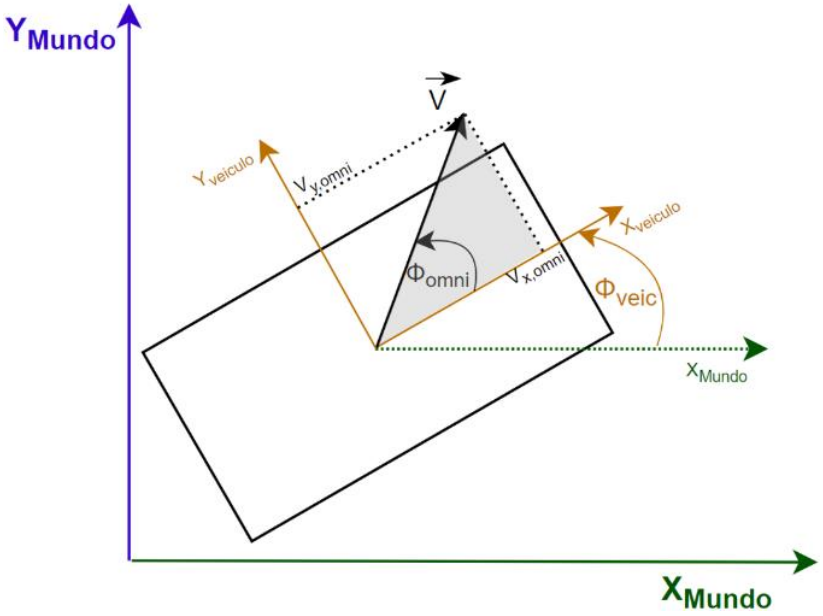


Figura 68: Projeção do vetor velocidade linear num plano bidimensional, em relação ao referencia móvel associado ao veículo.

As componentes cartesianas do vetor, $v_{x,omni}$ e $v_{y,omni}$, podem ser obtidas pela conversão retangular, conforme demonstrado nas equações 58 e 59, respetivamente.

$$v_{x,omni} = v_{omni}.\cos(\Phi_{omni}) \quad (58)$$

$$v_{y,omni} = v_{omni}.\sin(\Phi_{omni}) \quad (59)$$

7. MANOBRAS DE ACOSTAGEM

O controlo que define as operações de aproximação e afastamento aos locais de acostagem deve capacitar a plataforma móvel de executar manobras que possibilitem a sua convergência para uma dada posição e orientação no espaço. O controlador, para ambos os casos, recorre às capacidades holonómicas do veículo possibilitando a operação em espaços exíguos e complexos. O método de controlo considera ainda a presença de obstáculos na definição do movimento, podendo estabelecer pequenas correções na manobra, no caso de se demonstrar pertinente. Devido à menor tolerância de aproximação, a margem de segurança é inferior à considerada nos métodos de navegação e, como tal, a presença de operadores humanos nestes espaços deve ser evitada.

7.1. CONTROLO E DEFINIÇÃO DAS MANOBRAS DE ACOSTAGEM

O método de controlo dos movimentos de acostagem encontra-se implementado por um controlador linear, definido em função do erro da distância ao local desejado e do erro de orientação final. Apesar do sistema definir quatro tipos de manobras de acostagem, isto é, de aproximação aos locais de trabalho e *park* e de afastamento dos mesmos, os algoritmos de controlo são similares entre si. Deste modo, a análise fundamental do método equipara-se entre os quatro comportamentos.

A Figura 69 ilustra uma possível condição de aproximação a um local de acostagem. Para a situação enunciada, o sistema necessita de corrigir não só a orientação de navegação, como também de se movimentar para a posição desejada de acostagem. As posições e orientações consideradas são definidas em relação ao referencial externo (do "mundo").

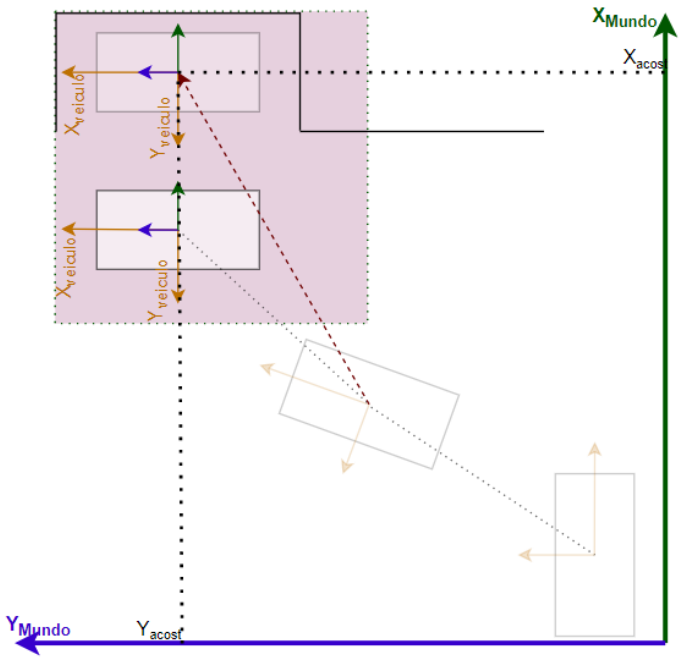


Figura 69: Manobra de aproximação a um local de acostagem. Como ilustrado, por segurança, a manobra é realizada na última fase de aproximação, destacada com cor encarnada.

7.1.1. Controlo da Orientação de Acostagem

A estratégia de correção da orientação da plataforma móvel no espaço é estabelecida por um controlador do tipo linear, em função do erro de orientação $\Phi_{acost,erro}$. Este erro é determinado em função da orientação de acostagem desejada, $\Phi_{acost,des}$, e da orientação do veículo, Φ_{veic}, equação 60. A orientação de acostagem desejada está contida na estrutura de mensagem do pedido do *task manager*, correspondente à variável *action_orientation*.

$$\Phi_{acost,erro} = \Phi_{acost,des} - \Phi_{veic} \tag{60}$$

A regra de controlo proporcional adotada determina a ação de controlo, no caso correspondente a ω_{veic}, com base no erro da variável a controlar, $\Phi_{acost,erro}$, impondo um determinado ganho, $K_{p,w}$, conforme a equação 61.

$$\omega_{veic} = K_{p,w}.\Phi_{acost,erro} \tag{61}$$

7.1.2. Controlo da Posição de Acostagem

A estratégia que define o controlo do movimento do veículo é estabelecida a dois níveis: no cálculo da direção do movimento e no controlo do módulo da velocidade linear, este em função da distância ao local desejado para a acostagem. O método é ainda complementado por um mecanismo de segurança que, em função do movimento gerado pelo controlador e dos obstáculos detetados, pode realizar ajustes às ações de controlo geradas por forma a evitar colisões com possíveis obstáculos.

a) Cálculo da Direção do Vetor Velocidade Linear

Dado o método de controlo do movimento linear se basear na projeção do vetor velocidade no plano cartesiano, é necessário determinar a orientação associada ao movimento. O cálculo é estabelecido com base na posição cartesiana da plataforma móvel no ambiente de navegação, (x_{veic}, y_{veic}), e na posição de acostagem desejada, $(x_{acost,des}, y_{acost,des})$, esta última contida na variável *targetPose*, pertencente à estrutura da mensagem do pedido da execução da tarefa de acostagem, por parte do *task manager*, conforme a equação 62. Dada a possibilidade de que a orientação de navegação não seja coincidente com a do veículo, é retirado o termo correspondente à orientação da plataforma no espaço, Φ_{veic}.

$$\theta_{acost} = \tan^{-1}\left(\frac{y_{acost,des} - y_{veic}}{x_{acost,des} - x_{veic}}\right) - \Phi_{veic} \tag{62}$$

b) Controlo Módulo da Velocidade Linear

A estratégia de controlo do módulo da velocidade linear fundamenta-se no erro de posicionamento, $dist_{acost,erro}$, este determinado entre a posição final de acostagem desejada, $(x_{acost,des}, y_{acost,des})$, e a posição do veículo no espaço, (x_{veic}, y_{veic}), conforme a equação 63.

$$dist_{acost,erro} = \sqrt{(x_{acost,des} - x_{veic})^2 - (y_{acost,des} - y_{veic})^2} \tag{63}$$

Em virtude da natureza intrínseca dos tipos de manobras, foi necessário adaptar a regra de controlo conforme as necessidades da manobra, distinguindo-se entre as manobras de aproximação e de afastamento.

i. Manobras de Aproximação aos Locais de Acostagem

Na condição de aproximação, é desejado que o módulo da velocidade linear diminua gradualmente em função da proximidade ao local de acostagem. Para o efeito, o método de controlo adotado fundamenta-se na regra de controlo proporcional linear, em função da distância ao local de acostagem, $dist_{acost,erro}$, impondo um ganho $K_{p,aprox}$, equação 64. Apesar dos comportamentos de aproximação e afastamento aos locais de acostagem serem projetados para atuação próxima aos locais, a ação de controlo deve ser limitada, especialmente na condição de $dist_{acost,erro}$ ter um valor inicial elevado.

$$v_{acost} = K_{p,aprox}.dist_{acost,erro} \tag{64}$$

ii. Manobras de Afastamento aos Locais de Acostagem

Por sua vez, na condição de afastamento aos locais de acostagem, o veículo deve aumentar gradualmente o módulo da velocidade linear, tendo valor máximo no caso de o erro ser próximo de nulo, no caso, $dist_{acost,erro} \approx 0$. Dada as considerações, a regra de controlo adotada baseia-se numa função do tipo quadrática, como descrito pela equação 65. A parábola, definida pela função, deve apresentar concavidade voltada para baixo, implicando que o coeficiente tenha valor negativo, $K_{p,afast} < 0$. Além do mais, deve-se garantir que o valor máximo da função ocorra na condição de erro nulo, traduzindo-se em $v_{acost} \approx V_{max}$ quando $dist_{acost,erro} \approx 0$. Para o efeito, o coeficiente define-se segundo a equação 66, relacionando a velocidade máxima considerada, V_{max}, com o a distância inicial ao local de acostagem, $dist_{acost,inicial}$.

$$v_{acost} = K_{p,afast} * \left(dist_{acost,erro}\right)^2 + V_{max} \tag{65}$$

$$K_{p,afast} = -\frac{V_{max}}{\left(dist_{acost,inicial}\right)^2} \tag{66}$$

c) Formulação do Vetor Velocidade Linear nas Componentes Cartesianas

Como referido, o controlo da posição de acostagem é formulado por um vetor velocidade, projetado nas respetivas componentes cartesianas, considerando, para o efeito, a direção do movimento linear, θ_{acost}, e o módulo da velocidade linear desejada, v_{acost}. Deste modo, obtém-se o sistema de equações 67 e 68, referentes à velocidade no eixo das abscissas e ordenadas, respetivamente.

$$v_{x,acost} = v_{acost} * \cos(\theta_{acost}) \quad (67)$$

$$v_{y,acost} = v_{acost} * \sin(\theta_{acost}) \quad (68)$$

7.1.3. Mecanismos de Segurança

As manobras de acostagem são consideradas processos críticos e delicados, isto devido à possível operação em espaços diminutos e à ilegibilidade dos movimentos apresentados durante o estabelecimento das manobras. Consequentemente, e de modo a implementar um maior grau de segurança, foi estabelecido uma margem de segurança variável, em função da distância ao local de acostagem. Esta deve diminuir em função da proximidade ao local final de paragem. Para tal, nos comportamentos de aproximação, a margem de segurança deve diminuir em função da redução do erro da distância, $dist_{acost,erro}$, equação 69. Paralelamente, no que se refere aos comportamentos de afastamento, a margem de segurança deve aumentar em função da redução de $dist_{acost,erro}$, equação 70. Deste modo, retarda-se a atuação dos mecanismos de segurança, em função da proximidade ao local de paragem de acostagem. A constante c possibilita o ajuste da margem de segurança a considerar, em função da distância de acostagem.

$$margem_{dist} = c * dist_{acost,erro} \quad (69)$$

$$margem_{dist} = c * \frac{1}{dist_{acost,erro}}, dist_{acost,erro} > 0 \quad (70)$$

O método de acostagem encontra-se complementado por um mecanismo de segurança que permite o ajuste do movimento definido pelo controlador proporcional, este dependente da margem de segurança. O mecanismo encontra-se capacitado de manipular as ações de controlo geradas pelo controlador linear no caso de não ser verificada a distância de segurança aos obstáculos. O algoritmo, numa primeira fase, verifica se os obstáculos circundantes ao veículo estão a uma distância superior à margem de segurança. Nesta iteração considera-se a distância linear mínima detetada em cada quadrante da plataforma móvel. No caso de se verificar a condição de segurança nos quatro quadrantes da plataforma móvel, o mecanismo não realiza ajustes às ações de controlo geradas. Caso contrário, o método determina as regiões do veículo que incumprem a margem de segurança. Nesta condição, o ajuste apenas é efetuado se o vetor velocidade estiver definido por componentes cartesianas coincidentes

à região ou regiões determinadas. Ademais, o método verifica se os obstáculos circundantes estejam a uma distância superior ao valor mínimo considerado, este constante. Se a condição não se verificar, o mecanismo de segurança imobiliza o veículo até que a margem mínima de segurança seja validada. A Figura 70 expõe o algoritmo de correção do movimento, anteriormente enunciado, na forma de fluxograma simplificado.

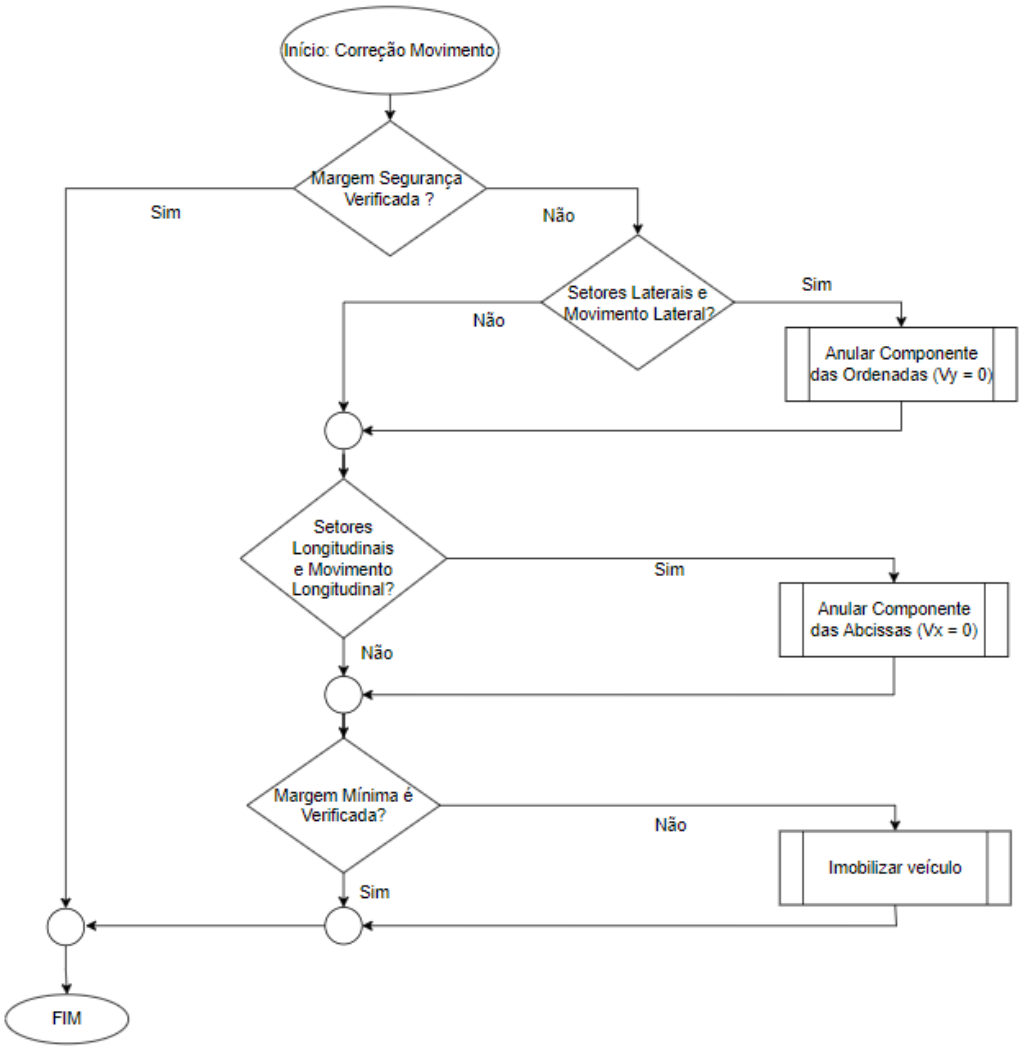


Figura 70: Fluxograma simplificado do algoritmo de correção do movimento de aproximação e afastamento aos locais de acostagem.

7.2. MANOBRAS DE APROXIMAÇÃO AOS LOCAIS DE *PARK*

Embora o algoritmo de aproximação ao local de acostagem seja similar para a condição de *dock* e *park*, nesta última deve-se ter em consideração as especificações de aproximação ao ponto de carregamento do veículo. Como enunciado em KUKA AG, (2021), a aproximação ao local de carregamento deve ser

realizada de tal forma que se garanta o contacto entre os conectores de carregamento do veículo e o ponto de carga no solo. Dessa forma, o estudo da aproximação ao local de *park* deve considerar a posição dos contactos de carregamento no solo bem como o *offset* entre a posição final de acostagem, correspondente ao centro de massa do veículo, e os conectores de carga presentes no próprio veículo. A orientação do veículo deve corresponder à orientação dos contactos de solo. Na estratégia adotada, a orientação dos eixos do veículo e do local de *park* devem ser coincidentes.

A Figura 71 ilustra a *pose* final a considerar para o processo de *park*, destacando as distâncias internas do veículo, a dimensão dos contactos de carregamento de solo e o *offset* a considerar. Considerando as especificações presentes em KUKA AG (2021), supõe-se um desfasamento de aproximadamente $1{,}50\ cm$ entre os limites estruturais do veículo e do local de carga. Como tal, e tendo em consideração as dimensões enunciadas, obtém-se um *offset* de $0{,}40\ m$. Este valor corresponde à diferença entre metade das dimensões internas do veículo e do ponto de carga, adicionando o desfasamento previsto, ($\frac{1{,}08\ m}{2} - \frac{0{,}31\ m}{2} + 0{,}015\ m \approx 0{,}40\ m$).

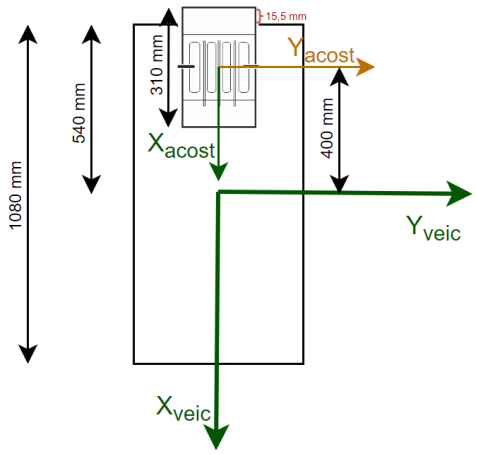


Figura 71: Representação do *offset* a considerar para a aproximação a um local de carga, em relação ao referencial do veículo. Considerando os referenciais ilustrados, a orientação do veículo deve ser coincidente com a especificada ao local de *park*.

A Figura 72 ilustra uma possível condição de aproximação ao local de *park*. Como é possível verificar, a estratégia de acostagem baseia-se no erro de posição, representado por $Dist_{erro}$, e no erro de orientação, Φ_{erro}. O processo de acostagem tem a particularidade de executar a aproximação final segundo um movimento paralelo e em sentido contrário ao eixo das abscissas, considerando o referencial presente na Figura 72. Este tipo de aproximação encontra-se especificado como requisito em KUKA AG (2021) no caso de aproximação ao ponto de carregamento.

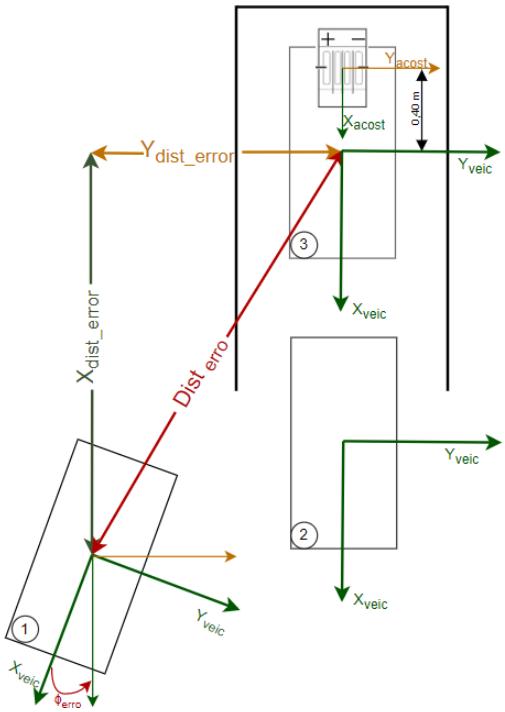


Figura 72: Ilustração do processo de aproximação a um local de carga. A manobra deve privilegiar, numa primeira fase, a correção da orientação do veículo.

8. IMPLEMENTAÇÃO

Neste capítulo é descrito a implementação e integração dos módulos que definem a arquitetura do sistema, evidenciando o fluxo de execução dos algoritmos bem como os esquemáticos de comunicação entre módulos. Ademais, dada a utilização do simulador como recurso de validação e verificação de comportamentos, é ainda descrito a composição do cenário de simulação considerado, assim como a estrutura de comunicação que permite a troca de dados entre a interface de simulação e o módulo controlador.

8.1. SIMULADOR

O projeto de investigação pretende conceber uma solução inteligente de controlo de pelo menos um manipulador móvel em ambiente industrial dinâmico, capacitando-o de realizar tarefas diversificadas, como a tarefa de transporte ou de manipulação de objetos. Para efeito de testes e validação de comportamentos, foi necessário desenvolver um cenário de simulação dinâmico, similar a um ambiente industrial, composto por zonas de trabalho e por locais de *park* ou estacionamento. O cenário de simulação contém ainda uma interface ROS, possibilitando a comunicação entre o módulo controlador e o simulador *CoppeliaSim*.

8.1.1. CENÁRIO DE SIMULAÇÃO

O cenário de simulação foi elaborado tendo por base não só as características típicas de um ambiente industrial complexo, como também os requisitos e especificações do projeto. Deste modo, o ambiente de simulação projetado é constituído por quatro zonas de trabalho, duas das quais relativas às operações de apoio a uma máquina ferramenta, por dois locais de *park* ou pontos de carga das baterias e por zonas navegáveis, delimitadas por armazéns e por zonas de trabalho. A Figura 73 apresenta o cenário de simulação utilizado para a validação da solução proposta.

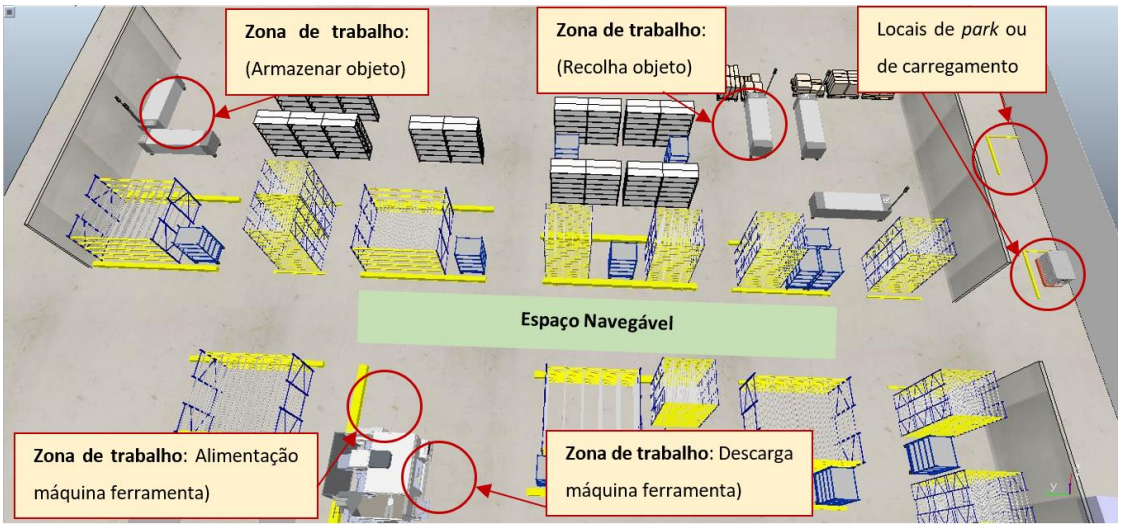


Figura 73: Cenário de simulação utilizado para validação da solução proposta.

A zona de navegação, definida como espaço navegável, apresenta desafios à operação da plataforma omnidirecional no ambiente descrito, sendo esta constituída por um corredor principal, de largura de 3,0 m, e por vias de acesso às zonas de trabalho, podendo variar entre 1,0 m e 1,8 m de largura. De modo a viabilizar a operação da plataforma móvel nas vias de acesso mais exíguas, o método de controlo dos movimentos, para estas condições, deve recorrer à maior manobrabilidade da plataforma omnidirecional. Esta estratégia, dada a reduzida margem de manobra, revela ser essencial para agilizar a operação no ambiente descrito. A Figura 74 apresenta as dimensões dos principais pontos de acesso aos locais de trabalho definidos.

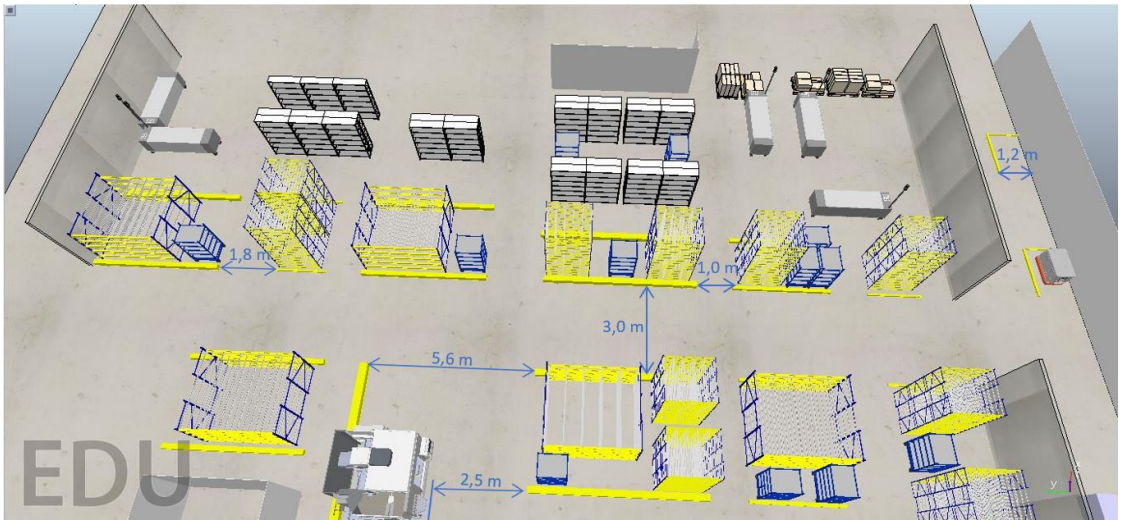


Figura 74: Dimensões dos principais pontos de acesso às zonas de trabalho e de *park*.

Em *CoppeliaSim*, os cenários de simulação são constituídos por objetos de cenário ou por modelos de simulação, sendo estes últimos constituídos por uma seleção de objetos de cenário, estruturados segundo uma hierarquia. Como tal, devido à melhor reprodução de um ambiente típico industrial, o cenário é constituído essencialmente por modelos de simulação, inclusive os elementos decorativos. De modo geral, a nível estrutural, o cenário define-se pelo modelo dinâmico do manipulador móvel, pelo modelo da máquina ferramenta, das mesas de apoio às zonas de trabalho e pelos elementos decorativos constituintes do ambiente. Estes constituintes são descritos individualmente, com maior rigor.

a) Manipulador Móvel

O manipulador móvel é constituído por um braço robótico, o LBR iiwa 14 R820, e por uma plataforma móvel omnidirecional, o KMP 200 *OmniMove*. Cada constituinte corresponde a um modelo de simulação distinto. Dado os objetivos desta dissertação, dá-se especial relevância à formulação do modelo dinâmico correspondente à plataforma móvel. Como representado na Figura 75, este é formulado por quatro objetos do tipo *joint*, relativos à composição das rodas mecânicas, por dois modelos do tipo *dummy*, associados à representação e modelação dos sensores *lidar*, e por uma *shape*, não dinâmica, referente ao modelo *CAD* da plataforma. Ademais, por forma a obter a informação da *pose* do veículo, o modelo foi complementado com dois objetos do tipo *dummy*, aferindo a função de referencial móvel.

O controlo e tratamento das informações recolhidas do cenário é realizado na *child script* associada ao modelo de simulação do manipulador móvel, permitindo ainda a comunicação com o controlador externo, via ROS *topics*. Destaca-se que a interface ROS encontra-se implementada numa *customization script* dedicada, *RosInterfaceHelp*.

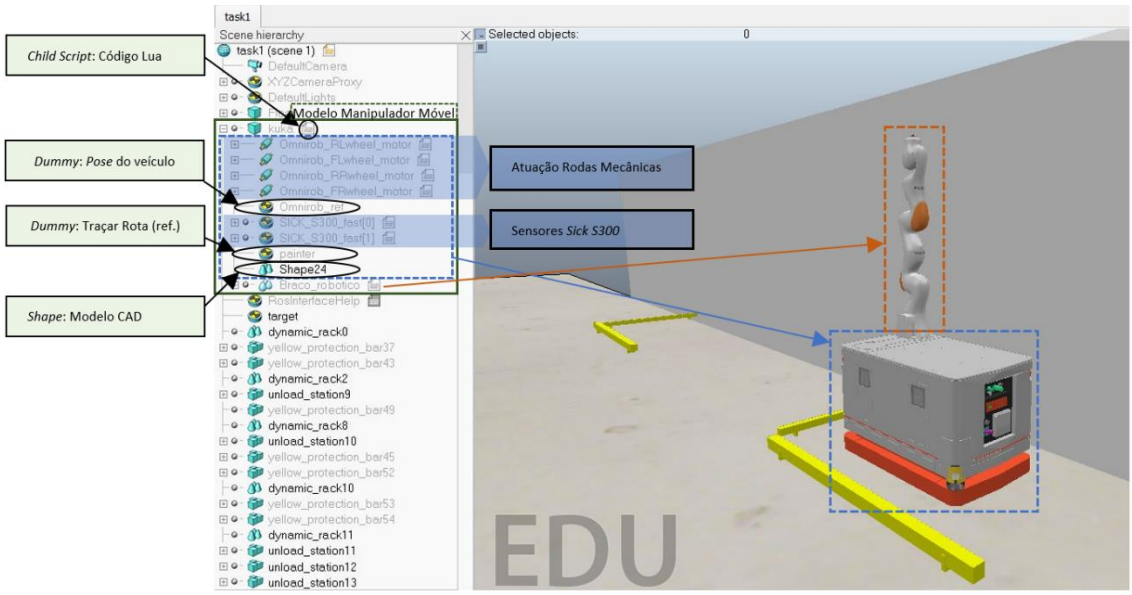


Figura 75: Estruturação modelo dinâmico do manipulador móvel.

b) Máquina Ferramenta

Em virtude da validação das tarefas que permitem realizar as operações de *machine tending*, o cenário de simulação considerado é composto por uma máquina ferramenta, no caso uma máquina fresadora. A máquina caracteriza-se por dois pontos de manipulação, a zona de *input* ou de alimentação e a zona de *output* ou de descarga da peça manipulada.

O modelo de simulação é formulado por três objetos do tipo *shape*, anexados à *shape* que contém a informação do modelo CAD da máquina, o *MillingMachine*, na Figura 76. O manipulador móvel tem como objetivo alimentar a máquina fresadora na zona de *input*, apoiando o produto a ser manipulado no suporte da máquina, identificado como *Load*. Semelhantemente, na operação de recolha, o manipulador móvel deve retirar o produto já manipulado do suporte de descarga, este identificado como *Unload*. Dado que as operações de maquinagem não serem objeto de estudo neste projeto de investigação, em simulação, estes processos encontram-se subentendidos.

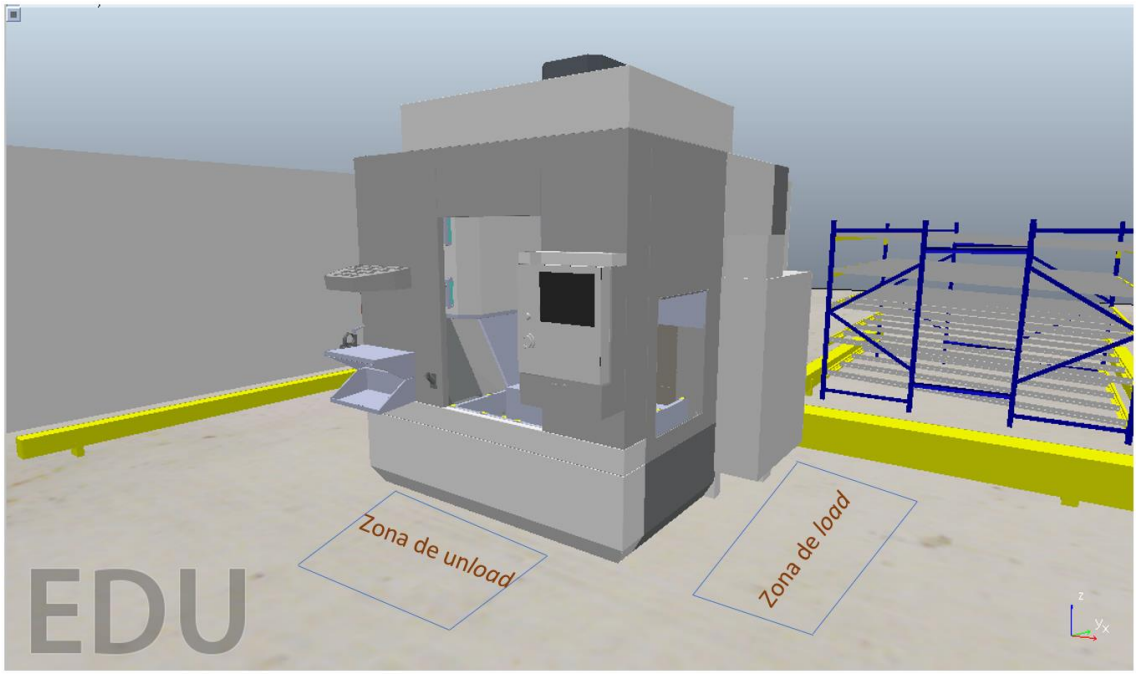


Figura 76: Ilustração da máquina ferramenta em ambiente de simulação. Destaque para as zonas de paragem do manipulador móvel por forma a realizar as operações de *machine tending*.

c) Zonas de Trabalho

No cenário de simulação projetado, as zonas de trabalho são delimitadas por elementos típicos de um ambiente industrial, como armazéns intermédios, paletes de suporte ou barras de segurança. Ao nível da hierarquia de cenário, estes são formulados por modelos de simulação independentes detetáveis e responsivos a colisões. Ademais, conforme a complexidade da sua representação, o modelo pode ser constituído por outros objetos, como o caso das paletes de suporte e a carga associada. No ambiente de simulação desenvolvido, foi considerado duas zonas de trabalho, ambas caracterizadas por modelos de mesas de apoio, também responsivos a colisões e detetáveis pelos sensores *lidar* (ver exemplo de zona de trabalho na Figura 77). Nas zonas de trabalho, o manipulador móvel tem como objetivo realizar as operações de carga e de descarga dos produtos a serem manipulados ou já manipulados, respetivamente.

Figura 77: Ilustração da estação de trabalho correspondente ao ponto de recolha do objeto a manipular. Destaque da zona de paragem do manipulador móvel.

8.1.2. COMUNICAÇÃO

A comunicação entre o módulo controlador e a interface de simulação é estabelecida via ROS *topic*. Para o efeito, e atendendo os requisitos do controlador, o simulador publica, de forma continua, as informações da *pose* do veículo no ambiente (tópico */pose_vehicle_world*), as componentes que definem a velocidade linear e angular do veículo (tópico */vehicle_velocity*) e os dados dos sensores *lidar* (tópicos */lidar_front_data* e */lidar_rear_data*), replicando a receção de dados de uma aplicação real. Por sua vez, o controlador publica as variáveis de controlo, isto é, as componentes que definem a velocidade desejada (tópico */nav_speed*) e os parâmetros de controlo do simulador (tópicos */startSimulation*, */pauseSimulation* e */stopSimulation*), permitindo, por exemplo, iniciar a simulação remotamente. A Figura 78 e a Figura 79 ilustram os tópicos e os tipos de mensagem associados à comunicação.

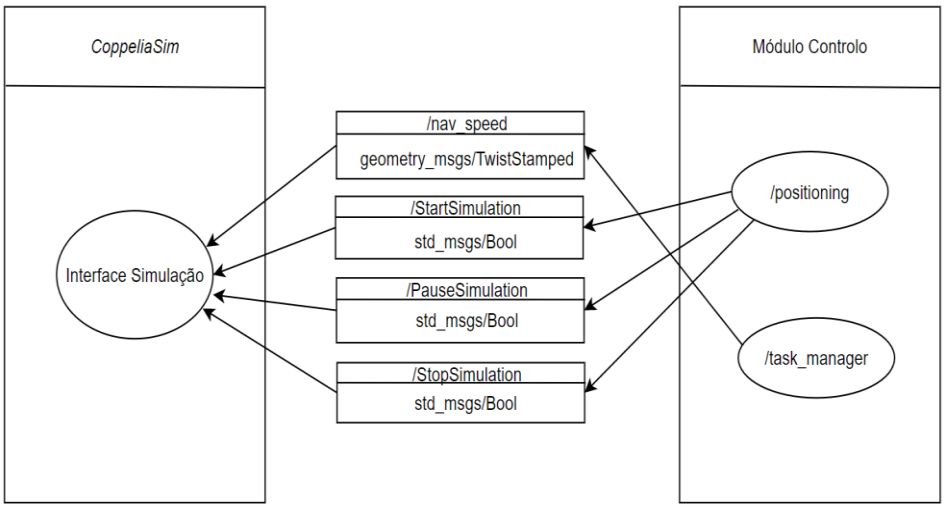


Figura 78: Comunicação ROS *topic* entre controlador e simulador.

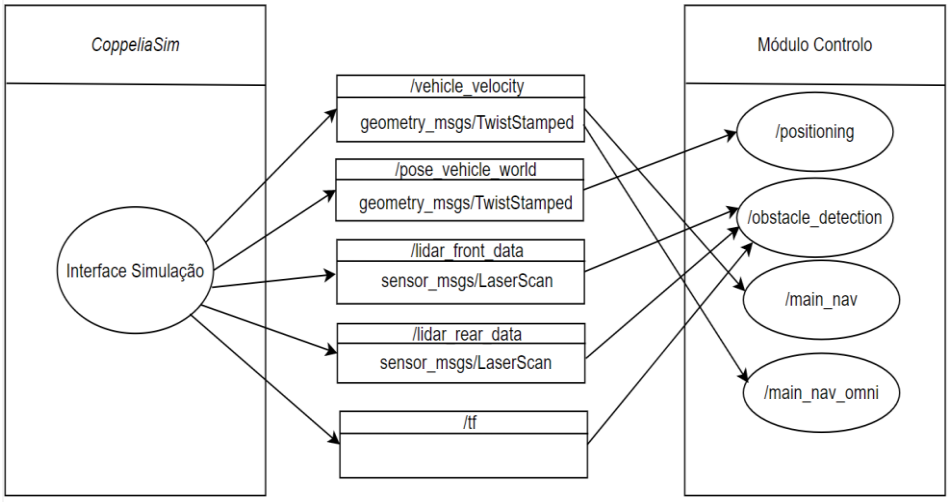


Figura 79: Comunicação ROS *topic* entre simulador e controlador.

Adicionalmente, e aproveitando as funcionalidades ROS, foi ainda implementado a biblioteca *tf2* que estabelece um padrão de controlo entre os referenciais de coordenadas, possibilitando a transformação de dados entre referenciais, no sistema configurado. Apesar de subentendido, a biblioteca utiliza tópicos para assegurar as comunicações, ROS *topic*.

8.1.3. Cenário Multi-Robô

Tendo em consideração o objetivo de gestão de uma possível frota de manipuladores móveis, requisito de um outro trabalho de dissertação, foi introduzido um segundo modelo dinâmico no cenário de simulação (Figura 80). Os modelos são similares entre si, variando os nomes dos objetos que os constituem e dos tópicos que estabelecem a comunicação com o controlador. O controlo é estabelecido

de forma independente, por intermédio do *middleware* ROS, partilhando os métodos e algoritmos de controlo e gestão de comportamentos.

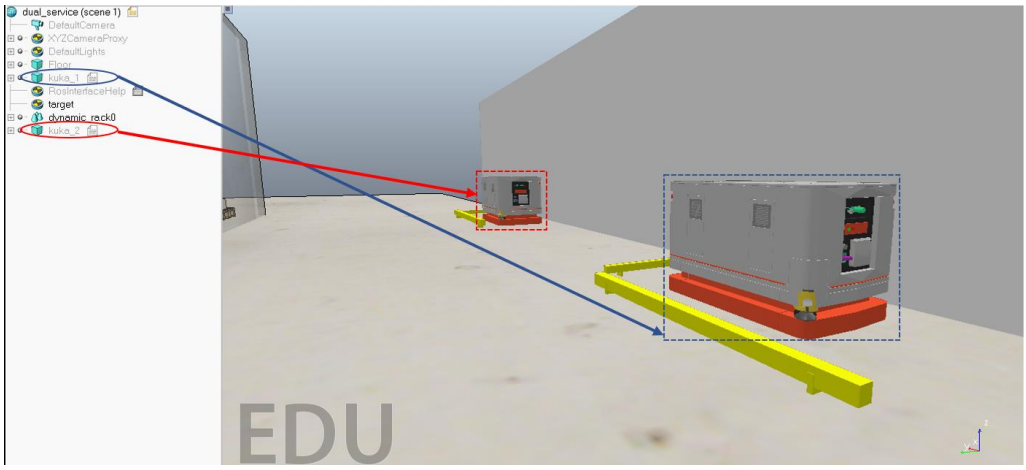


Figura 80: Ambiente de simulação composto por duas plataformas móveis.

8.2. Deteção de Obstáculos

A componente responsável por evitar colisões com obstáculos recorre a dois sensores *laser SICK* S300 *Expert*, já presentes na plataforma KMP 200 *Omnimove*, cujo intuito reside na capacidade de detetar a presença de obstáculos estáticos e dinâmicos durante o movimento da plataforma móvel. No entanto, a informação fornecida pelos sensores necessita de ser processada de modo que o controlador, que implementa a dinâmica de navegação autónoma, obtenha os dados num formato adequado. A Figura 81 apresenta um esquemático simplificado da estratégia de conversão de dados adotada, sendo destacado os módulos de transformação entre referenciais, entre a *pose* do sensor e o centro de massa do veículo, assim como a componente que restringe o número de feixes dos sensores àqueles de interesse aos sistemas dinâmicos não-lineares.

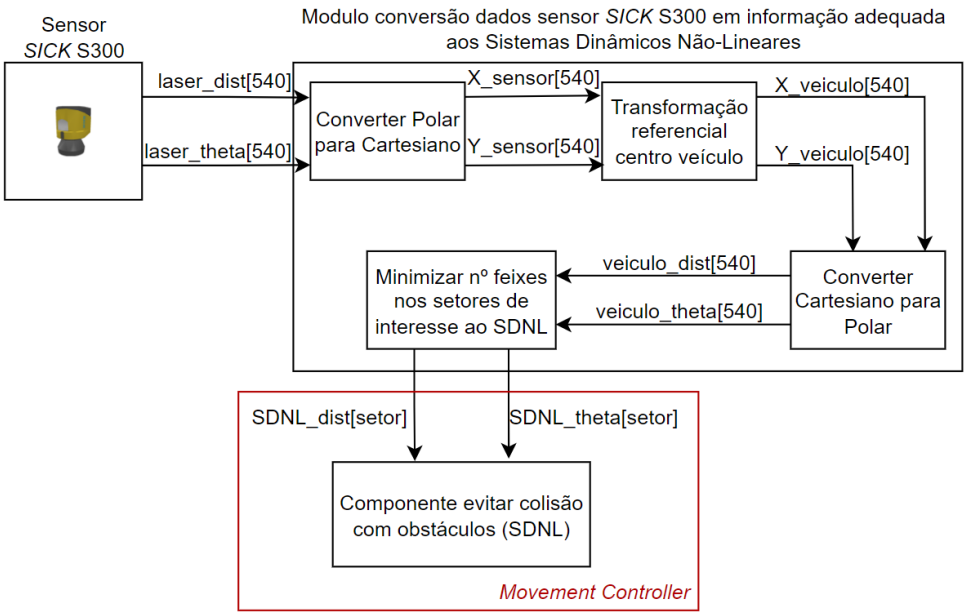


Figura 81: Esquemático simplificado da estratégia de conversão dos dados recolhidos pelo sensor *lidar*. A informação é recolhida numa projeção *laser* bidimensional, composta por feixes.

8.2.1. Conversão Dados Sensor *SICK* S300 em Setores de Interesse à Dinâmica dos Sistemas Dinâmicos Não-Lineares

Os sensores *SICK* S300, conforme o analisado no subcapítulo Sensores, têm a capacidade de sensorizar o ambiente circundante através de 540 feixes *laser* por sensor, contabilizando um total de 1080 feixes. Por forma a otimizar o algoritmo e garantir a estabilidade do método, verifica-se a necessidade de reduzir o número de setores de interesse para a dinâmica incumbida por evitar colisões com obstáculos. Como enunciado em Louro et al., (2019) e Bicho, (1999), apesar da redução da informação sensorial, o sistema consegue cumprir com os requisitos de navegação autónoma em ambientes dinâmicos.

a) Determinar número de setores a considerar na dinâmica de evitar obstáculos

A sensorização do ambiente circundante ao veículo, no caso dos sistemas dinâmicos não-lineares, foi realizada em relação ao centro de massa do veículo, estabelecendo um determinado número de setores, espaçados uniformemente entre si.

Os setores de interesse à dinâmica de evitar colisões com obstáculos estáticos e dinâmicos variam em função da janela de varredura, isto é, da gama de sensorização a considerar, assim como do número de setores considerados. Quanto maior for o número, maior a resolução de sensorização (menor espaçamento entre os setores), no entanto, é de relevar que o tempo de processamento da dinâmica de

navegação autónoma aumenta consideravelmente, podendo até ultrapassar o *timestep*, o que, no caso ponderado, poderia originar comportamentos irregulares ou incompreensíveis no movimento do veículo.

$$Espaçamento\ setores = \frac{Ângulo\ de\ varredura}{Número\ de\ setores} \tag{71}$$

Para o efeito, foi considerado uma janela de varredura de 210 ˚, definida por 21 setores. Nestas condições, com base na equação 71, determina-se o espaçamento entre setores, no caso igual a 10 ˚, θ= 10 ˚. O zero de referência foi considerado na direção frontal da plataforma móvel, isto é, no eixo das abcissas do referencial do veículo. Visto que na direção frontal do veículo equivale o ângulo de zero graus, foi necessário impor um número de setores ímpar. Por conseguinte, o ângulo de varredura varia entre -105 ˚e +105 ˚, primeiro e último setor, respetivamente.

Uma possível disposição dos setores de interesse à dinâmica de evitar obstáculos pode ser observada na Figura 82, sendo que *n* representa o número de setores a considerar.

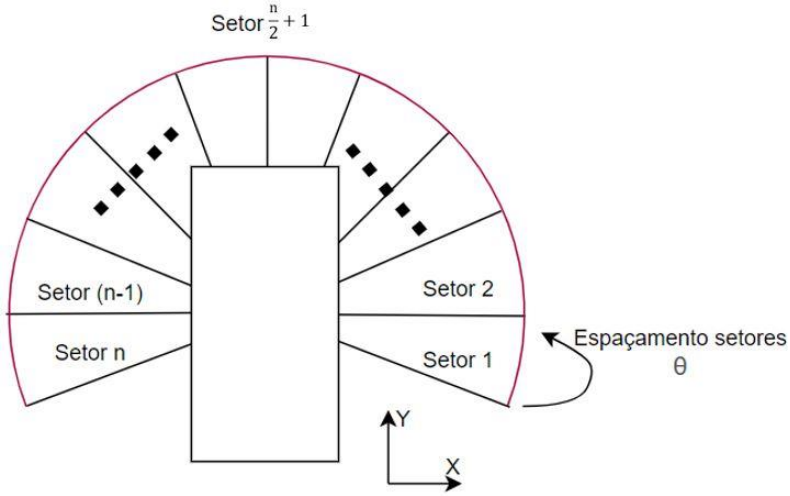


Figura 82: Possível distribuição de n setores de interesse à dinâmica de evitar obstáculos.

b) Determinar segmento de reta para cada setor

A estratégia de conversão dos dados dos sensores *SICK* S300 *Expert* em setores da dinâmica pode ser dividida em três fases, por cada sensor. Numa primeira fase, determina-se os pontos cartesianos de cada setor, considerando que não seja detetado qualquer obstáculo. Por sua vez, e com base na informação recolhida dos *lidars*, determina-se a equação do segmento de reta entre dois feixes *laser*.

Tendo esta informação, é possível determinar a distância aos obstáculos por cada setor, através do cálculo do ponto de interseção entre os segmentos de reta dos setores e dos feixes (dois consecutivos). Na condição de existir múltiplas soluções de interseção de retas válidas, opta-se por considerar apenas a menor distância, representando o pior cenário. Nos três tópicos que se seguem é analisada, com algum detalhe, a estratégia adotada para a conversão em setores da dinâmica.

i. Determinar pontos cartesianos de cada setor

Os pontos cartesianos de cada setor, admitindo que não sejam detetados quaisquer obstáculos, podem ser determinados pelas equações 72 e 73, considerando, para o efeito, a máxima distância detetável pelo sensor *SICK* S300, $dist_{max}$, bem como os ângulos de cada setor, θ_i, em relação ao referencial do veículo.

$$x_{\max\ \_setor_i} = dist_{max}\ \cos(\theta_i) \tag{72}$$

$$y_{\max\ \_setor_i} = dist_{max}\ \sin(\theta_i) \tag{73}$$

Os ângulos de cada setor determinam-se recorrendo à equação 74, tendo como referência a direção frontal da plataforma omnidirecional, sendo que i corresponde ao setor atual, n ao número de setores considerados e $gama_{varredura}$ à janela de varredura considerada.

$$\theta_i = -\frac{gama_{varredura}}{2} + \frac{gama_{varredura}}{n-1} \tag{74}$$

Na Figura 83 está exemplificada a condição de cálculo para o setor 9, $i = 9$, considerando um total de 21 setores.

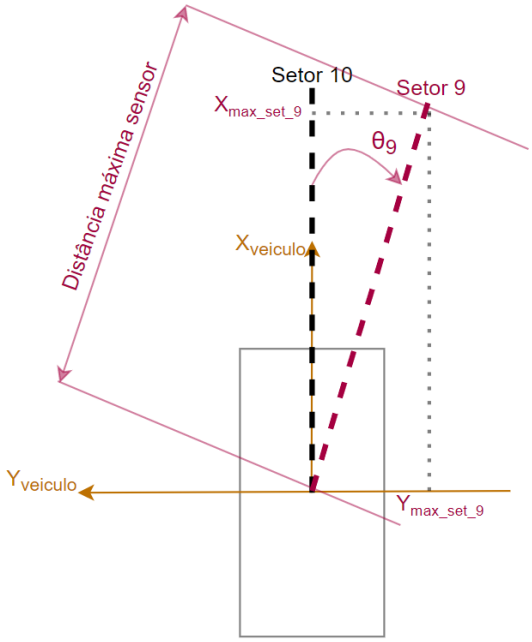


Figura 83: Ilustração da condição de cálculo do setor 9, tendo como referência a orientação frontal da plataforma omnidirecional. Os pontos cartesianos são obtidos para a distância máxima detetável admitida.

Assumindo o referencial da plataforma móvel, a partir do seu centro de massa, é possível determinar a equação dos segmentos de reta de cada setor, segundo a equação padrão 75, sendo que m corresponde ao declive do segmento reta e b ao coeficiente linear. Uma vez que os segmentos de reta intersetam a origem do referencial, como constatado na Figura 83, o coeficiente linear apresenta valor nulo, equação 76. Por sua vez, os declives dos segmentos de reta dos setores podem ser determinados segundo a equação 77, sendo que $y_{\max\ \_setor\_i}$ e $x_{\max\ \_setor\_i}$ correspondem aos pontos cartesianos do setor i, para o caso dos setores não detetarem obstáculos, determinados nas equações 72 e 73.

$$y = m\,x + b \tag{75}$$

$$b = 0 \tag{76}$$

$$m = \frac{y_{\max\ \_setor\_i}}{x_{\max\ \_setor\_i}} \tag{77}$$

ii. Determinar segmento de reta entre dois feixes *laser*

Na estratégia adotada, os segmentos de reta são calculados entre dois pontos cartesianos, correspondentes às posições finais de dois feixes consecutivos. Deste modo, recorrendo à equação 75,

determina-se a expressão que caracteriza cada segmento de reta. Cada segmento de reta define-se pelo seu declive, m_i, assim como pelo coeficiente linear associado, b_i.

$$m_i = \frac{y_{feixe_i-1} - y_{feixe_i}}{x_{feixe_i-1} - x_{feixe_i}} \quad ; \quad x_{feixe_i-1} - x_{feixe_i} \neq 0 \tag{78}$$

$$b_i = y_{feixe_i} - m_i \, x_{feixe_i} \tag{79}$$

A Figura 84 retrata os cálculos apresentados nas equações 78 e 79, considerando a deteção de um obstáculo por dois feixes *laser*, representados pelos segmentos de reta de cor azul, ao passo que o segmento de cor encarnada corresponde ao segmento de reta entre os dois pontos cartesianos dos feixes *laser*.

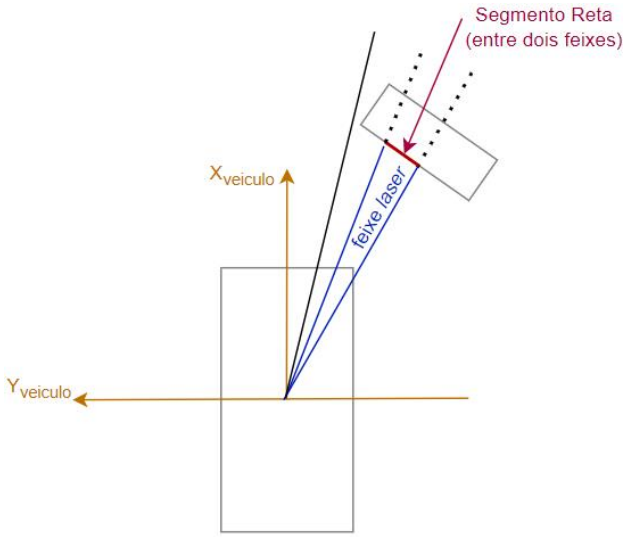


Figura 84: Ilustração do segmento de reta entre dois feixes *laser* consecutivos com distância inferior à máxima admitida.

iii. Interseção entre o segmento reta dos feixes e do setor dinâmico

No caso de o segmento de reta correspondente ao setor dinâmico estar compreendido no intervalo entre dois feixes consecutivos, o cálculo do ponto de interseção é realizado tendo em consideração as equações 80 e 81. As expressões recorrem aos parâmetros apresentados nos dois tópicos anteriores, sendo m e m_i relativos aos declives dos segmentos de reta que definem o setor dinâmico e o segmento de reta entre dois feixes consecutivos, respetivamente.

$$x_{inter,i} = \frac{b_i}{m - m_i} \quad ; \quad m - m_i \neq 0 \tag{80}$$

$$y_{inter,i} = m.\ x_{inter,i} \tag{81}$$

Devido à condição necessária de existência, $m - m_i \neq 0$, o cálculo do ponto de interseção foi dividido em três condições possíveis, considerando a indeterminação do cálculo de m, $x_{\max\ \_setor\_i} = 0$, assim como de m_i, $x_{feixe_i - 1} - x_{feixe_i} = 0$. Caso nenhuma das condições se verifique, o cálculo é realizado segundo o sistema de equações 80 e 81. As condições enunciadas encontram-se retratadas nos três tópicos que se seguem:

- Caso $\left(\boldsymbol{x_{feixe_i - 1} - x_{feixe_i}}\right) = \mathbf{0}$

Nesta condição o obstáculo foi detetado por dois feixes *laser* consecutivos que partilham o mesmo valor de x_{feixe}. Desse modo, o cálculo do ponto de interseção entre os segmentos de reta calcula-se segundo as equações 82 e 83.

$$x_{inter,i} = x_{feixe_i} \tag{82}$$

$$y_{inter,i} = m.\ x_{inter,i} \tag{83}$$

Como se pode verificar na Figura 85, o valor de $x_{inter,i}$ pode ser facilmente determinado, atribuindo diretamente o valor de x_{feixe_i} ou de $x_{feixe_{i-1}}$. Por sua vez, $y_{inter,i}$ calcula-se em função do valor de $x_{inter,i}$.

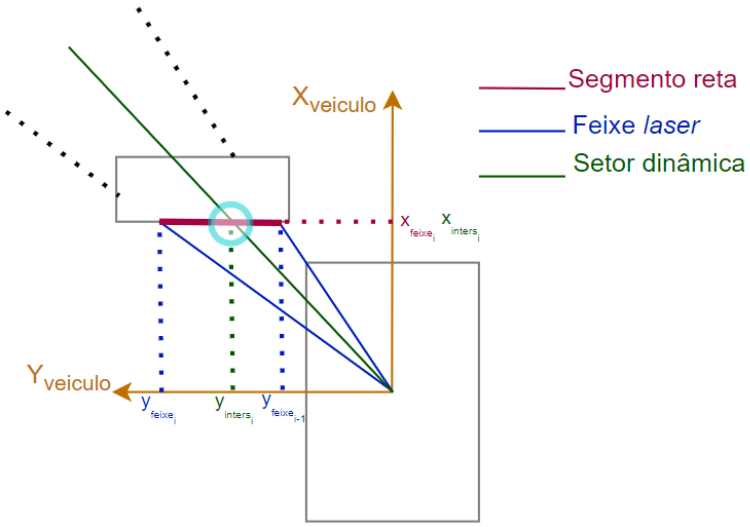


Figura 85: Condição de cálculo do ponto de interseção entre o feixe do setor de interesse à dinâmica e o segmento de reta determinado, se $(x_{feixe_i-1} - x_{feixe_i}) = 0$.

- Caso $(\boldsymbol{x_{\mathbf{max}\ \_setor\_i}}) = \mathbf{0}$

Neste caso, o obstáculo foi determinado para o setor dinâmico com orientação de $\theta_i = 90°$ ou $\theta_i = -90°$, coincidente com o eixo das ordenadas do referencial do veículo. Tendo em consideração a condição, o cálculo do ponto de interseção entre os segmentos de reta calcula-se segundo as equações 84 e 85.

$$x_{inter,i} = 0 \tag{84}$$

$$y_{inter,i} = b_i \tag{85}$$

Como se verifica na Figura 86, o obstáculo foi detetado por dois feixes consecutivos ao qual pertence o setor dinâmico $\theta_i = 90°$. Nessa condição, considera-se $x_{inter,i}$ como nulo, determinando-se apenas o valor de $y_{inter,i}$, que corresponde ao coeficiente linear.

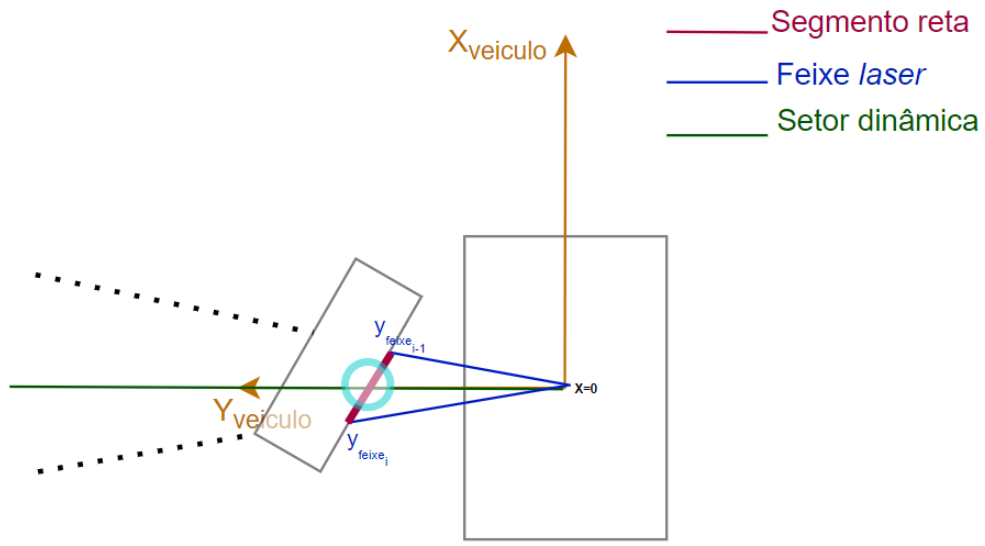


Figura 86: Condição de cálculo do ponto de interseção entre o feixe do setor de interesse à dinâmica e o segmento de reta determinado, se $(x_{max\ \_setor\_i}) = 0$.

- Caso não seja verificada nenhuma das condições enunciadas

Para o caso de nenhumas das condições anteriores se verificar, o cálculo do ponto de interseção realiza-se segundo o sistema de equações descrito por 80 e 81. A Figura 87 idealiza esta condição, considerada como *standard*.

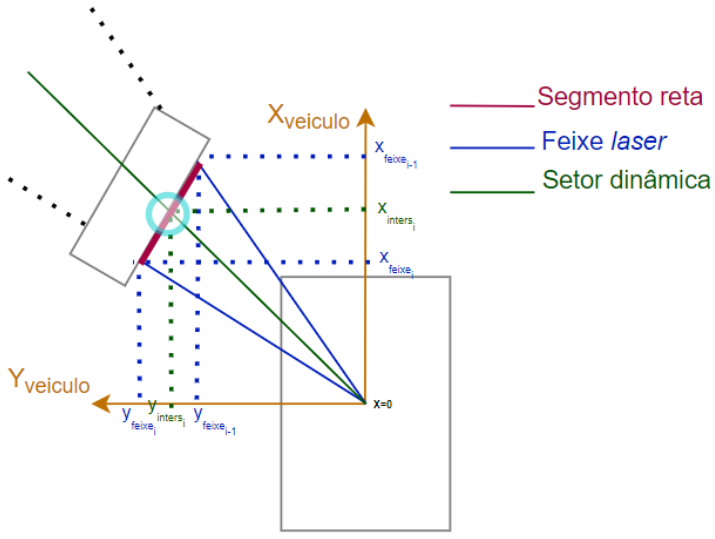


Figura 87: Condição de cálculo do ponto de interseção do feixe entre o setor de interesse à dinâmica e o segmento de reta determinado, na condição *standard*.

8.2.2. DETERMINAR DISTÂNCIA INTERNA DA PLATAFORMA MÓVEL

A dinâmica de evitar obstáculos, no caso dos sistemas dinâmicos não-lineares, recorre a um conjunto de setores de interesse, com origem no centro de massa da plataforma móvel. Dessa forma, o comprimento de cada setor inclui o corpo do próprio veículo, originando distâncias erradicas ao controlador. Por conseguinte, verifica-se a necessidade de determinar, para cada um dos setores definidos, as distâncias internas da plataforma omnidirecional.

a) Divisão do veículo em duas secções de cálculo

As distâncias internas da plataforma móvel determinam-se considerando duas secções de cálculo possíveis: a região frontal e traseira do veículo e as laterais, esquerda e direita.

Como se verifica na Figura 88, a plataforma móvel foi aproximada a um retângulo, considerando as dimensões reais do KMP 200 *Omnimove*. As secções admitidas encontram-se destacadas na Figura 88, tendo como limite as diagonais do retângulo idealizado.

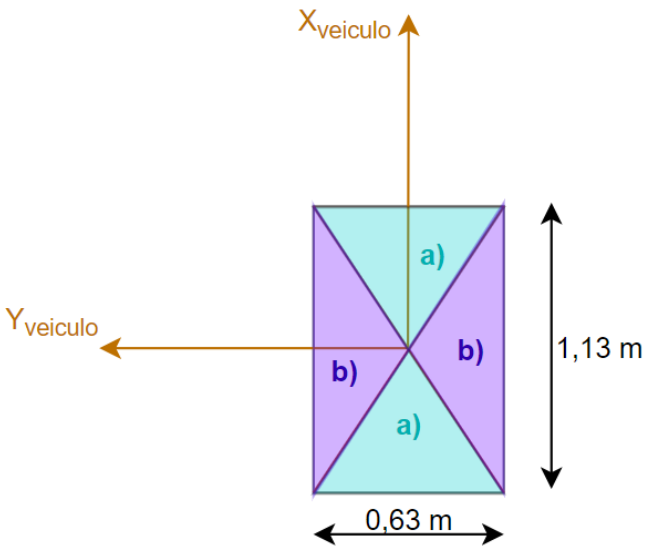


Figura 88: Ilustração das secções de cálculo consideradas. Região frontal e traseira: a); Laterais esquerda e direita: b).

b) Cálculo distâncias internas por setor

A estratégia de cálculo recorre à trigonometria como forma de determinar os ângulos que limitam as secções, α_a e α_b, assim como a distância interna, estabelecendo segmentos de reta sobrepostos aos setores de interesse à dinâmica de evitar obstáculos.

O valor de α pode ser obtido pela equação 86, tendo em consideração as dimensões da plataforma retangular. No caso específico do KMP 200 *Omnimove*, considera-se uma largura de 0,63 m e um comprimento de 1,13 m, obtendo $\alpha \cong 29{,}14^{\circ}$.

$$\alpha = \tan^{-1}\left(\frac{\frac{largura}{2}}{\frac{comprimento}{2}}\right) \tag{86}$$

Atendendo ao referencial do veículo apresentado na Figura 89, é possível concluir que α_a e α_b são valores simétricos, sendo que $\alpha_a \cong -29{,}14^{\circ}$ e $\alpha_b \cong 29{,}14^{\circ}$, definindo as duas secções de cálculo apresentadas.

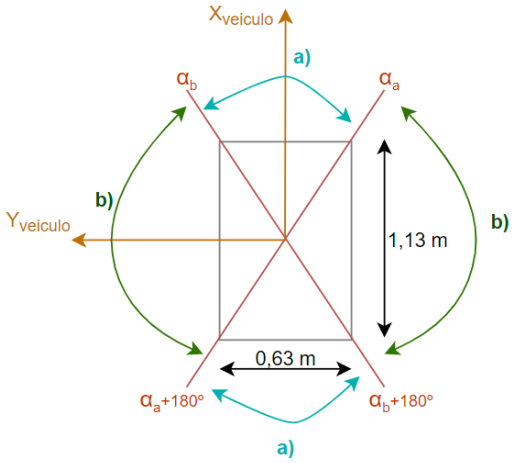


Figura 89: Representação dos limites entre secções. Verifica-se ainda a simetria entre α_a e α_b.

i. No caso de o setor estar compreendido na região frontal ou traseira do veículo

O segmento de reta correspondente à distância interna da plataforma móvel determina-se segundo a equação 87, tendo em consideração o ângulo do setor de interesse da componente de evitar obstáculos, bem como do comprimento do próprio veículo.

$$dist\_inter_i = \left|\frac{(\frac{comprimento}{2})}{\cos(\theta_i)}\right| \tag{87}$$

O cálculo é realizado para todos os setores que estejam compreendidos na gama destacada na Figura 90.

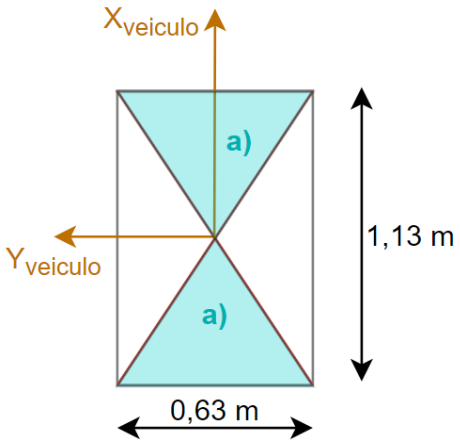


Figura 90: Destaque da secção frontal e traseira da plataforma móvel.

ii. No caso de o setor estar compreendido na região lateral do veículo

A distância interna da plataforma móvel, nesta condição, determina-se recorrendo à equação 88, em função da largura do veículo e do ângulo do setor da dinâmica, θ_i.

$$dist\_inter_i = \left| \frac{(\frac{largura}{2})}{\sin(\theta_i)} \right| \quad (88)$$

No caso de os setores dinâmicos estarem contidos na região destacada na Figura 91, a distância interna da plataforma móvel calcula-se segundo a equação 88.

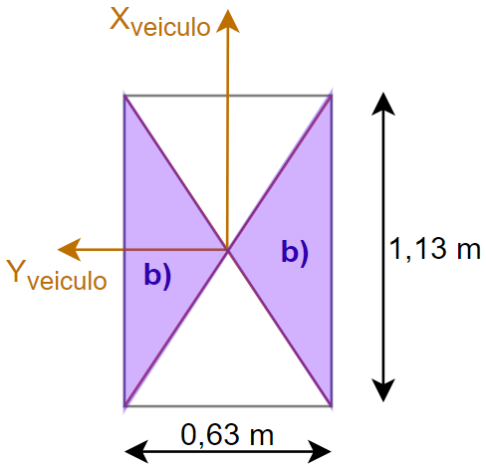


Figura 91: Destaque das secções laterais da plataforma móvel.

8.2.3. DETERMINAR DISTÂNCIAS LINEARES EM RELAÇÃO AO CORPO DO VEÍCULO

A tarefa de navegação autónoma em piso fabril deve considerar a manutenção de um afastamento mínimo de segurança entre o corpo do veículo e o obstáculo mais próximo. Desse modo, verifica-se a necessidade de determinar, em todos os instantes, as distâncias mínimas lineares entre o limite da plataforma móvel e o obstáculo mais próximo. Durante as manobras de acostagem, caracterizadas pela sua natureza crítica e delicada, as distâncias lineares podem ser consideradas tanto como mecanismo de segurança, embora seja admitido uma margem menor aos obstáculos, como também de obtenção de uma maior precisão durante a fase de aproximação.

a) Estratégia obtenção distâncias lineares

As distâncias lineares foram obtidas em quatro secções, abrangendo todos os limites da plataforma móvel. As distâncias lineares frontais e laterais direita foram obtidas pelo sensor *SICK* S300 frontal do veículo. Identicamente, as distâncias traseiras e laterais esquerda foram recolhidas pelo sensor *laser* traseiro da plataforma móvel. As posições dos obstáculos foram consideradas no referencial do veículo, através da relação de referenciais entre os sensores e o referencial da plataforma móvel.

Similarmente ao realizado nos setores de interesse na dinâmica de navegação autónoma, foi considerado um conjunto de setores lineares nas secções laterais, 31 setores em relação às laterais esquerda e direita e 15 setores na região frontal e traseira da plataforma móvel. A discrepância do número de setores utilizados deve-se, naturalmente, à diferença das dimensões do veículo omnidirecional. Numa primeira fase, é verificado se os pontos cartesianos determinados pelos sensores

laser coincidem com a secção de interesse, isto é, se se encontram na zona abrangida pelos setores anteriormente definidos, verificando ainda qual das quatro secções possíveis se enquadra. Caso o obstáculo detetado se encontre na região frontal ou traseira da plataforma móvel, a distância linear obtém-se segundo a equação 89. Assim, segundo o referido, a distância linear, em relação ao referencial do veículo, encontra-se na coordenada das abcissas, x_{feixe_i}, sendo necessário descontar a distância interna do veículo, correspondente a metade do comprimento.

$$dist_{linear_j} = |x_{feixe_i}| - \frac{comprimento}{2} \tag{89}$$

Não obstante, no caso de o obstáculo detetado esteja contido na região lateral, direita ou esquerda, a distância lateral determina-se através da equação 90. A distância linear obtém-se diretamente pela coordenada das ordenadas, y_{feixe_i}, em relação ao referencial do veículo. Do mesmo modo, é retirado o segmento de reta correspondente à distância interna do veículo, no caso, metade da largura da plataforma móvel.

$$dist_{linear_j} = |y_{feixe_i}| - \frac{largura}{2} \tag{90}$$

A Figura 92 exemplifica a condição de deteção de um obstáculo na secção lateral direita da plataforma móvel. Neste caso, as distâncias laterais são obtidas pelo sensor frontal do veículo, obtendo a coordenada do obstáculo. O valor da distância linear, neste contexto obtida no eixo das ordenadas, é atribuído ao setor linear que contém o valor das abscissas mais próximo, definindo uma margem de *threshold* na seleção do setor. A margem de escolha entre os setores lineares corresponde à metade da distância entre dois setores lineares consecutivos.

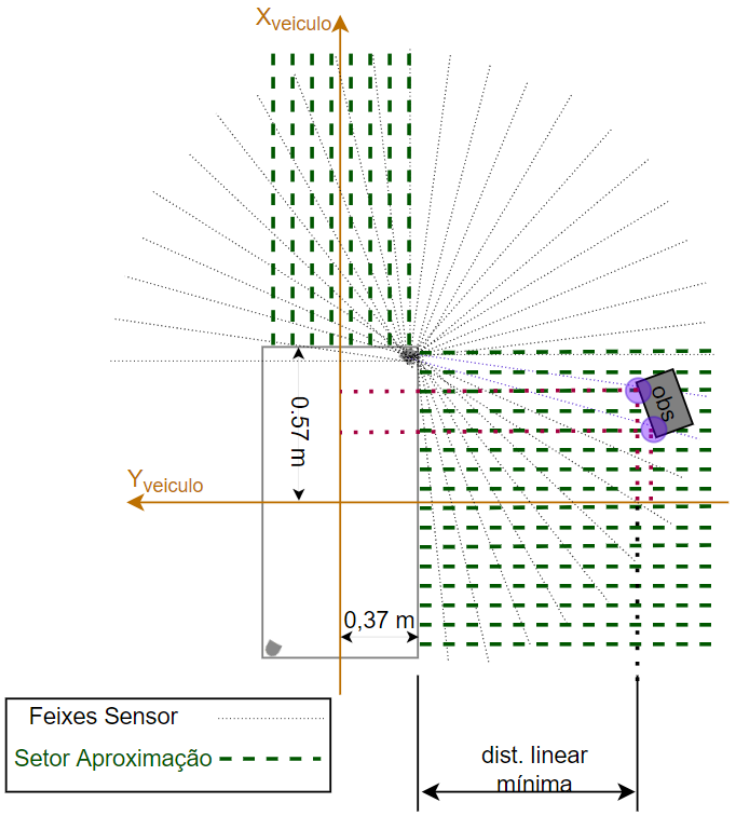


Figura 92: Ilustração da condição de um obstáculo ser detetado na secção lateral direita do veículo. Apesar de o obstáculo ser detetado por vários setores, o algoritmo considera apenas a menor distância determinada. Dada a localização do obstáculo, as informações são recolhidas pelo sensor frontal.

A distância mínima ao obstáculo pode ser obtida através do menor valor do conjunto dos setores lineares da secção correspondente. Extraindo a distância mínima das quatro secções consideradas, é possível definir margens de segurança durante a navegação relativamente à distância entre o veículo e o obstáculo mais próximo, implementando mecanismos de segurança com base na distância e na sensibilidade da manobra.

8.2.4. Método de Deteção de Obstáculos: *Obstacle Detection*

A informação sensorial recolhida pelos dois sensores *lidar SICK* S300 é processada e manipulada pela componente *obstacle detection*. Os dados sensoriais são recebidos tendo por base o *middleware* ROS, estabelecido pelo mecanismo de comunicação ROS *topics*. Desta forma, o algoritmo de execução de tratamento e processamento de dados abstrai-se do emissor da informação. Consequentemente, o mecanismo encontra-se capacitado de ser implementado num sistema robótico real bem como em ambientes de simulação computacionais, considerando a uniformidade da receção dos dados. A informação sensorial é recebida pelos tópicos */lidar_front_data* e */lidar_rear_data*, relativos aos

sensores posicionados na secção frontal e posterior, respetivamente, seguindo a estrutura de mensagem típica dos sensores *SICK* S300. Os tópicos são definidos pelo tipo de mensagem ROS *sensor_msgs/LaserScan*, contendo campos identificativos da mensagem, informação dos parâmetros de sensorização e da informação do espaço circundante ao sistema.

Uma vez comprovada a receção de uma nova mensagem, são realizados dois métodos de processamento de informação distintos, um relativo à determinação dos setores de interesse à dinâmica de navegação e outro relativo ao cálculo das distâncias lineares de segurança. O algoritmo que implementa os processos segue os fundamentos indicados no subcapítulo Deteção de Obstáculos.

O método permanece em estado *idle* até que seja verificado a receção de dados sensoriais mais atualizados. Por sua vez, os novos dados são convertidos tendo como referência o centro de massa do veículo, possibilitando os cálculos relacionados à determinação da distância associada a cada setor dinâmico assim como das distâncias lineares de segurança. A informação processada é enviada aos módulos que implementam os comportamentos de geração de movimentos, via ROS *topics*.

A máquina de estados representada na Figura 93 ilustra o fluxo de execução dos métodos que implementam o algoritmo enunciado.

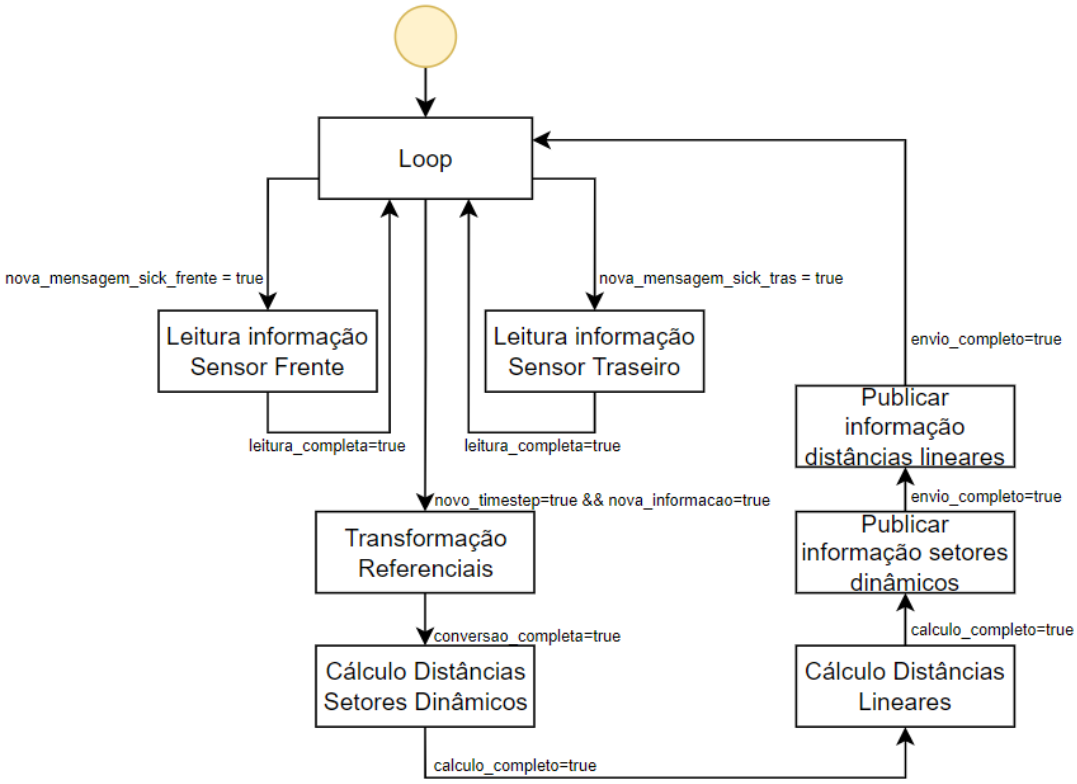


Figura 93: Máquina de estados que define a componente *obstacle detection*, responsável pela aquisição e processamento da informação recolhida pelos sensores *laser*.

As distâncias discriminadas em setores de interesse à dinâmica de navegação autónoma são publicadas a cada ciclo de computação no tópico */sector_dist_obs*. Conforme anteriormente analisado, os algoritmos de navegação autónoma em espaços dinâmicos necessitam da informação do espaço envolvente ao sistema, num conjunto de distâncias e ângulos, representando o espaço segundo um plano setorial. Esta informação encontra-se contida no tipo de mensagem *miar_msgs::distance_sector* definida no tópico. A Tabela 6 expõe a estrutura da mensagem, destacando as variáveis *theta_obs* e *distances*.

Tabela 6: Estrutura e descrição do tipo de mensagem que permite a comunicação entre a componente *obstacle detection* e as componentes *navigation* e *navigation omni*.

Nome Variável	Tipo de Variável	Breve Descrição
header	std_msgs/Header	Identificação da Mensagem
nSectors	int	Número de Setores Considerados
range_field	float	Ângulo de Varrimento
theta_obs	float	Orientação de Cada Setor
distances	float	Distância Absoluta na Respetiva Orientação

Como constatável na Figura 94, apenas os comportamentos de *navigation* e *navigation omni* subscrevem ao tópico */sector_dist_obs*.

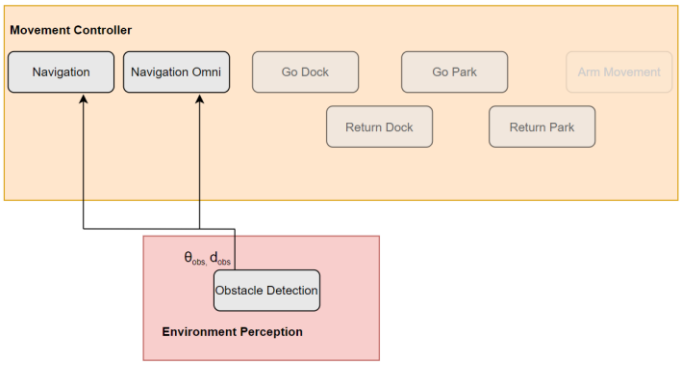


Figura 94: Comunicação via ROS *topic* entre a componente *obstacle detection* e as componentes *navigation* e *navigation omni*.

As distâncias lineares do espaço envolvente são publicadas a cada ciclo de computação no tópico */protec_dist_obs*. Este tópico é descrito pela variável *miar_msgs::protect_distance*, estruturada de forma

a conter não só os dados sensoriais processados, como também informação relacionada com a identificação da mensagem e especificações de sensorização. A Tabela 7 descreve a estrutura da mensagem contida no tópico, destacando as variáveis que contêm a informação das distâncias lineares. Na mesma mensagem existem ainda elementos que permitem determinar a localização de um objeto em relação ao plano do próprio veículo.

Tabela 7: Estrutura e descrição da estrutura da mensagem relativa à mensagem *miar_msgs::protect_distance*, contendo as distâncias lineares entre o veículo e os obstáculos. A mensagem é publicada para todas as componentes do módulo *Movement Controller* que definem o movimento do veículo.

Nome Variável	Tipo de Variável	Breve Descrição
header	std_msgs/Header	Identificação da Mensagem
front_dist_obs	float	Conjunto Distâncias Frontais
rear_dist_obs	float	Conjunto Distâncias Traseiras
left_dist_obs	float	Conjunto Distâncias Setor Esquerdo
right_dist_obs	float	Conjunto Distâncias Setor Direito
min_lateral_position	float	Primeiro Feixe Linear Lateral
min_fr_position	float	Primeiro Feixe Linear Frente/Trás
inc_between_points_lateral	float	Incremento entre Feixes Laterais
inc_between_points_fr	float	Incremento entre Feixes Frente/Trás

Todos os comportamentos relacionados com a geração de movimentos do veículo omnidirecional subscrevem ao tópico */protec_dist_obs*, embora com finalidades distintas. Como já verificado, os comportamentos de navegação autónoma em ambientes dinâmicos implementam o mecanismo de paragem de segurança em função das distâncias lineares, nos quatro quadrantes do veículo. Por sua vez, os módulos que implementam os comportamentos de acostagem além de implementarem mecanismos de paragem de segurança, também realizam correções de movimento considerando as distâncias lineares, caso a direção do movimento seja coincidente com o obstáculo detetado. A Figura 95 esquematiza, simplificadamente, os módulos que subscrevem ao tópico */protec_dist_obs*.

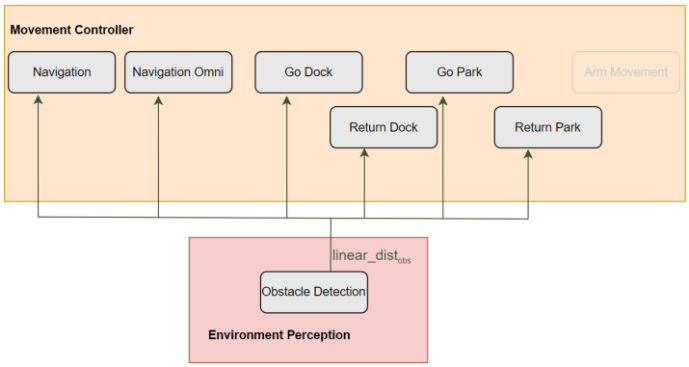


Figura 95: Comunicação via ROS *topic* entre a componente *obstacle detection* e todas as componentes do módulo *Movement Controller* que subscrevem ao tópico */protec_dist_obs*.

8.3. NAVEGAÇÃO NÃO-HOLONÓMICA: *NAVIGATION*

A componente *navigation* controla o movimento da plataforma móvel entre dois pontos definidos no espaço. Efetivamente, o controlador segue um conjunto de *via points* que definem a rota entre ambos os pontos, tendo em consideração a presença de obstáculos estáticos e dinâmicos. Os *via points* são recebidos via ROS *action server* (Figura 96), do *task manager*, na receção do *goal*.

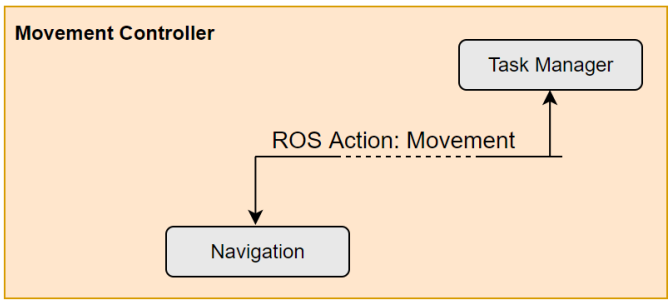


Figura 96: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *navigation*.

A Tabela 8 apresenta a estrutura de mensagem relativa à comunicação bidirecional entre o *task manager* e o *navigation*, estabelecida pelo ROS *action movement*.

Tabela 8: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *task manager* e a componente *navigation*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	route	miar_msgs/GraphNode	Definição da rota a ser percorrida
	action_orientation	float	Orientação final de acostagem
Result	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado da tarefa
Feedback	route_performed	miar_msgs/GraphNode	Rota já percorrida
	vrobot	float	Velocidade longitudinal determinada
	wrobot	float	Velocidade angular determinada

Apesar das capacidades holonómicas da plataforma, o movimento deve ser legível aos operadores humanos, isto é, ser facilmente percetível ao ser humano. Para tal, o movimento deve-se assemelhar ao de um veículo com sistema de direção convencional. Numa plataforma omnidirecional, este tipo de movimento pode ser conseguível recorrendo apenas a dois graus de liberdade, relativos ao vetor velocidade longitudinal, v_x, e velocidade angular, w.

A implementação do comportamento de navegação autónoma legível segue a estrutura de uma máquina de estados, semelhante à apresentada na Figura 97. O fluxo de execução depende da comunicação com o *task manager*, concretamente da receção de um novo *goal* ou de interrupção da execução do processo, assim como do estado de execução do próprio processo.

Como representado no fluxograma da Figura 97, numa fase inicial, o processo encontra-se em *idle* até à receção de um novo *goal* do *task manager*. Uma vez verificada a interrupção ROS *Action*, a mensagem é aceite e armazenada segundo o seu conteúdo. Após o tratamento da mensagem recebida, inicia-se o algoritmo de navegação autónomo, baseado nos sistemas dinâmicos não-lineares, abordado no subcapítulo Sistemas Dinâmicos Não-Lineares. Este determina o valor da velocidade longitudinal e angular, em função da posição do *via point* i, referente à rota recebida, do posicionamento do veículo no espaço e da informação sensorial do espaço circundante. Contudo, apesar da dinâmica considerar a presença dos obstáculos, o módulo implementa uma verificação das distâncias laterais entre os limites da plataforma móvel e os obstáculos. No caso de as distâncias de segurança serem cumpridas, a dinâmica determinada é enviada à interface (no caso ao simulador), caso contrário o veículo fica imóvel, enviando o estado para o *task manager*. O processo permanece neste ciclo até que o sistema alcance o *via point* i e não seja aferido nenhum erro, considerando que o processo não seja interrompido externamente pelo *task manager*. Uma vez verificada a condição de proximidade ao *via point*, é verificado

se este corresponde ao último da rota solicitada. Em caso afirmativo, o processo de navegação termina em estado de sucesso. De outro modo, considera-se o próximo *via point* da rota recebida, repetindo o ciclo enunciado. Como é possível verificar na Figura 97, o processo pode ser interrompido em qualquer estado, caso o *client* do *movement Action* o solicite, no caso o *task manager*.

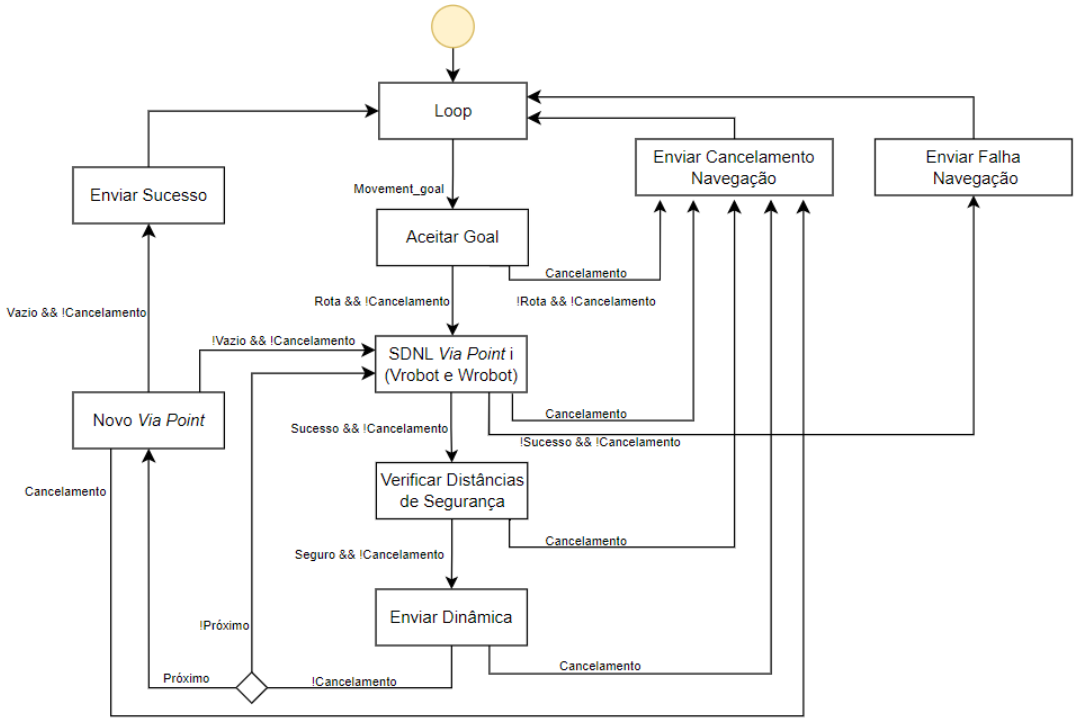


Figura 97: Máquina de estados que define o comportamento de navegação autónoma não-holonómica.

8.4. Navegação Holonómica

A componente *navigation omnidirectional* define o movimento da plataforma omnidirecional entre dois pontos no espaço. O percurso encontra-se definido por um conjunto de *via points* visando otimizar o trajeto e suavizar as manobras definidas pelo veículo. A dinâmica de navegação, similarmente ao comportamento *navigation*, considera a presença de obstáculos estáticos e dinâmicos no ambiente de navegação. O conjunto de *via points* é recebido via ROS *action server* (Figura 98), do *task manager*, aquando da receção do *goal*.

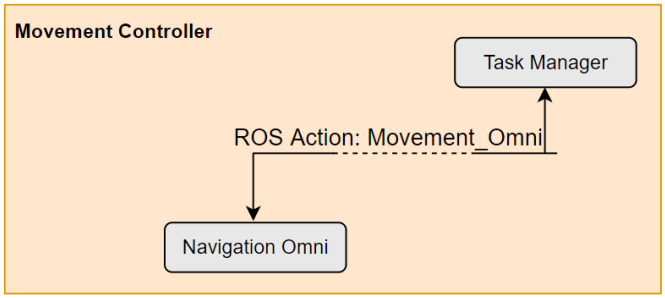


Figura 98: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *navigation omni*.

A Tabela 9 expõe a estrutura de mensagem relativa à comunicação bidirecional entre o *task manager* e o *navigation omnidirectional*, estabelecida pelo ROS *action movement_omni*.

Tabela 9: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *task manager* e a componente *navigation omni*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	route	miar_msgs/GraphNode	Definição da rota a ser percorrida
	action_orientation	float	Orientação final de acostagem
Result	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado da tarefa
Feedback	route_performed	miar_msgs/GraphNode	Rota já percorrida
	vx_robot	float	Velocidade longitudinal determinada
	vy_robot	float	Velocidade lateral determinada
	wrobot	float	Velocidade angular determinada

O movimento define-se pelos três graus de liberdade da plataforma omnidirecional, permitindo obter uma maior flexibilidade e manobrabilidade na execução das manobras. Estas características revelam especial interesse no caso de o veículo navegar em espaços mais exíguos. No entanto, o veículo pode apresentar movimentos complexos e pouco intuitivos. Dessa forma, contrariamente ao módulo *navigation*, os movimentos podem ser não legíveis aos operadores humanos.

O fluxo de execução do algoritmo de controlo de navegação omnidirecional segue a estrutura de máquina de estados presente na Figura 99. A transição entre estados encontra-se dependente da receção de um novo *goal*, do *task manager*, assim como do estado do próprio processo. Ademais, o processo pode ser interrompido externamente, em qualquer estado, caso o *client* do ROS *action* o solicite.

Na primeira fase, enquanto não for verificada a receção de um *goal* do cliente do ROS *action*, o processo encontra-se preso num ciclo de espera. Após a receção do pedido, a mensagem é aceite e armazenada segundo o conteúdo do *goal*. O método de navegação autónoma inicia-se, baseado nos sistemas dinâmicos não-lineares ajustado às capacidades holonómicas da plataforma móvel, conforme o descrito no capítulo Navegação Omnidirecional. O controlador implementa o movimento do veículo, definindo os vetores velocidade linear, nas respetivas componentes cartesianas, e velocidade angular. De modo idêntico ao de navegação legível, este determina as componentes de controlo em função da posição do *via point* i, bem como da informação do espaço circundante ao veículo e do seu posicionamento no espaço. As distâncias de segurança, em relação aos obstáculos próximos, são validadas antes da dinâmica de controlo ser enviada à interface. Caso as distâncias de segurança não sejam efetivas, o veículo permanece imóvel até que a condição se verifique. Todavia, nas restantes situações, a dinâmica é enviada. Este processo iterativo repete-se até que o sistema esteja na proximidade do *via point* i e não seja verificado nenhum erro, considerando que o processo não seja interrompido externamente pelo *task manager*. Uma vez validada a condição de proximidade ao *via point*, o algoritmo verifica se este corresponde ao último da rota recebida. No caso de o confirmar, o processo termina em estado de sucesso, retornando ao estado de *idle*. Caso contrário, a dinâmica de controlo considera o próximo *via point* como alvo, repetindo o ciclo descrito. Como constatável na Figura 99, o processo pode ser interrompido em qualquer estado, na condição de o *client* do *movement_omni Action* o solicitar, no caso o *task manager*.

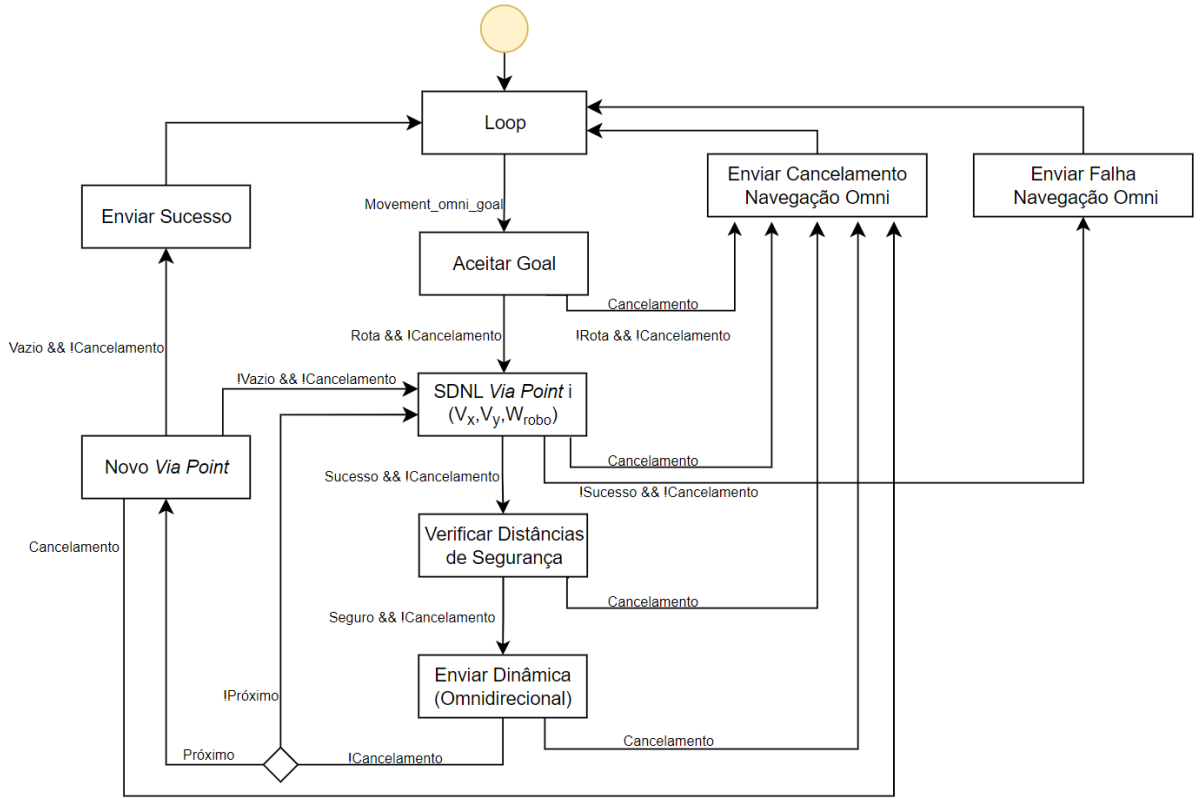


Figura 99: Máquina de estados que define o comportamento de navegação autónoma holonómica.

8.5. Manobras de Acostagem

As manobras de acostagem definem-se por quatro métodos distintos, diferenciados pela finalidade da operação, isto é, se é um processo de aproximação ou de afastamento ao local de acostagem, e pelo local de acostagem, podendo ser caracterizado como um local de trabalho ou local de estacionamento ou de carga (das baterias).

8.5.1. Comportamento de Aproximação aos Locais de Trabalho: *Go to Dock*

O módulo *go to dock* define o comportamento de aproximação aos locais de trabalho especificados. Os movimentos são gerados por um controlador linear do tipo proporcional, com base no erro de *pose*, em relação à posição e orientação final desejada. O algoritmo requer como *input* a *pose* final de acostagem para cada tarefa, flexibilizando o método para os diversos locais de acostagem. O mecanismo encontra-se capacitado de executar correções à trajetória gerada, bem como de imobilizar o veículo, caso a norma de segurança seja violada. Similarmente aos comportamentos analisados anteriormente, este estabelece comunicação via ROS *action server* (Figura 100) com o *task manager*, possibilitando a receção do *goal*,

no qual contém a *pose* final desejada, assim como o fornecimento de *feedback* periódico e do estado final da tarefa ao *server*.

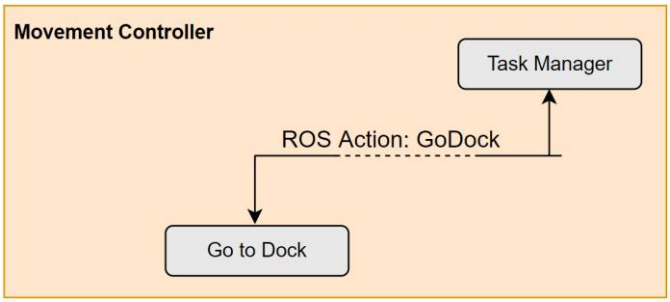


Figura 100: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *go to dock*.

A Tabela 10 apresenta a estrutura de mensagem relativa à comunicação bidirecional entre o *task manager* e o *go to dock*, estabelecida pelo ROS *action go dock*.

Tabela 10: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *task manager* e a componente *go to dock*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	targetPose	miar_msgs/GraphNode	Definição local de acostagem
	action_orientation	float	Orientação final de acostagem
	state_machine_index	int	Máquina de estados (do processo)
	action_goal	string	Especificar tipo de tarefa
Result	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado da tarefa
Feedback	vx_robot	float	Velocidade longitudinal determinada
	vy_robot	float	Velocidade lateral determinada
	wrobot	float	Velocidade angular determinada
	state_machine_index	int	Index máquina de estados (processo)

O movimento define-se pelos três graus de liberdade do veículo, obtendo uma maior agilidade e manobrabilidade durante a aproximação, facto que se considera um atributo pertinente em espaços complexos. Contudo, o veículo pode descrever movimentos complexos e pouco intuitivos. Assim, apesar dos mecanismos de segurança associados, a presença de operadores humanos deve ser evitada durante

a tarefa de aproximação, devido à ilegibilidade das manobras e à restrita margem de segurança considerada.

O algoritmo de aproximação segue uma estrutura do tipo máquina de estados, similar ao enunciado nos comportamentos anteriores. A transição entre estados pode ocorrer aquando da receção de um novo *goal* do *task manager*, caso o sistema esteja contido em estado de *idle*, como também do estado do próprio processo, no caso de execução do algoritmo de geração de trajetórias. O processo pode ser interrompido em qualquer estado, caso o módulo receba o pedido do ROS *client*.

Numa primeira iteração, o processo encontra-se num *loop* até que seja verificado a receção de um novo *goal* válido via ROS *action*. Uma vez aceite e armazenado, inicia-se o algoritmo de geração de movimentos. Nesta etapa, o controlador proporcional, conforme o erro de *pose*, estabelece o movimento de aproximação abstraindo-se do espaço envolvente ao veículo. Por sua vez, o módulo verifica a presença de obstáculos na trajetória definida pelo controlador. Caso se verifique a possibilidade de colisão, o mecanismo de segurança executa correções ao movimento estipulado, podendo imobilizar o veículo no caso de as condições mínimas de segurança não serem validadas. O algoritmo de geração de movimentos é executado enquanto o erro de *pose* for superior ao aceitável. A Figura 101 ilustra, com algum grau de abstração associado, o fluxo de execução do algoritmo enunciado.

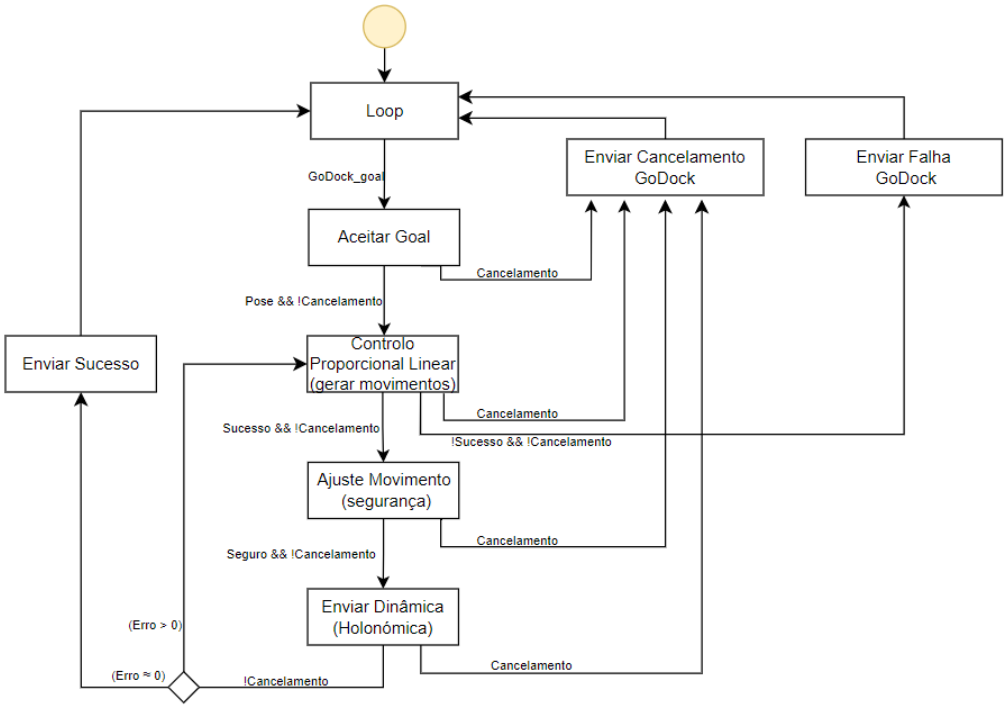


Figura 101: Máquina de estados que define o comportamento de aproximação ao local de acostagem.

8.5.2. Comportamento de Aproximação aos Locais de Park: *Go to Park*

A estratégia de controlo da geração de movimentos de aproximação ao local de *park* fundamenta-se no *go to dock*. A principal diferença incide nas restrições de aproximação, devido ao ponto de carga da bateria do veículo. O módulo estabelece comunicação bidirecional com o *task manager* via ROS *action server* (Figura 102), recebendo deste a *pose* desejada de acostagem, já admitindo o *offset* enunciado no subcapítulo Manobras de Aproximação aos Locais de *Park*, como *goal*. O *task manager* recebe periodicamente informação do estado do processo como *feedback*, assim como o resultado final da tarefa.

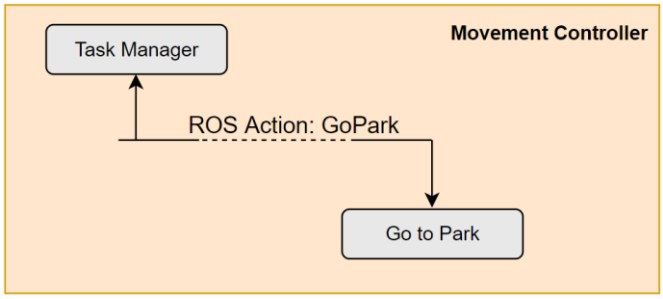


Figura 102: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *go to park*.

A Tabela 11 apresenta a estrutura de mensagem relativa à comunicação bidirecional entre o *task manager* e o *go to park*, estabelecida pelo ROS *action go park*.

Tabela 11: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *task manager* e a componente *go to park*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	targetPose	miar_msgs/GraphNode	Definição local de acostagem
	action_orientation	float	Orientação final de acostagem
	state_machine_index	int	Máquina de estados (do processo)
	action_goal	string	Especificar tipo de tarefa
Result	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado da tarefa
Feedback	vx_robot	float	Velocidade longitudinal determinada
	vy_robot	float	Velocidade lateral determinada
	wrobot	float	Velocidade angular determinada
	state_machine_index	int	Index máquina de estados (processo)

O mecanismo de controlo aplicado ao método de aproximação ao local de *park* assemelha-se ao enunciado para o comportamento de *go to dock*, partilhando os mecanismos de segurança e o método de geração de movimentos. Contudo, devido às restrições de aproximação aos conectores de carga no solo, verificou-se a necessidade de adicionar um mecanismo auxiliar de movimentos. Este encontra-se implementado como suplemento à função que define o controlador proporcional, priorizando numa primeira fase a correção da orientação de navegação. Ademais, o mecanismo estabelece uma trajetória circular, facilitando a atuação do mecanismo de correção de movimentos. Deste modo, na fase final de aproximação, o sistema apresenta um erro de orientação mínimo, realizando pequenas compensações na posição. Nas equações 91 e 92 são constatáveis as expressões que definem o mecanismo auxiliar de movimentos.

$$v_{x,acost} = v_{x,acost}.kx_{park} \quad ; \quad 0 < kx_{park} \leq 1 \tag{91}$$

$$v_{y,acost} = v_{y,acost}.ky_{park} \quad ; \quad 0 < ky_{park} \leq 1 \tag{92}$$

Conforme verificável na Figura 103, a máquina de estados que define o comportamento de aproximação ao local de *park* é similar ao apresentado para o comportamento de *go to dock*, sendo apenas distinguível pela presença do mecanismo auxiliar de movimentos após a geração de movimentos.

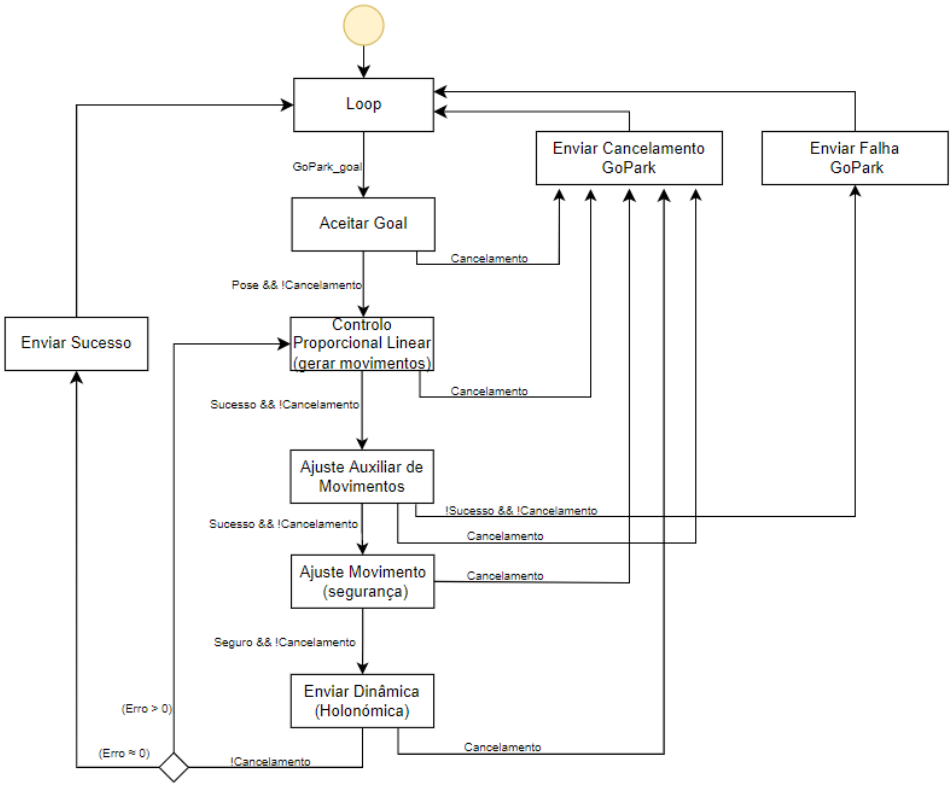


Figura 103: Máquina de estados que define o comportamento de aproximação ao local de *park* ou de carga.

8.5.3. COMPORTAMENTOS DE AFASTAMENTO: *RETURN FROM DOCK* E *RETURN FROM PARK*

As estratégias de afastamento aos locais de acostagem partilham os métodos de geração de movimentos e de execução do fluxo de controlo. A particularização dos comportamentos em causa deve-se à diferenciação do tipo de tarefa a executar, por parte do *task manager*, assim como à possibilidade de implementação de funcionalidades específicas a um determinado comportamento. Os módulos estabelecem comunicação bidirecional com o *task manager*, via ROS *action server* (Figura 104 e Figura 105), recebendo deste a posição desejada. Ademais, o *task manager* recebe informações periódicas do estado do processo, assim como o resultado da operação em causa.

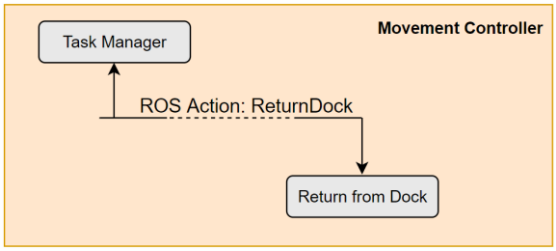


Figura 104: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *return from dock*.

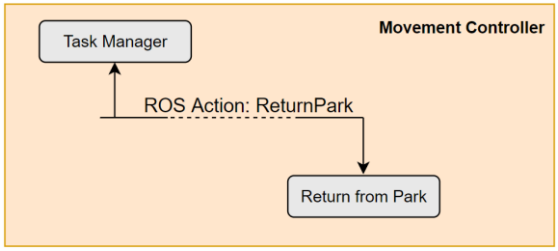


Figura 105: Comunicação via *ActionLib* entre o módulo *task manager* e a componente *return from park*.

A Tabela 12 apresenta a estrutura de mensagem relativa à comunicação bidirecional entre o *task manager* e os comportamentos de afastamento aos locais de acostagem. Na estratégia implementada ambos os comportamentos partilham o mesmo tipo de estrutura de mensagem, distinguindo-se pelo nome da *action*.

Tabela 12: Estrutura e descrição do tipo de mensagem que estabelece a comunicação entre o *task manager* e as componentes *return from dock* e *return from park*.

	Nome Variável	Tipo de Variável	Breve Descrição
Goal	targetPose	miar_msgs/GraphNode	Definição ponto auxiliar de manobra
	state_machine_index	int	Máquina de estados (do processo)
Feedback	result_value	int	Estado conclusão da tarefa
	result_description	string	Descrição do resultado da tarefa
Result	vx_robot	float	Velocidade longitudinal determinada
	vy_robot	float	Velocidade lateral determinada
	wrobot	float	Velocidade angular determinada
	state_machine_index	int	Index máquina de estados (processo)

Os métodos de geração de movimentos dos comportamentos apresentam similaridades aos módulos que definem o comportamento de aproximação aos locais de acostagem, diferindo na estratégia de variação do módulo da velocidade linear em função do erro de posição, como referido no subcapítulo Controlo e Definição das Manobras de Acostagem. Contrariamente ao requerido nas tarefas de aproximação, a velocidade linear do veículo deve aumentar de forma gradual, em função da proximidade ao local final desejado, sendo máxima neste ponto de manobra.

Como se verifica na Figura 106, a estratégia de controlo e fluxo de execução do método é idêntica à apresentada no módulo de aproximação aos locais de acostagem sendo, deste modo, análoga à que

define o *go to dock*, diferindo no algoritmo de controlo de geração de movimentos, implementado no controlador proporcional quadrático.

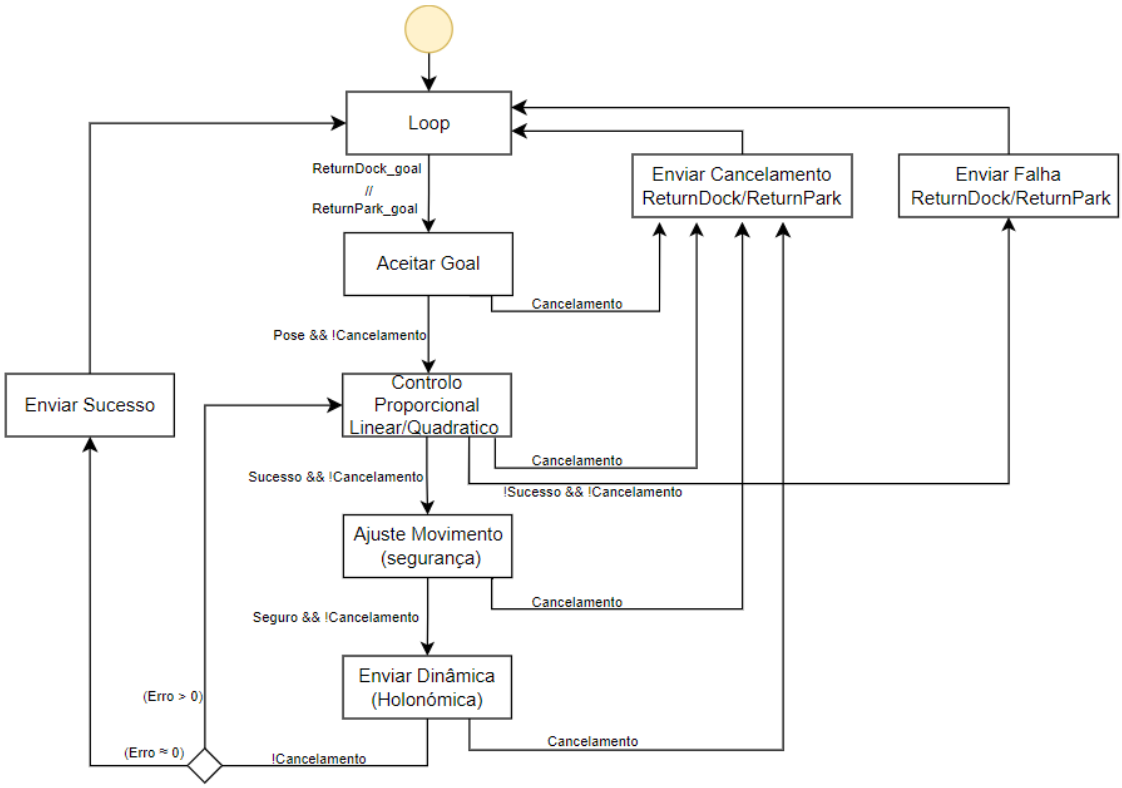


Figura 106: Máquina de estados que define o comportamento de afastamento ao local de acostagem e de *park* ou de carga.

9. TESTES E RESULTADOS

Neste capítulo são descritos e apresentados os testes, e respetivos resultados, referentes à verificação e validação dos comportamentos que definem a estratégia de controlo dos movimentos da plataforma móvel KMP 200 *OmniMove*. A análise dos comportamentos foi realizada em ambiente de simulação, considerando os requisitos e especificações do projeto. Para o efeito, foi considerado o cenário de simulação apresentado no subcapítulo Cenário de Simulação, replicando um ambiente industrial complexo, em *CoppeliaSim*. O método de controlo estabelece comunicação por intermédio do *middleware* ROS, possibilitando a verificação da coordenação das operações e validação dos comportamentos que estabelecem a tarefa requerida, conforme as condições do espaço.

Para o propósito, foram estabelecidas duas tarefas típicas: *load milling* (alimentação de uma máquina ferramenta fresadora) e *unload milling* (recolha do material manipulado da máquina ferramenta). Ambas as tarefas são solicitadas pelo *service manager* ao módulo controlador. Por último, considerando o requisito de navegar autonomamente em ambientes industriais dinâmicos, foi ainda averiguado o comportamento do sistema na presença de modelos dinâmicos no cenário de simulação.

9.1. TAREFA DE ALIMENTAÇÃO DA MÁQUINA FERRAMENTA: *LOAD MILLING*

Dado o propósito de cooperação e integração dos processos industriais por mecanismos autónomos, foi especificado a tarefa de *load milling*, que consiste na alimentação de uma máquina ferramenta, no caso do tipo fresadora. A tarefa foi especificada em 3 ações principais:

1. Navegar e acostar na estação de trabalho especificada como ponto de recolha do material a manipular (tendo como ponto de partida o local de *park*);

2. Navegar e acostar no local especificado como ponto de alimentação à máquina ferramenta;

3. Navegar e realizar a operação de estacionamento no local de *park* (coincidindo com o ponto de partida);

As ações especificadas são descritas pela coordenação dos comportamentos que compõem o módulo *Movement Controller*, por parte do *task manager*. De seguida é descrito com maior detalhe as ações que definem a tarefa de alimentação da máquina ferramenta.

9.1.1. Navegação e Acostagem ao Local de Trabalho (recolha material a manipular)

A ação de navegação e acostagem à estação de trabalho especificada como ponto de recolha do material a manipular é discriminada em 4 comportamentos distintos: *return from park*, *navigation*, *navigation omnidirectional* e *go to dock*. A orquestração do acionamento dos comportamentos encontra-se ilustrada na Figura 107.

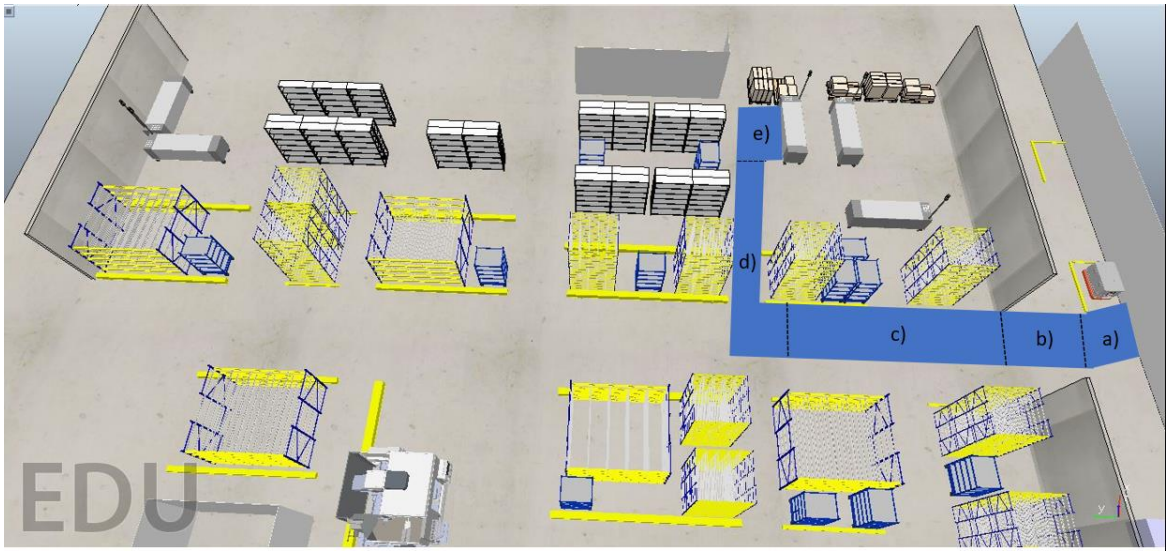
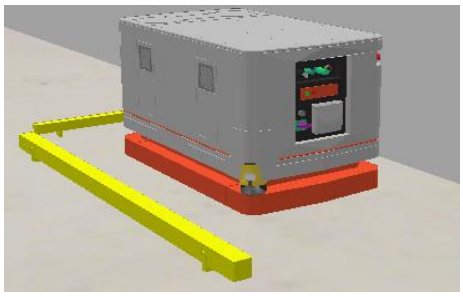


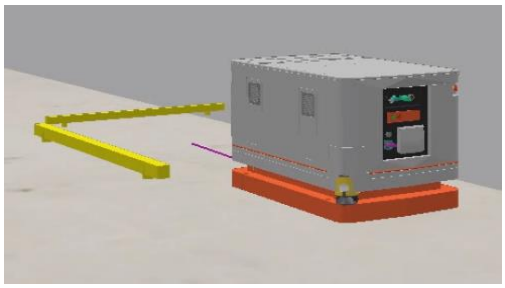
Figura 107: Representação dos comportamentos que definem a ação de navegação e acostagem ao local de trabalho especificado. a) *return from park*; b) *navigation omnidirectional*; c) *navigation*; d) *navigation omnidirectional*; e) *go to dock*.

Para a tarefa considerada, numa primeira fase, o sistema molda-se segundo o comportamento de afastamento ao local de *park* ou dos conectores de carga da bateria, este definido pelo procedimento *return form park* (Figura 108 a) e b)). Uma vez finalizada a operação, o veículo necessita de realizar o procedimento de correção da orientação de navegação e de posição para a rota estabelecida. Dada a presença de obstáculos próximos, a operação define-se pelo comportamento *navigation omnidirectional* (Figura 108 c)). O comportamento de *navigation* não é viável para o procedimento em causa devido a este ser caracterizado por uma maior margem de segurança (possibilitando a atuação precoce dos mecanismos de segurança) e por definir movimentos não holonómicos. Posteriormente, no corredor principal, o movimento da plataforma móvel define-se por movimentos não holonómicos e legíveis aos operadores humanos. Esta operação é moldada pelo comportamento *navigation* (Figura 108 d)). A cerca de 1 m de distância ao corredor de acesso à estação de trabalho, o movimento passa a ser estabelecido pelo comportamento *navigation omnidirectional* (Figura 108 e)). Na proximidade à estação de trabalho,

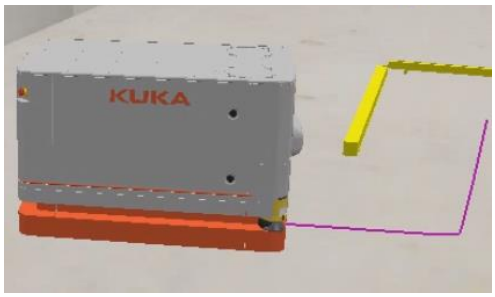
o movimento caracteriza-se pelo comportamento *go to dock*, corrigindo a posição e orientação do veículo de modo a maximizar o espaço de trabalho do braço robótico (Figura 108 f) e g)).



(a) *Pose* inicial (local de *park*)



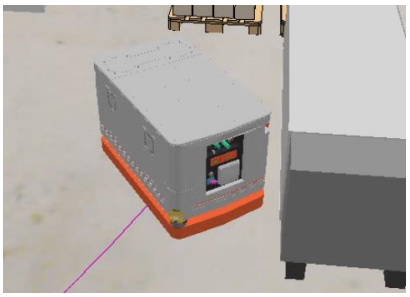
(b) Operação afastamento ao local de *park*



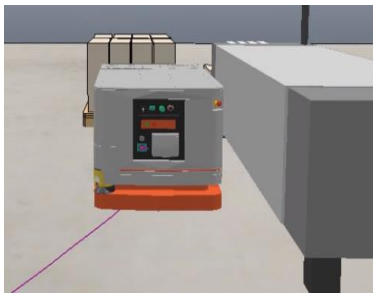
(c) Ajuste à rota por movimentos holonómicos

(d) Navegação legível pelo corredor principal

(e) Corredor de acesso ao local de trabalho



(f) Operação de acostagem ao local de trabalho



(g) *Pose* final (no local de trabalho especificado)

Figura 108: Sequência de comportamentos que definem a ação de navegação e acostagem ao primeiro local de trabalho. A tarefa pode ser verificada na integra em https://youtu.be/BYTV8281Dfc.

Como se verifica na sequência de operações presente na Figura 108, o método de controlo foi capaz de navegar no ambiente descrito, adaptando as ações de controlo conforme os dados recolhidos do espaço circundante e da informação recebida do pedido por parte do *Service Manager*. Como já referido,

a informação do *Service Manager* contém o percurso a ser definido, discretizado por *via points*, a indicação do tipo de comportamento mais adequado à navegação (*navigation* ou *navigation omni*) e a *pose* final desejada. Na sequência de operações apresentada, destaca-se o procedimento de entrada ao corredor de acesso ao local de trabalho, Figura 108 e), recorrendo à estratégia que define a trajetória segundo movimentos holonómicos. Devido à crítica proximidade de obstáculos (menor que 0,20 m) e à reduzida velocidade, a plataforma omnidirecional, segundo o método, apresenta alguma oscilação de navegação.

9.1.2. Navegação e Acostagem na Máquina Ferramenta (zona de alimentação)

A ação de navegação e acostagem na zona de alimentação da máquina ferramenta considerada caracteriza-se em 4 comportamentos: *return from dock*, *navigation omnidirectional*, *navigation* e *go to dock*. A gestão de comportamentos adotada encontra-se ilustrada na Figura 109, esta requerida pelo *Service Maneger* no pedido de execução da tarefa global (tarefa de *load milling*).

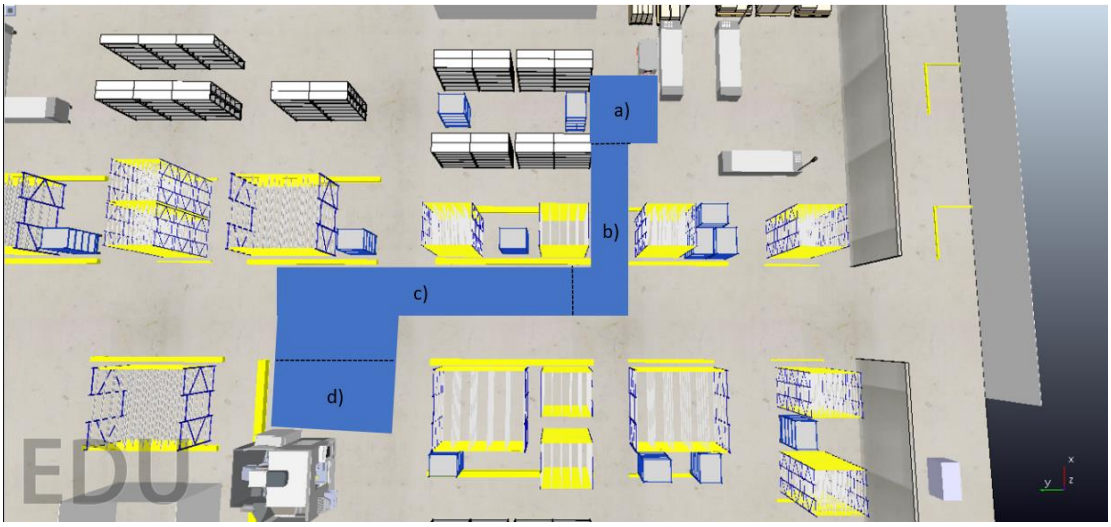
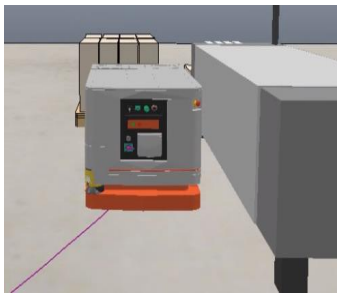


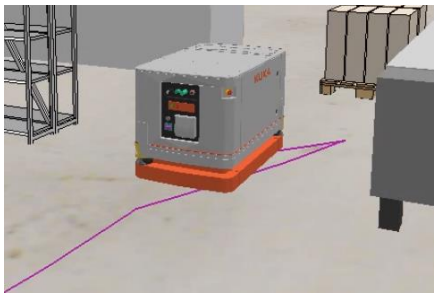
Figura 109: Representação da gestão de comportamentos que definem a ação de navegação e acostagem na zona de alimentação da máquina ferramenta. a) *return from dock*, b) *navigation omnidirectional*; c) *navigation*; d) *go to dock*.

Numa primeira fase, o sistema realiza a operação de afastamento ao local de acostagem. Esta operação é selecionada pelo *task manager* que verifica a última operação concluída, acionando o comportamento complementar, no caso o *return from dock* (Figura 110 a) e b)). Posteriormente, é estabelecido o percurso de acesso ao corredor principal. A rota é similar à estabelecida no acesso ao local de trabalho, sendo caracterizada pela critica proximidade aos elementos que constitui o cenário.

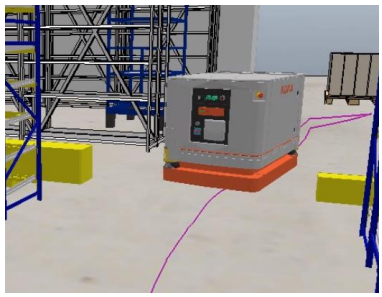
Dada a reduzida área de manobra, o comportamento é definido pela estratégia de navegação holonómica, *navigation omnidirectional* (Figura 110 c)). Uma vez realizada a manobra de acesso ao corredor principal, o sistema molda-se pela estratégia de movimentos não-holonómicos, definida pelo comportamento *navigation* (Figura 110 d)). Este comportamento mantém-se até que se verifique a proximidade com a zona de alimentação da máquina ferramenta. Este processo é viável dado o espaço útil de manobra no acesso ao local de acostagem. Por sua vez, na proximidade à máquina ferramenta, o sistema governa-se pela ação do comportamento *go to dock*, corrigindo a posição e orientação da plataforma omnidirecional de modo a viabilizar a operação do braço manipulador na operação de *machine tending* (Figura 110 e) e f)).



(a) *Pose* inicial da ação (local de trabalho)

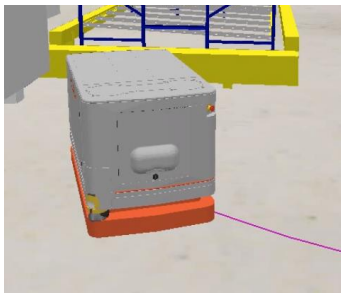


(b) Operação de afastamento ao local de *dock*



(c) Navegação holonómica no corredor de acesso

(d) Navegação legível pelo corredor principal



(e) Processo de *dock* na máquina ferramenta

(f) *Pose* final (na máquina ferramenta)

Figura 110: Sequencia de comportamentos que definem a ação de navegação e acostagem na máquina ferramenta (zona alimentação). A tarefa pode ser verificada na integra em https://youtu.be/BYTV8281Dfc.

Como constatável na Figura 110, para a sequência de operações considerada, o sistema consegue realizar os procedimentos previstos, adaptando as ações de controlo conforme as condições verificadas.

Identicamente ao verificado na primeira ação, a plataforma móvel apresenta ligeira oscilação na navegação na zona de acesso ao corredor principal, devido à reduzida zona de passagem (obstáculos próximos às laterais do corpo do veículo). Nesta ação, destaca-se a operação de acostagem à máquina ferramenta (zona de alimentação), Figura 110 (e), definida pelo comportamento *go to dock*. As ações de movimento revelaram-se eficazes para o procedimento de correção da posição e orientação do manipulador móvel, obtendo, deste modo, uma *pose* adequada à operação do braço robótico.

9.1.3. Navegação e Acostagem ao Local de *Park* (local de estacionamento)

A ação de navegação e acostar no local de *park* ou de estacionamento foi descrita em 4 comportamentos: *return from dock*, *navigation omnidirectional*, *navigation* e *go to park*. A orquestração dos comportamentos segue a representação contida na Figura 111.

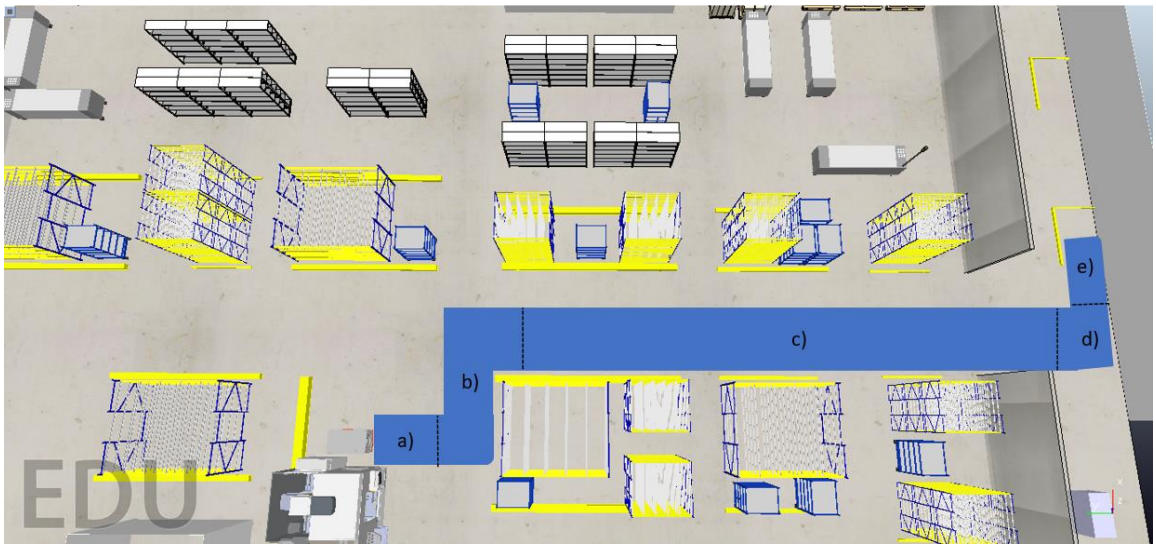


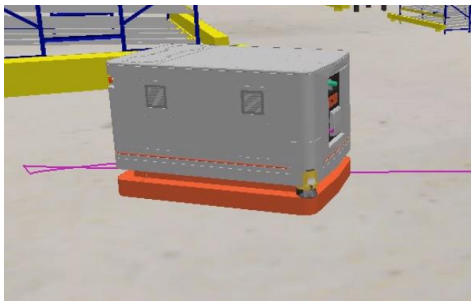
Figura 111: Representação da orquestração dos comportamentos que definem a ação de navegação e acostagem ao local de park. a) *return from dock*; b) *navigation omnidirectional*; c) *navigation*; d) *navigation omnidirectional*; e) *go to park*.

Similarmente ao indicado na ação anterior, numa primeira iteração, o sistema verifica a última operação da tarefa concluída, selecionando o comportamento complementar, no caso o *return from dock*, definindo, portanto, o comportamento de afastamento ao local de acostagem (Figura 112 a) e b)). Na ação considerada, a orientação da plataforma não é corrigida no comportamento de afastamento. Deste modo, assim que seja verificado uma distância aceitável à máquina ferramenta é designado o método de controlo de movimentos holonómicos, obtendo deste modo um maior grau de manobrabilidade e flexibilidade no ajuste da orientação. Desta forma, como verificável na Figura 112 (c),

a zona de manobra não coincide com a área do corredor principal, permitindo ainda acelerar a execução da manobra. Posteriormente, assim que o veículo esteja contido no corredor principal, o comportamento do sistema processa-se pela estratégia de movimentos não-holonómicos, estabelecida pelo comportamento *navigation* (Figura 112 d)). No caso considerado, este comportamento encontra-se ativo até ao ponto de cruzamento do corredor principal e a zona de parqueamento, contudo, como demonstrado na tarefa 9.2, este movimento poderia ser estabelecido até à periferia do local estabelecido para *park*. Neste caso, a distinção dos comportamentos incide na margem de segurança considerada à parede delimitadora do cenário. Nesta ação, como o processo é designado pelo comportamento *navigation omnidirectional* (Figura 112 e)), o sistema permite uma maior proximidade da plataforma móvel à parede delimitadora do cenário (ação tardia dos mecanismos de paragem de segurança). Por último, na proximidade ao local de estacionamento, o movimento caracteriza-se pela ação do comportamento *go to park*, estabelecendo uma trajetória de aproximação apropriada à conexão com os conectores de carga presentes no solo (Figura 112 f) e g)).

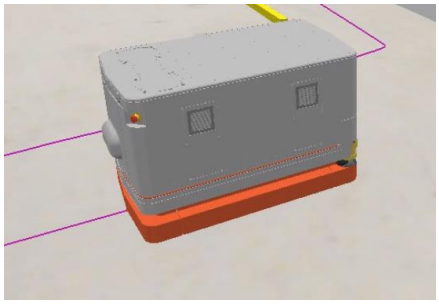
(a) *Pose* inicial da ação (máquina ferramenta)

(b) Afastamento à zona de alimentação

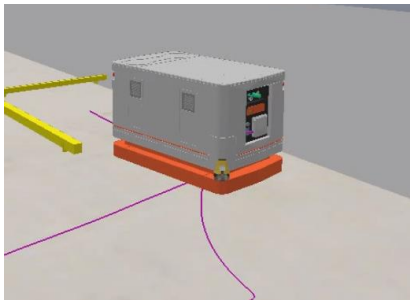


(c) Correção orientação navegação (holonómica)

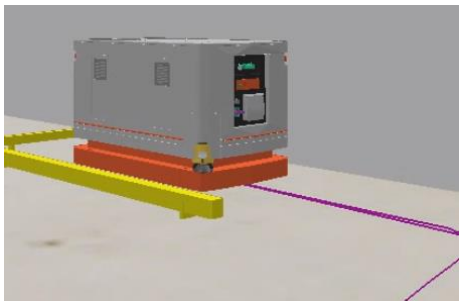
(d) Navegação legível pelo corredor principal



(e) Aproximação zona *park* (holonómico)



(f) Processo de *park* (correção *pose*)



(g) *Pose* final (no local de *park*)

Figura 112: Sequência de comportamentos que definem a ação de retorno ao local de estacionamento (*park*). A tarefa pode ser verificada na integra em https://youtu.be/BYTV8281Dfc.

Como verificável na sequência de comportamentos considerada, o sistema conseguiu realizar a tarefa de retorno ao local de *park* ou de carregamento com sucesso. Dentro dos comportamentos adotados, destaca-se a capacidade de ajuste da orientação no acesso ao corredor principal, Figura 112 c), e da definição da trajetória de aproximação ao local de *park*, Figura 112 f), priorizando, numa primeira

instância, a correção da orientação. Esta prevalência deve-se à atuação do mecanismo auxiliar de movimentos, indicado no subcapítulo Comportamento de Aproximação aos Locais de Park: *Go to Park*.

Para a tarefa considerada, uma vez concluída a ação do comportamento *go to park*, o *task manager* retorna ao *Service Manager* a indicação de término da tarefa em estado de sucesso (cumpriu as 3 ações previstas). Desse modo, após o envio da indicação, o sistema fica apto à receção de um novo pedido ou serviço por parte do servidor.

9.2. Tarefa de Recolha e Armazenamento dos Produtos Manipulados: *Unload Milling*

A tarefa de *unload milling* corresponde à execução das ações complementares à tarefa anterior retratada. Esta consiste na recolha do produto, considerado como já manipulado pela máquina ferramenta, e armazená-lo na zona de trabalho especificada para o efeito. Para o caso em estudo, considera-se a tarefa como concluída assim que a plataforma móvel regresse ao ponto de partida da tarefa (local de *park*). Para o efeito, a tarefa foi discriminada em 3 ações principais:

1. Navegar e acostar no local especificado como ponto de recolha do produto manipulado, na máquina ferramenta;
2. Navegar e acostar na estação de trabalho especificada como ponto de entrega do produto manipulado;
3. Navegar e realizar a operação de estacionamento no local de *park* (coincidindo com o ponto de partida);

As ações são interpretadas pelo *task manager*, gerindo a coordenação dos comportamentos conforme os objetivos intermédios. O comportamento do sistema para as subsecivas ações é caracterizado com maior detalhe nos subcapítulos subsequentes.

9.2.1. Navegação e Acostagem na Máquina Ferramenta (zona de recolha)

A ação de navegação e acostagem na zona de recolha do material manipulado da máquina ferramenta caracteriza-se por 4 comportamentos: *return from park*, *navigation omnidirectional*, *navigation* e *go to dock*. A coordenação dos comportamentos segue a representação presente na Figura 113.

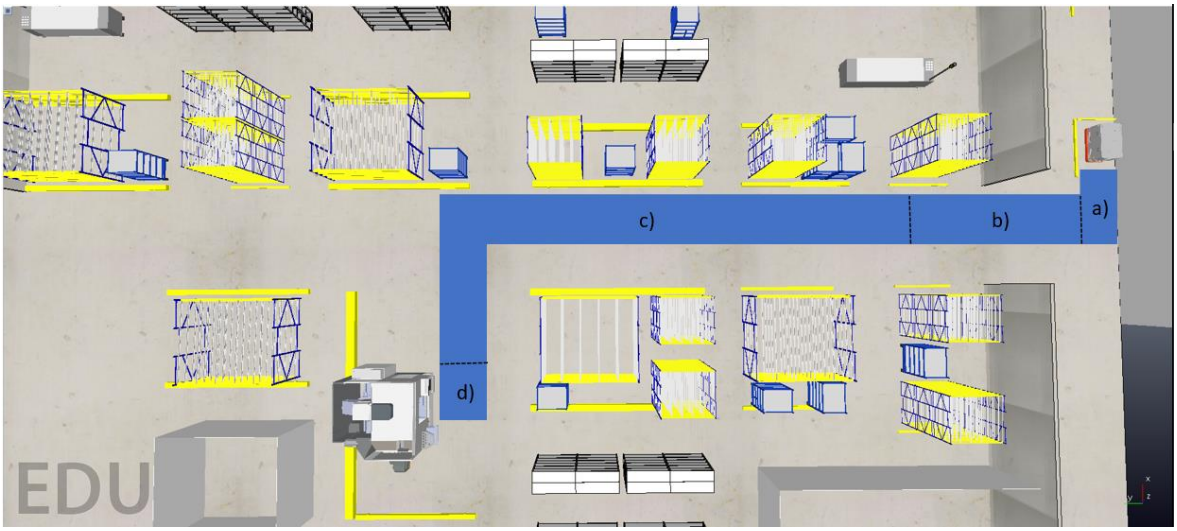
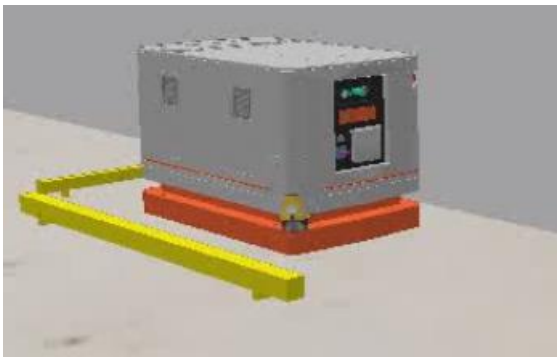
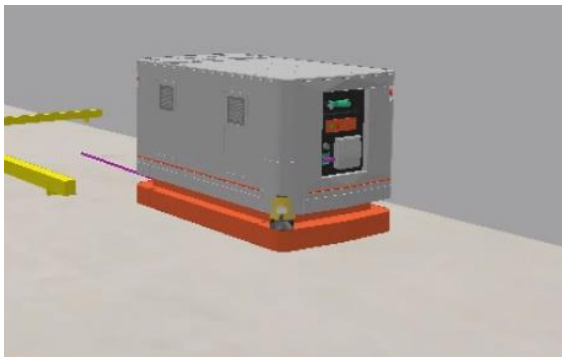


Figura 113: Representação da gestão de comportamentos considerada que define a ação de navegação e acostagem na zona de recolha da máquina ferramenta. a) *return from park*; b) *navigation omnidirectional*; c) *navigation*; d) *go to dock*.

Presumindo que o sistema se encontre em *park*, numa primeira fase, o *task manager* seleciona a estratégia de afastamento ao local de *park*, esta definida pelo *return from park* (Figura 114 a) e b)). Posteriormente, o sistema inicia a operação de navegação para o local de acostagem. Para o efeito, numa primeira iteração, o veículo realiza a manobra de acesso ao corredor principal, esta estabelecida pelo comportamento *navigation omnidirectional* (Figura 114 c)). Embora a manobra possa ser estabelecida pelo comportamento *navigation*, a crítica proximidade à parede limitadora faria acionar os mecanismos de segurança, imobilizando a plataforma móvel. Contudo, uma vez no corredor principal, o comportamento molda-se segundo movimentos não-holonómicos, estabelecido pela componente *navigation* (Figura 114 d)). Similarmente ao verificado na tarefa de *load*, devido ao amplo espaço útil de manobra na zona da máquina ferramenta, o comportamento *navigation* mantém-se até à proximidade da zona de *unload* da máquina fresadora. Subsequentemente, na proximidade ao local de manipulação, o comportamento do sistema rege-se pelo mecanismo *go to dock*, corrigindo a posição e a orientação da plataforma omnidirecional de modo a viabilizar a posterior operação de *machine tending* (Figura 114 e) e f)).



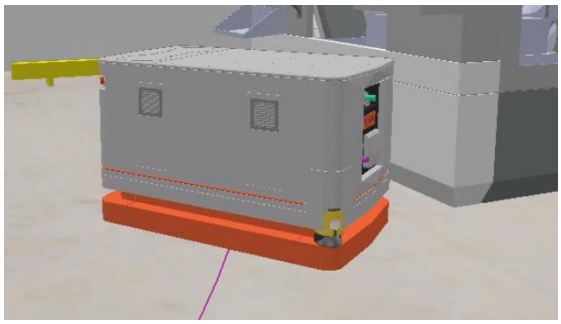
(a) *Pose* inicial (local de *park*)



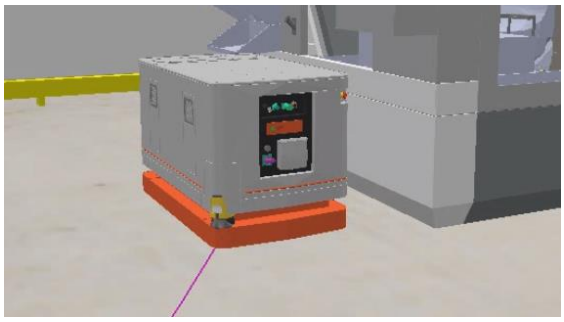
(b) Operação afastamento ao local de *park*

(c) Acesso ao corredor principal (holonómico)

(d) Navegação legível pelo corredor principal



(e) Manobra de acostagem à máquina ferramenta



(f) *Pose* final (na máquina ferramenta)

Figura 114: Sequencia de comportamentos que definem a ação de navegação e acostagem na máquina ferramenta (zona de recolha). A tarefa pode ser verificada na integra em https://youtu.be/AzKsFf4vPHY.

A sequência de comportamentos presente na Figura 114 apresenta as principais operações que definem a tarefa proposta. Como se verifica, pelo conjunto de comportamentos definido, o sistema foi capaz de navegar e realizar o processo de acostagem à zona de *unload*, na máquina ferramenta. Esta última manobra, Figura 114 (e), revela ser uma operação de caracter crítico, dado que além de recorrer às capacidades holonómicas da plataforma móvel, também deve ser um processo preciso. Destaca-se que a *pose* final do manipulador móvel define a área de trabalho do braço manipulador, influenciando diretamente a operação de *machine tending*.

9.2.2. Navegação e Acostagem ao Local de Trabalho (armazenar material manipulado)

A ação de navegação e acostagem à estação de trabalho relativa ao ponto de armazenamento do material já manipulado é descrita em 4 componentes: *return from dock*, *omnidirectional navigation*, *navigation* e *go to dock*. A gestão de comportamentos adotada segue a ilustração presente na Figura 115.

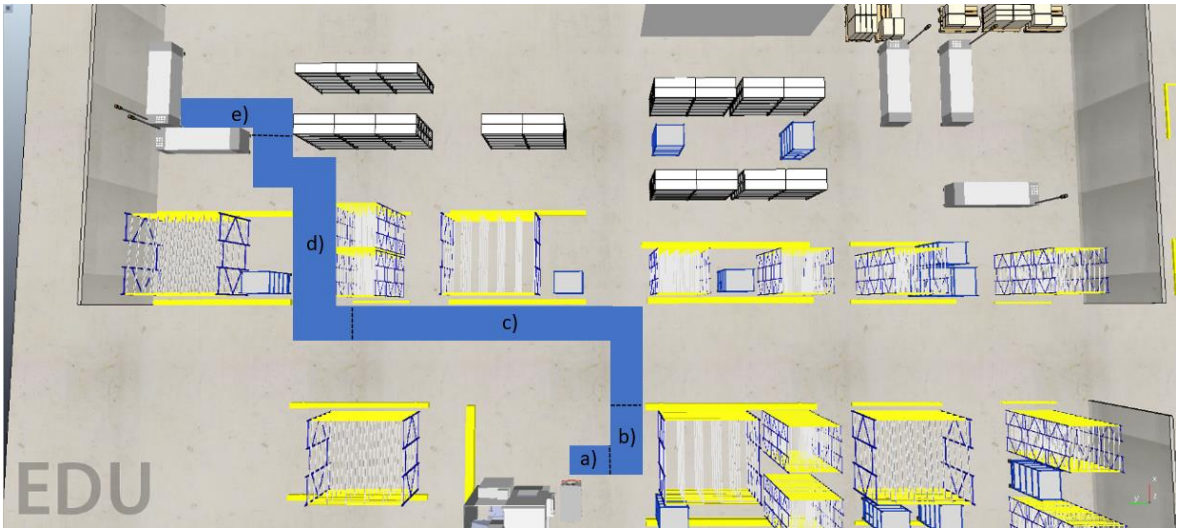
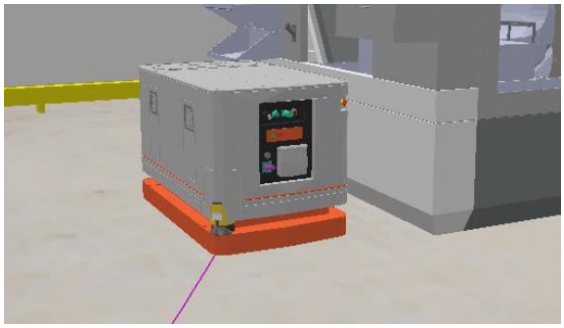


Figura 115:Representação da gestão de comportamentos adotada que definem a ação de navegação e acostagem ao local de acostagem. a) *return from dock*; b) *navigation omnidirectional*; c) *navigation*; d) *navigation omnidirectional*; e) *go to dock*.

Após a conclusão da operação de *machine tending*, o *task manager* verifica a última componente de aproximação ativa, selecionando o comportamento complementar para a operação de afastamento, no caso o *return from dock* (Figura 116 a) e b)). Dada a *pose* final da manobra, Figura 116 b), o sistema apresenta uma orientação quase coincidente com a da rota pretendida, minimizando a correção da orientação de navegação. Contudo, para o caso em estudo, a manobra de acesso ao corredor principal foi estabelecida pelo comportamento *navigation omnidirectional* (Figura 116 c)), oferecendo uma maior flexibilidade e manobrabilidade para os desvios de possíveis obstáculos. Por sua vez, no corredor principal de navegação, o sistema molda-se pelo comportamento *navigation* (Figura 116 d)), definindo, portanto, movimentos não-holonómicos. Este comportamento mantém-se ativo até à proximidade da via de acesso ao local estabelecido como ponto de entrega do material manipulado, passando, posteriormente, a ser regido pela estratégia de navegação holonómica, *navigation omnidirectional* (Figura 116 e)). O comportamento de navegação holonómica revela ser a estratégia adequada para o efeito dado o reduzido espaço de manobra. Por último, o sistema realiza a operação de aproximação ao local de acostagem, delineada pelo comportamento *go to dock*, corrigindo a *pose* da plataforma móvel (Figura 116 f) e g)).



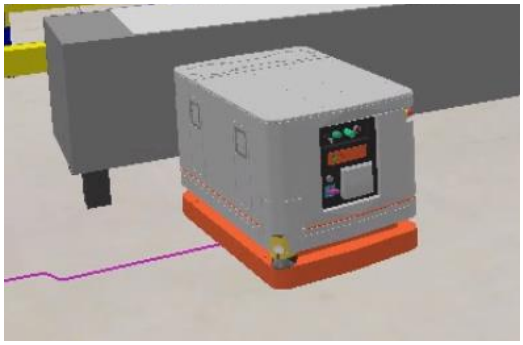
(a) *Pose* inicial da ação (máquina ferramenta)

(b) Afastamento à zona de recolha (máquina)

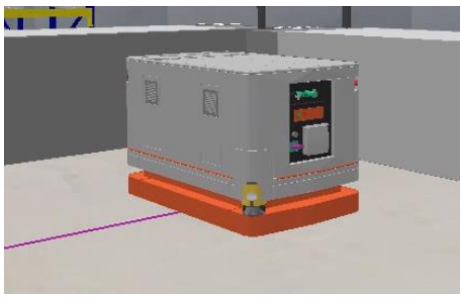
(c) Acesso ao corredor principal (holonómica)

(d) Navegação legível pelo corredor principal

(e) Corredor de acesso ao local de trabalho



(f) Operação de acostagem ao local de trabalho



(g) *Pose* final (no local de trabalho requerido)

Figura 116: Sequência de comportamentos que definem a ação de navegação e acostagem ao local de trabalho requerido. A tarefa pode ser verificada na integra em https://youtu.be/AzKsFf4vPHY.

Conforme verificável na Figura 116, dada a sequência de comportamentos proposta para a ação, o sistema foi capaz de realizar os procedimentos previstos, ajustando as ações de controlo conforme as

condições de navegação e manobra verificadas. Da sequência de comportamentos apresentada, destaca-se a manobra de entrada à via de acesso ao local de trabalho, Figura 116 e), bem como a operação de correção da *pose* da plataforma móvel, sendo necessário compensar a trajetória da manobra devido à presença da mesa de apoio na estação de trabalho, Figura 116 f).

9.2.3. NAVEGAÇÃO E ACOSTAGEM AO LOCAL DE *PARK* (LOCAL DE ESTACIONAMENTO)

A ação de navegação e acostagem ao local de *park* ou de estacionamento caracteriza-se pela ação de 4 comportamentos: *return from dock*, navigation *omnidirectional*, *navigation* e *go to park*. A orquestração dos comportamentos segue a ilustração presente na Figura 117.

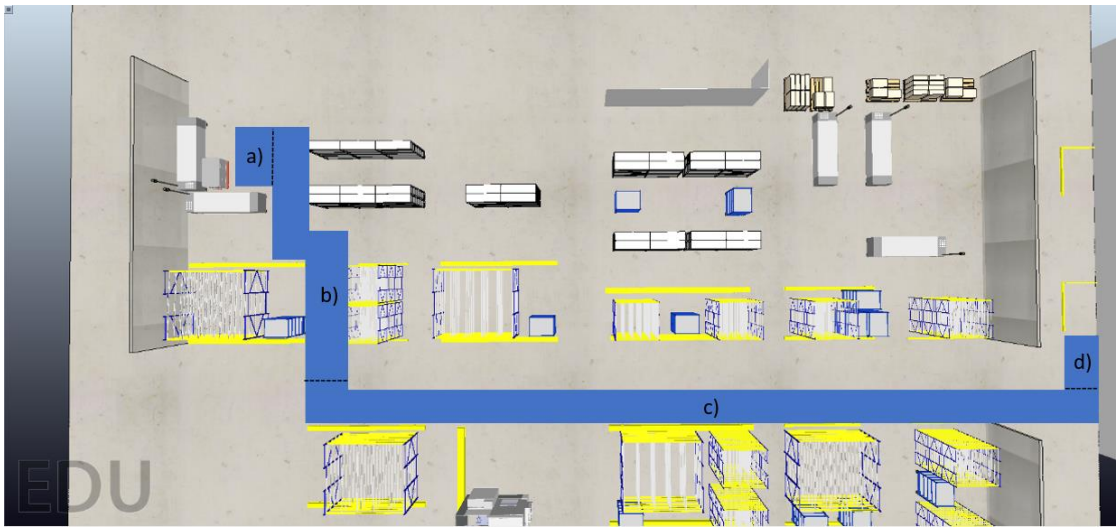
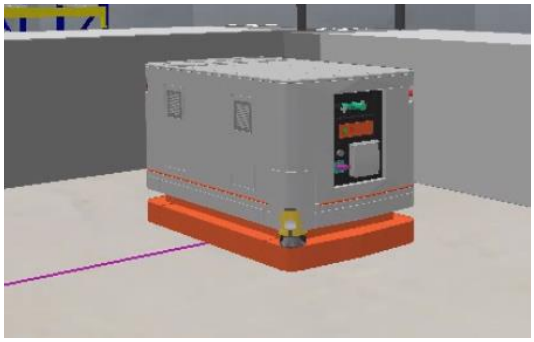


Figura 117: Representação da orquestração dos comportamentos que definem a ação de navegação e acostagem ao local de park. a) *return from dock*; b) *navigation omnidirectional*; c) *navigation*; d) *go to park*.

A ação de retorno ao local de *park* é similar à apresentada na tarefa anterior, subcapítulo 9.1.3, diferenciando o local de início da ação e o comportamento adotado para a aproximação à zona de *park*. Para o efeito, numa primeira iteração, o sistema procede à operação de afastamento ao local de trabalho, definida pelo comportamento *return from dock* (Figura 118 a) e b)). Uma vez que a orientação da plataforma móvel não é compensada na manobra de afastamento (não foi requerido), a componente de navegação deve assegurar que esta seja corrigida, de modo a coincidir com a trajetória planeada, bem como evitar colisões com obstáculos presentes. Dado o reduzido espaço de manobra, este procedimento deve ser estabelecido pelo método de controlo de movimentos holonómicos, definido pelo comportamento *navigation omnidirectional* (Figura 118 c)). Por sua vez, assim que a manobra de acesso

ao corredor principal seja estabelecida, o sistema molda-se segundo a estratégia de movimentos não-holonómicos ou legíveis, gerida pela componente *navigation* (Figura 118 d)). Contrariamente ao requerido para a tarefa anterior, este comportamento mantém-se ativo até à proximidade com a zona de início de manobra de *park*., demonstrando a flexibilidade da solução. Contudo, destaca-se que, para esta situação, a manobra de acostagem deve ser executada a uma maior distância à parede limitadora (devido à atuação dos mecanismos de segurança). Por último, dá-se o comportamento de *go to park*, estabelecendo uma trajetória adequada à aproximação aos conectores de carga presentes no solo (Figura 118 e) e f)).

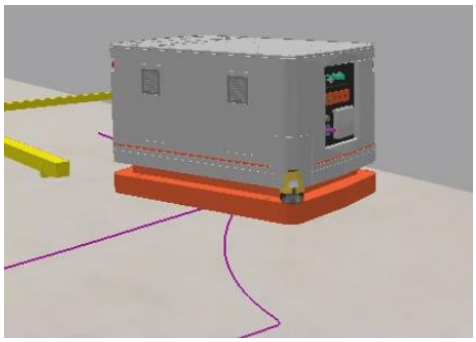


(a) *Pose* inicial da ação (local de trabalho)

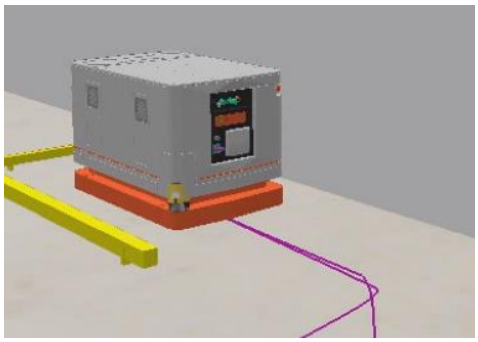
(b) Operação de afastamento ao local de *dock*

(c) Correção orientação navegação (holonómica)

(d) Navegação legível pelo corredor principal



(e) Processo de *park* (correção *pose*)



(f) *Pose* final (no local de *park*)

Figura 118: Sequência de comportamentos que definem a ação de navegação e acostagem ao local de estacionamento (*park*). A tarefa pode ser verificada na integra em https://youtu.be/AzKsFf4vPHY.

Dada a sequência de comportamentos indicada, conclui-se que o sistema conseguir realizar a ação de retorno ao local de *park* com sucesso. Dos comportamentos definidos destaca-se a capacidade de ajuste da orientação na manobra de saída ao local de trabalho (ponto de entrega do produto manipulado), Figura 118 c), bem como a manobra de aproximação aos conectores de carregamento da bateria, estabelecido pelo comportamento *go to park*. Similarmente ao indicado para a tarefa anterior, no subcapítulo 9.1.3, a estratégia de aproximação prioriza, numa primeira instância da manobra, a correção da orientação em detrimento da compensação do erro de posicionamento, estabelecendo o tipo de manobra aconselhada à conexão aos conectores de carga presentes no solo.

Uma vez concluída a operação de aproximação ao local de *park*, dá-se por concluída a tarefa de *Unload Milling* (definida pelas 3 ações estudadas). Dessa forma, após a recessão da indicação do término da ação, o *task manager* retorna o estado de sucesso ao *Service Manager*, transpondo a disponibilidade do sistema à execução de uma nova tarefa ou serviço.

9.3. Comportamento em Ambientes Dinâmicos

Considerando o objetivo de navegar autonomamente em ambientes industriais dinâmicos, de forma segura e consciente da presença de operadores humanos, foi averiguado o comportamento do sistema na execução das tarefas anteriormente definidas, considerando, neste caso, a presença de modelos dinâmicos no cenário de simulação.

Para o propósito, em ambiente de simulação, foi analisado o comportamento do sistema tanto na presença de modelos dinâmicos similares a seres humanos, bem como na presença de um outro modelo dinâmico da plataforma móvel.

9.3.1. Navegação em Ambientes Partilhados com Operadores Humanos

Dado ser espetável a presença de obstáculos dinâmicos durante a navegação no ambiente industrial, podendo esta ser descrita pelas componentes de *navigation* e de *navigation omnidirectional*, é dado especial relevância à análise comportamental do sistema nas situações em que o mesmo é moldado por essas componentes.

Para efeitos de teste, foi requerido ao sistema a realização da tarefa descrita em 9.1 no ambiente apresentado anteriormente, diferenciando unicamente pela presença de modelos dinâmicos similares a seres humanos, disponíveis na biblioteca do *software* de simulação *CoppeliaSim*. Dos modelos

disponíveis, foi selecionado o *Walking Bill* (Figura 119), sendo capaz de navegar aleatoriamente no cenário de simulação. Apesar de limitado, o modelo é capaz de evitar colisões com os obstáculos presentes no cenário (para e realiza uma rotação sobre si próprio até determinar uma trajetória linear desimpedida).

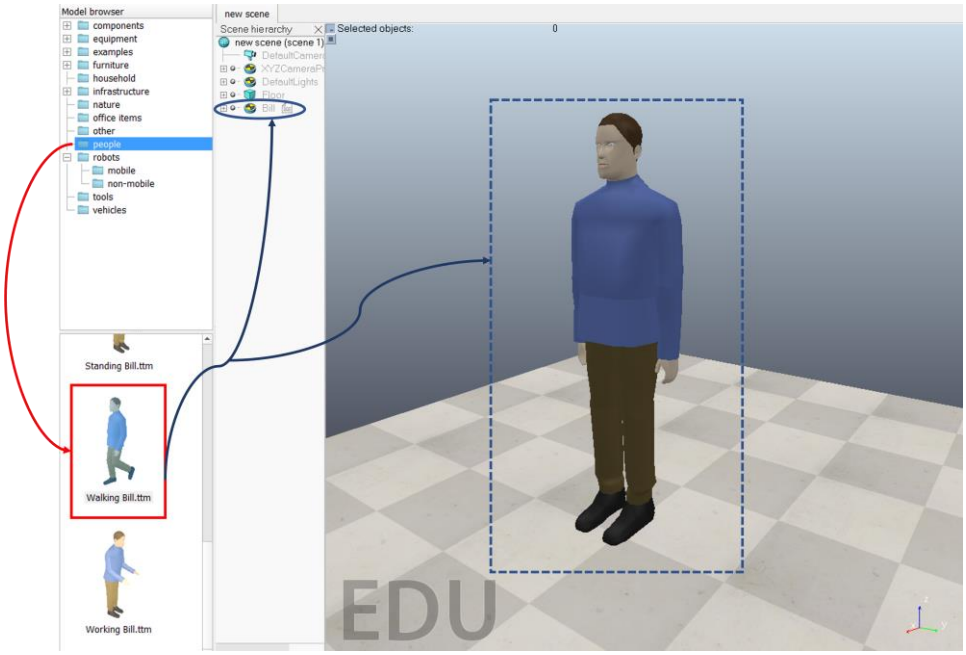


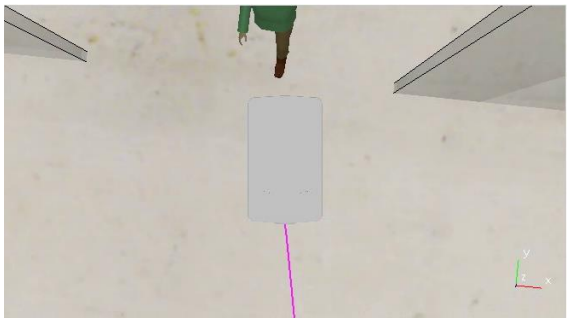
Figura 119: Modelo dinâmico do ser humano selecionado nos testes realizados, já disponível no *CoppeliaSim* (*Walking Bill*).

a) Acesso ao corredor principal:

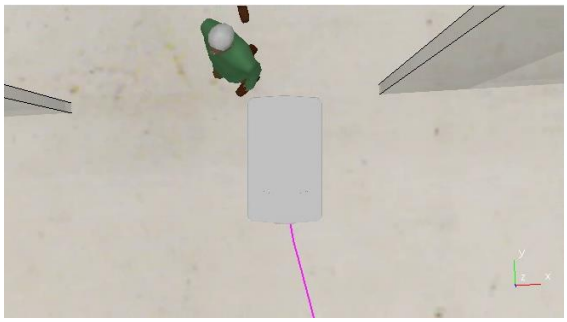
Na fase inicial da tarefa, o sistema foi moldado pelos comportamentos de afastamento ao local de *park* e *navigation omnidirectional*. Conforme referido anteriormente, a presença de obstáculos dinâmicos deve ser evitada (devido à reduzida margem de segurança). Deste modo, nesta iteração deu-se especial relevância ao acesso ao corredor principal.

Nesta validação, verificou-se a presença de um ser humano na rota de navegação do veículo (Figura 120 a)). Dada a circunstância, o sistema iniciou a ação de desvio pela direita do veículo (esquerda do operador). Embora o comportamento holonómico, devido à rapidez de aproximação do operador humano ao veículo, foi acionado o mecanismo de paragem de segurança (proximidade critica ao veículo), (Figura 120 b)). Uma vez verificada as condições de segurança (afastamento do operador humano) o sistema iniciou a ação de correção da trajetória até à presença de um segundo ser humano no espaço de navegação (Figura 120 c)). Neste segundo caso, o veículo procedeu ao desvio do obstáculo pela sua esquerda, tentando o contornar. Contudo, conforme ilustrado na Figura 120 d), o operador humano

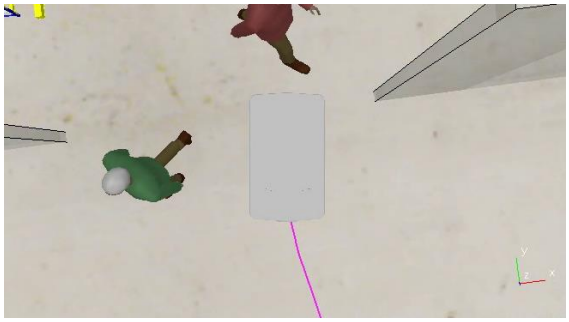
permaneceu imóvel, obstruindo a trajetória inicial de fuga. A proximidade crítica ao veículo origina a ação do mecanismo de paragem de segurança, permanecendo ativo durante a presença do operador humano. A ação dos mecanismos de paragem deveu-se à rapidez de aproximação do operador humano ao próprio manipulador móvel, sendo, dessa forma, incapaz de realizar as operações de fuga em tempo útil. Estas ações poderiam ser minimizadas se a parametrização do sistema fosse mais reativa, contudo, nessa condição, os movimentos do veículo seriam repentinos ("mais bruscos") e imprevisíveis. Após a libertação do espaço, o sistema efetuou a ação de correção de trajetória (Figura 120 e)), retomando a trajetória padrão (Figura 120 f)). Destaca-se que os resultados apresentados foram condições singulares de operação, dada a imprevisibilidade dos modelos dinâmicos dos seres humanos.



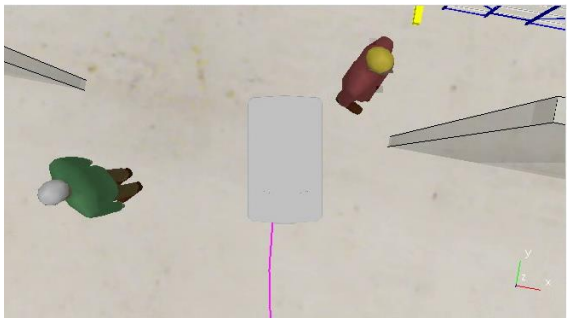
(a) Aproximação de um operador humano



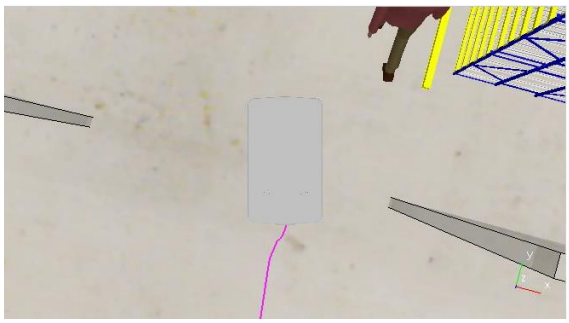
(b) Ação de desvio e paragem (risco colisão)



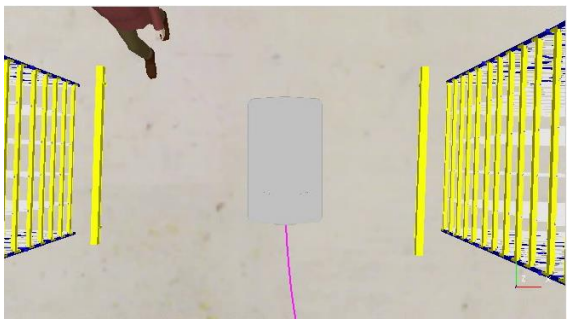
(c) Presença de outro operador humano



(d) Ação de paragem (proximidade crítica)



(e) Correção de trajetória



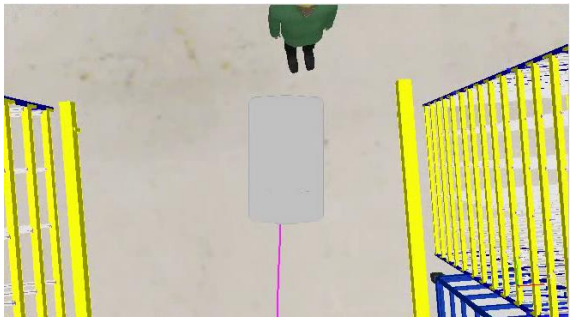
(f) Operação normalizada (rota desimpedida)

Figura 120: Sequência de comportamentos do sistema em zona congestionada. A tarefa pode ser verificada na integra em https://youtu.be/qrymVdjKcBA.

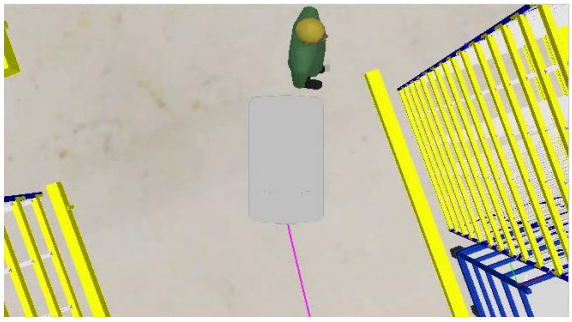
b) Acesso ao local de trabalho (corredor secundário de acesso à zona de recolha):

A operação de entrada ao corredor de acesso ao local de trabalho referente à zona de recolha dos materiais a manipular é dos processos mais delicadas da tarefa requerida. Esta operação foi moldada pelo comportamento *navigation omnidirectional*, garantindo maior manobrabilidade no acesso ao mesmo, tendo ainda a flexibilidade de realizar ajustes à manobra no caso de serem detetados obstáculos.

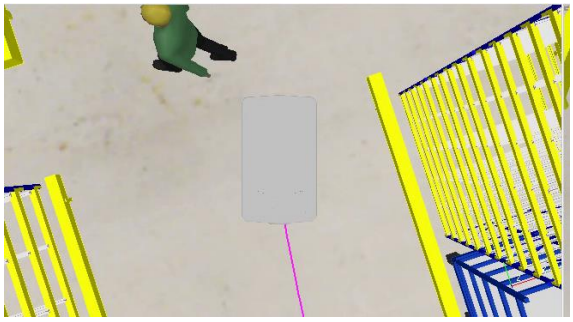
Nesta simulação, o sistema detetou a presença de um obstáculo na zona de entrada ao corredor de acesso à zona de trabalho (Figura 121 a)). O sistema realizou a ação de desvio de obstáculos, realizando uma rotação centrada no operador humano, tendo como objetivo encontrar uma possível trajetória desimpedida. Contudo, a disposição do operador humano impossibilitou o acesso ao corredor, e, devido à proximidade crítica, o mecanismo de paragem de segurança foi ativado, imobilizando o veículo (Figura 121 b)). Após verificar a desobstrução do acesso (Figura 121 c)), o sistema retomou a operação de navegação, corrigindo a trajetória. Por sua vez, após a manobra de compensação, foi detetado um novo obstáculo (operador humano) em trajetória de colisão com a do veículo (Figura 121 d)), iniciando, deste modo, uma ação de desvio. Contudo, devido à rapidez de aproximação e ao reduzido espaço de fuga, o sistema foi incapaz de se desviar do obstáculo, sendo acionado o mecanismo de paragem (Figura 121 e)). Uma vez verificada a libertação do espaço, o sistema retomou a operação, realizando ajustes à trajetória de acesso ao corredor. Destaca-se o facto de o movimento ser definido por movimentos holonómicos, sendo, dessa forma, capaz de estabelecer a manobra ilustrada entre as sequências Figura 121 e) e Figura 121 f).



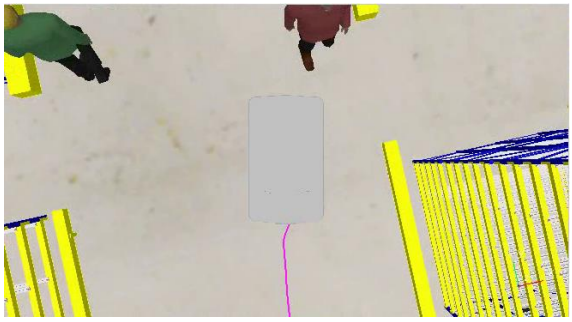
(a) Aproximação de um operador humano



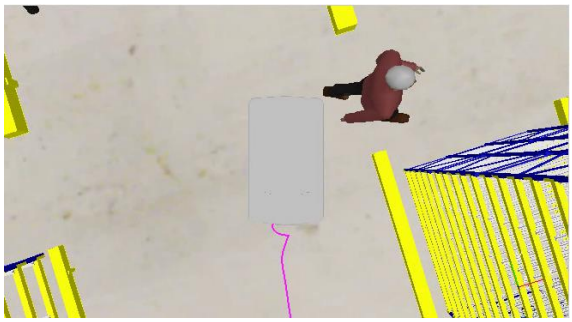
(b) Ação de desvio e paragem segurança



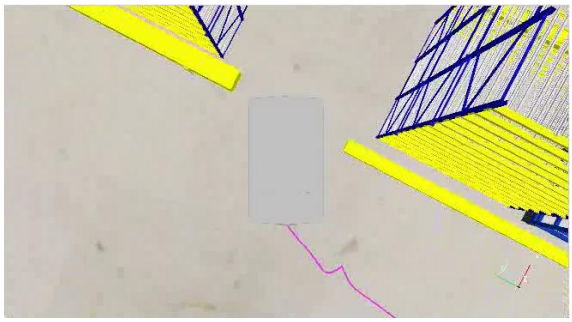
(c) Desobstrução da zona de acesso



(d) Aproximação de outro operador humano



(e) Ação de desvio e paragem de segurança



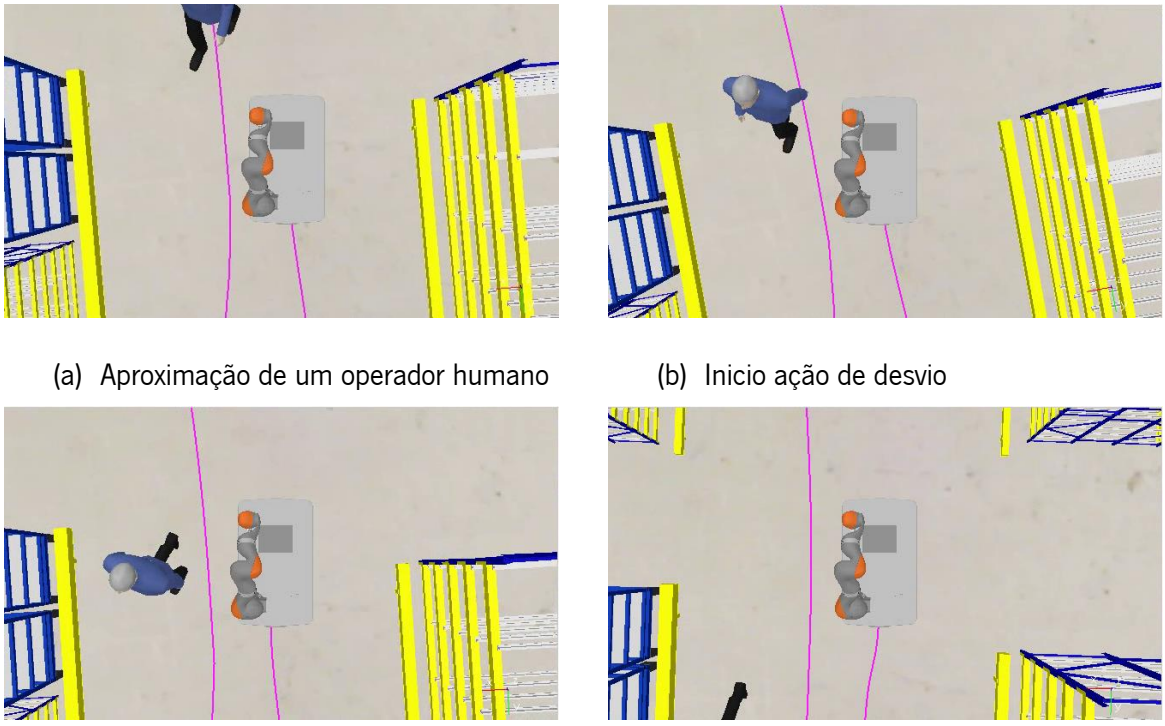
(f) Correção da trajetória de acesso

Figura 121:Sequência de ações de desvio e paragem de segurança durante a entrada ao corredor de acesso ao local de trabalho. A tarefa pode ser verificada na integra em https://youtu.be/qrymVdjKcBA.

c) Navegação no corredor principal:

A operação de navegação pelo corredor principal foi estabelecida pelo comportamento *navigation*, sendo caracterizável pela geração de movimentos legíveis aos operadores humanos. Deste modo, nestas condições, a ação de fuga da plataforma omnidirecional deve assemelhar-se à manobra de fuga de um veículo de direção diferencial. Dado que o comportamento verificado na Figura 122 foi à posteriori das simulações anteriormente apresentadas, destaca-se a presença do braço manipulador na plataforma móvel, constituindo o manipulador móvel KMR iiwa.

Nesta validação, na fase de retorno ao local de *park*, foi detetada a presença de um obstáculo durante a navegação pelo corredor principal (Figura 122 a)). Para o efeito, o veículo iniciou a ação de desvio ao operador humano, definindo uma curva ligeira à sua direita (Figura 122 b)). Esta ação permitiu o cruzamento seguro entre o manipulador móvel e o operador humano (Figura 122 c)), evitando, consequentemente, a ação do mecanismo de paragem. Após passagem pelo operador humano, o sistema retornou ao estado de operação normalizado, corrigindo, para o efeito, a trajetória de navegação (Figura 122 d)).



(a) Aproximação de um operador humano (b) Inicio ação de desvio

(c) Manutenção distância segurança (d) Operação normalizada

Figura 122: Sequência de ações de desvio durante a navegação no corredor principal (retorno ao local de *park*). A tarefa pode ser verificada na integra em https://youtu.be/SUva9dcU9Rg.

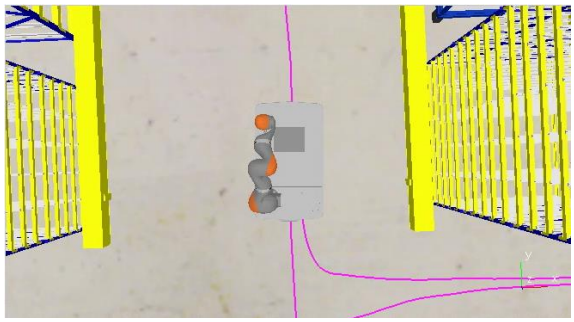
9.3.2. Navegação em Ambiente Composto por Múltiplas Plataformas

Considerando a possibilidade de partilha do espaço de navegação por múltiplos agentes dinâmicos, este subcapítulo tem como finalidade a averiguação do comportamento do sistema na presença de outro modelo dinâmico do manipulador móvel.

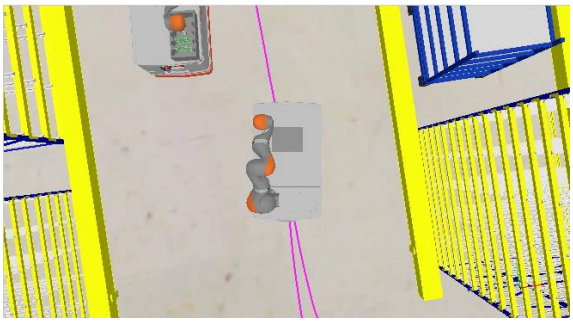
Para efeitos de teste, foi analisado o comportamento do manipulador móvel destacado na sequência temporal da Figura 123 a), sendo requerido a realização da tarefa descrita em 9.1. Ao cenário descrito foi adicionado um segundo modelo dinâmico do manipulador móvel (ver subcapítulo 8.1.3), este sem

tarefa atribuída, apresentando apenas o comportamento de navegação pelo corredor principal (tendo como finalidade a geração de tráfego).

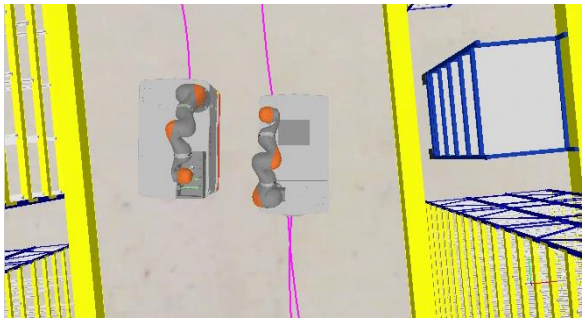
A sequência de ações apresentada na Figura 123 refere-se à operação de retorno do manipulador móvel ao local de *park*, esta descrita pelo comportamento *navigation*. Numa primeira fase, o sistema apresentou uma navegação normalizada, característica de um espaço amplo e desimpedido (Figura 123 a)). Na proximidade ao segundo manipulador móvel, o sistema iniciou a ação de desvio, estabelecendo uma curva ligeira pela direita (Figura 123 b)). Este procedimento permitiu a manutenção de uma distância superior à ação do mecanismo de paragem de segurança (Figura 123 c)). Uma vez verificado o cruzamento, o sistema retornou à trajetória de navegação padrão (Figura 123 d)). Destaca-se que, apesar de não ter sido atribuído a execução de uma tarefa, o segundo manipulador móvel foi controlado pela mesma estratégia de controlo do manipulador de destaque (*navigation*), capacitando-o do comportamento de desvio de obstáculos.



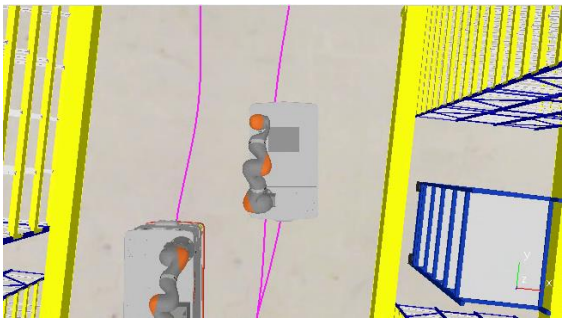
(a) Navegação corredor principal



(b) Início ação de desvio



(c) Manutenção distância segurança



(d) Regresso operação normalizada

Figura 123: Sequência de ações de desvio na presença de outro modelo dinâmico durante a navegação no corredor principal. A tarefa pode ser verificada na integra em https://youtu.be/xqOwibJajHs.

10. CONCLUSÕES E TRABALHO FUTURO

Neste capítulo é realizada uma breve discussão do enquadramento e requisitos do projeto, abordando aspetos como a formulação e estruturação da solução proposta bem como a viabilidade da solução face aos resultados apresentados. Por sua vez, são também debatidas algumas perspetivas a serem aprofundadas e desenvolvidas como trabalho futuro.

10.1. CONCLUSÃO

A rápida conectividade e a facilidade de transporte vivida atualmente permitiram a expansão do mercado económico a nível mundial, propulsionando o denominado mercado competitivo. Apesar desta tendência beneficiar os consumidores, devido à maior oferta, esta tende a prejudicar os diversos setores produtores da atividade. Por consequência, as indústrias estão cada vez mais competitivas, estimulando a eficiência dos processos intralogísticos, traduzindo-se numa estratégia redirecionada à gestão de recursos e otimização do desempenho operacional. Para o efeito, evidencia-se que a maioria das estratégias se baseia no desenvolvimento tecnológico, com especial realce nas soluções atualmente disponíveis no mundo da automação e da robótica. Dada a problemática, o atual projeto pretendeu apresentar uma solução inteligente e flexível que possibilitasse a partilha de mecanismos operacionais entre diferentes tipos de processos industriais e que, simultaneamente, auxiliasse os processos associados à logística. Considerando as imposições do projeto, foi desenvolvido um método de controlo e gestão de comportamentos que capacita uma plataforma omnidirecional (KMP 200 *omniMove*) de navegar autonomamente em ambientes industriais dinâmicos, de forma consciente e eficiente, estruturando uma solução de logística interna. Ademais, por forma a incorporar o conceito de partilha de mecanismos entre processos industriais, foram ainda implementadas estratégias que possibilitam a execução de manobras de acostagem aos locais pretendidos (máquinas de trabalho e locais de *park*), permitindo a posterior manipulação (se nos locais de trabalho).

A fim de atingir os objetivos traçados, foi, numa fase inicial, realizada uma análise literária das temáticas abordadas, tendo especial enfoque nas estratégias e metodologias adotadas por diversos autores para as problemáticas. Dentro das quais, foram revistas as estratégias de controlo de movimentos que permitem a navegação autónoma em ambientes dinâmicos, sendo, posteriormente, estabelecido uma comparação de desempenho face às exigências de implementação e computacional. Além do mais, apesar de terem sido estudadas técnicas de geração e controlo de movimentos holonómicos, foi do interesse realçar o método de navegação autónoma que fosse passível, de forma

mais simplificada, de ser ajustado à definição de movimentos holonómicos. Para o efeito, a solução proposta recorreu aos sistemas dinâmicos não-lineares como planeador local, sendo flexível e adaptável a eventuais alterações do ambiente de navegação. Além do mais, foram averiguadas possíveis soluções que possibilitassem a execução de manobras de acostagem, de forma autónoma, abrangendo tanto técnicas para veículos de direção diferencial como também para veículos omnidirecionais. Considerando os requisitos e as características holonómicas da plataforma móvel, a estratégia adotada baseou-se no controlo do tipo proporcional, este em função dos erros de posição e de orientação relativamente à *pose* final desejada.

A solução proposta estruturou-se em três módulos gerais, formulando uma hierarquia organizada, implementando as componentes de controlo e de sensorização (e derivados) numa composição modular. Esta metodologia permitiu o desenvolvimento paralelo e independente dos comportamentos requeridos, possibilitando a caracterização de uma determinada tarefa numa atuação sequencial de ações ou de comportamentos. A integração e orquestração dos comportamentos foi gerida por uma componente dedicada, denominada por *task manager*, responsável pela interpretação do pedido ou tarefa requerida, do *Service Manager*, traduzindo-o em operações elementares, interpretáveis pelas componentes constituintes do módulo *Movement Controller*.

O estudo do comportamento do modelo dinâmico da plataforma móvel em ambiente de simulação revelou ser uma iteração crucial no desenvolvimento do projeto, estabelecendo-se como um recurso de suporte à validação das componentes que integram a solução proposta. Inicialmente, as componentes e respetivos comportamentos foram verificados isoladamente, aferindo a eficácia e desempenho face aos desafios de cenário. Numa segunda fase, os testes de simulação permitiram validar a solução desenvolvida como um todo, validando os requisitos e objetivos de projeto inicialmente previstos. Nesta fase, foi também averiguado a capacidade de receção de um pedido externo do *Service Manager*, bem como a gestão de ativação dos comportamentos em virtude das tarefas solicitadas pelo servidor.

Os testes e resultados apresentados no capítulo Testes e Resultados demonstram que os objetivos inicialmente propostos foram alcançados, validando, em simulação, a estratégia de controlo dos movimentos da plataforma móvel e de coordenação e gestão de comportamentos que permitem a execução de uma determinada tarefa, em ambiente complexo. A validação em simulação permitiu corroborar a estabilidade do método proposto, sustentando a exequibilidade do conceito e ideologia do projeto (apresentada no capítulo Introdução), viabilizando a implementação real da solução proposta nesta dissertação.

10.2. TRABALHO FUTURO

O trabalho apresentado nesta dissertação estabelece um marco ao desenvolvimento de novas soluções inteligentes e flexíveis que maximizem a eficiência de processos industriais, sendo suscetível a melhorias e possíveis alterações dos métodos implementados. Revisitando a estratégia de controlo das operações de aproximação e afastamento, e estabelecendo comparação para com o método de controlo dos movimentos holonómicos, identifica-se a possibilidade de alteração da estratégia de controlo do erro de orientação, podendo este passar a ser definido por um sistema dinâmico.

Sendo este um trabalho de cariz prático, a implementação real da solução revela ser um tópico essencial ao desenvolvimento futuro, possibilitando a identificação e correção de possíveis vulnerabilidades do sistema em contexto real. Um dos pontos que em contexto real terá de ser reequacionado é a forma como o robô reconhece a posição de acostagem. A posição e orientação providenciado pelo sistema de posicionamento não garante uma acostagem com posicionamento preciso junto às máquinas ou linhas de montagem. O uso de marcadores junto dessas posições poderá ser uma solução, servindo como pontos de referência ao sistema nos processos de aproximação. Nesta solução é necessário um sistema de visão capaz de detetar os marcadores e recolher a informação presente nos mesmos.

Ademais, a criação de um módulo de interação com os operadores humanos também poderá vir a ser equacionada, providenciando uma comunicação do veículo com as pessoas, podendo, por exemplo, elucidar a manobra de desvio (caso seja necessário compensar a rota de navegação), avisar a sua presença no espaço, podendo até pedir que desempeçam o caminho, ou mesmo em casos específicos, pedir auxílio, requisitando o controlo "manual", podendo este ser estabelecido por um módulo de operação remota.

Sendo este um trabalho de um projeto global de investigação, que engloba outras temáticas (desenvolvidas por outros trabalhos de dissertação), é pretendido como objetivo futuro incorporá-las num sistema único. Para o efeito, como trabalho futuro, pretende-se validar a incorporação da solução proposta neste trabalho de dissertação com a que implementa o movimento do braço robótico (nas estações de trabalho), realizando tarefas de *machine tending* e de *pick and place*. Além do mais, é ainda pretendido a incorporação de uma frota de manipuladores móveis no mesmo ambiente de navegação sendo, para este propósito, necessário a integração com um sistema de gestão de frotas, também definido como objetivo de outro tema de dissertação.

REFERÊNCIAS

AG SICK. (2021). *Described product S300*.

Basheer Essa, A., Al-Mayyahi, A., Wang, W., Hussien, A., & Birch, P. (2017). Motion control design for unmanned ground vehicle in dynamic environment using intelligent controller. In *International Journal of Intelligent Computing and Cybernetics* (Vol. 10, Issue 4). http://sro.sussex.ac.ukThisversionisavailablefromSussexResearchOnline:http://sro.sussex.ac.uk/id/eprint/67171/

Bavelos, A. C., Kousi, N., Gkournelos, C., Lotsaris, K., Aivaliotis, S., Michalos, G., & Makris, S. (2021). Enabling flexibility in manufacturing by integrating shopfloor and process perception for mobile robot workers. *Applied Sciences (Switzerland)*, *11*(9). https://doi.org/10.3390/app11093985

Biao H, Cao Z, & Zhou MC. (2021). *n efficient RRT-based framework for planning short and smooth wheeled robot motion under kinodynamic constraints.*

Bicho, E. (1999). *Dynamic Approach to Behavior-Based Robotics*.

Bogaerts, B., Sels, S., Vanlanduit, S., & Penne, R. (2020). Connecting the CoppeliaSim robotics simulator to virtual reality. *SoftwareX*, *11*. https://doi.org/10.1016/j.softx.2020.100426

Chen YF, Everett M, LiuM, & How JP. (2017). *Socially aware motion planning with deep reinforcement learning.*

Cherni, F., Boutereaa, Y., Rekik, C., & Derbel, N. (2016). *Path Planning for mobile robots using fuzzy logic controller in the presence of static and moving obstacles*.

Choi, J., Lee, G., & Lee, C. (2021). Reinforcement learning-based dynamic obstacle avoidance and integration of path planning. *Intelligent Service Robotics*. https://doi.org/10.1007/s11370-021-00387-2

Christensen, H. I. (2012). Formulation of a U.S. national strategy for robotics. *IEEE Robotics and Automation Magazine*, *19*(2). https://doi.org/10.1109/MRA.2012.2193931

Collins, J., Chand, S., Vanderkop, A., & Howard, D. (2021). A review of physics simulators for robotic applications. In *IEEE Access* (Vol. 9, pp. 51416–51431). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2021.3068769

Eric Rohmer, Surya P. N. Singh, & Marc Freese. (2013). *coppeliaSim_v-rep_iros2013*.

Fahmizal, & Kuo, C. H. (2016). Trajectory and heading tracking of a mecanum wheeled robot using fuzzy logic control. *Proceedings of the 2016 International Conference on Instrumentation, Control, and Automation, ICA 2016*, 54–59. https://doi.org/10.1109/ICA.2016.7811475

Fan Guangrui, & Wang Geng. (2017). *Vision-Based Autonomous Docking and Re-charging System for Mobile Robot in Warehouse Environment*.

Faria, M., Melo, F. S., & Paiva, A. (2021). Understanding robots: Making robots more legible in multi-party interactions. *2021 30th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2021*, 1031–1036. https://doi.org/10.1109/RO-MAN50785.2021.9515485

Frankovsky, P., Pastor, M., Dominik, L., Kicko, M., Trebuna, P., Hroncova, D., & Kelemen, M. (2018). Wheeled mobile robot in structured environment. *12th International Conference ELEKTRO 2018, 2018 ELEKTRO Conference Proceedings*, 1–5. https://doi.org/10.1109/ELEKTRO.2018.8398375

Gigante, F., José Núñez, M., Luis Sánchez, J., Molina, J., & Cantero, J. I. (2022). *Mobile Robotics Experimentation in Industrial Environment*.

Goodman, & Zavoro. (2009). *Kinematics_Textbook_1-3.2*.

Han, Y., & Zhu, Q. (2019). Robust optimal control of omni-directional mobile robot using model predictive control method. *Chinese Control Conference, CCC*, *2019-July*, 4679–4684. https://doi.org/10.23919/ChiCC.2019.8865344

Heggem, C., & Wahl, N. M. (2020). *Mobile Navigation and Manipulation Configuration and Control of the KMR iiwa with ROS2*.

Hilke, A. (2022, May 18). *AMRs vs. AGVs: The Differences Explained*. Conger Industries Inc. Retrieved November 21, 2022, from https://www.conger.com/amr-vs-agv/

Ignatiev, K. v., Kopichev, M. M., & Putov, A. v. (2016). Autonomous omni-wheeled mobile robots. *2016 2nd International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2016 - Proceedings*. https://doi.org/10.1109/ICIEAM.2016.7910957

Ioan Doroftei, Victor Grosu, & Veaceslav Spinu. (2007). *Omnidirectional Mobile Robot – Design and Implementation*. InTech.

Jahanian, O., & Karimi, G. (2006). Locomotion systems in robotic application. *2006 IEEE International Conference on Robotics and Biomimetics, ROBIO 2006*, 689–696. https://doi.org/10.1109/ROBIO.2006.340290

Jie Zhang, & Xiaobo Liu-Henke. (2020). *Model-based design of the vehicle dynamics control for an omnidirectional automated guided vehicle (AGV)*.

Kruse, T., Basili, P., Glasauer, S., & Kirsch, A. (2012). Legible robot navigation in the proximity of moving humans. *Proceedings of IEEE Workshop on Advanced Robotics and Its Social Impacts, ARSO*, 83–88. https://doi.org/10.1109/ARSO.2012.6213404

KUKA AG. (2021). *Mobile Robots KMP 200 omniMove Transport Vehicle Assembly and Operating Instructions*. www.kuka.com

Kumar, P. N., & Narayan, Y. S. (2011). Simulation in Robotics. In *RECENT ADVANCES IN MANUFACTURING ENGINEERING & TECHNOLOGY*. https://www.researchgate.net/publication/261097756

Li X, Cao Q, Sun M, & Yang. (2019). *Fast motion planning via free C-space estimation based on deep neural network.*

Long P, Fanl T, Liao X, Liu W, Zhang H, & Pan J. (2018). *Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning.*

Louro, L., Malheiro, T., Guimaraes, P., Machado, T., Monteiro, S., Silva, P. V., Erlhagen, W., & Bicho, E. (2019). *Motion Control for Autonomous Tugger Vehicles in Dynamic Factory Floors Shared with Human Operators**.

Louro, L., Teixeira, D., Malheiro, T., Mesquita, L., Machado, T., Monteiro, S., Erlhagen dept Mathematics, W., & Bicho, E. (2020). *A safe autonomous stacker in human shared workspaces**.

Marin-Plaza, P., Hussein, A., Martin, D., & de La Escalera, A. (2018). Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous Vehicles. *Journal of Advanced Transportation*, *2018*. https://doi.org/10.1155/2018/6392697

Markis, A., Papa, M., Kaselautzke, D., Rathmair, M., Sattinger, V., & Brandstötter, M. (2019). *Safety of Mobile Robot Systems in Industrial Applications*.

Melo, A. F., & Corneal, L. M. (2020). Case study: evaluation of the automation of material handling with mobile robots. *International Journal of Quality Innovation*, *6*(1). https://doi.org/10.1186/s40887-020-00037-y

Mestiri, Y., & Gonçalves José Lima Mohamed Aymen Slim, J. (2021). *Mobile Manipulator Robot: Omni 3 Wheels Manipulator Robot Work performed under the supervision of*.

Murray, S., Floyd-Jones, W., Qi, Y., Sorin, D., Konidaris, G., & Robotics, D. (2016). Robot motion planning on a chip. *Robotics: Science and Systems*, *12*. https://doi.org/10.15607/rss.2016.xii.004

Papa, M., Kaselautzke, D., Stuja, K., & Wölfel, W. (2018). Different safety certifiable concepts for mobile robots in industrial environments. *Annals of DAAAM and Proceedings of the International DAAAM Symposium*, *29*(1), 791–800. https://doi.org/10.2507/29th.daaam.proceedings.115

Porunov, M., & Trifonov, A. (2020, October 6). Development of a Special Module for an Industrial Collaborative Robot for Assembling Standard Threaded Products. *2020 International Multi-Conference on Industrial Engineering and Modern Technologies, FarEastCon 2020*. https://doi.org/10.1109/FarEastCon50210.2020.9271533

Qian Jia, Chao Chang, Shuqing Liu, Luhao Zhang, & Sidi Zhang. (2019). *Motion Control of Omnidirectional Mobile Robot Based on Fuzzy PID*.

Romanov, A. M., & Tararin, A. A. (2021). An Automatic Docking System for Wheeled Mobile Robots. *Proceedings of the 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering, ElConRus 2021*, 1040–1045. https://doi.org/10.1109/ElConRus51938.2021.9396509

Shamshiri, R. R., Hameed, I. A., Karkee, M., & Weltzien, C. (2018). Robotic Harvesting of Fruiting Vegetables: A Simulation Approach in V-REP, ROS and MATLAB. In *Automation in Agriculture - Securing Food Supplies for Future Generations*. InTech. https://doi.org/10.5772/intechopen.73861

Sharma, A., Zanotti, P., & Musunur, L. P. (2019). Enabling the Electric Future of Mobility: Robotic Automation for Electric Vehicle Battery Assembly. *IEEE Access*, *7*, 170961–170991. https://doi.org/10.1109/ACCESS.2019.2953712

Shneier, M., & Bostelman, R. (2015). *Literature Review of Mobile Robots for Manufacturing*. https://doi.org/10.6028/NIST.IR.8022

Siki, Z., & Takács, B. (2021). Automatic recognition of ArUco codes in land surveying tasks. *Baltic Journal of Modern Computing*, *9*(1), 115–125. https://doi.org/10.22364/BJMC.2021.9.1.06

Spiers, A., Khan, S. G., & Herrmann, G. (2016). Biologically inspired control of humanoid robot arms: Robust and adaptive approaches. In *Biologically Inspired Control of Humanoid Robot Arms: Robust and Adaptive Approaches*. Springer International Publishing. https://doi.org/10.1007/978-3-319-30160-0

Statista. (2022, January). Statista - The Statistics Portal. Retrieved November 21, 2022, from https://www.statista.com

Stowers, D. (2018). *Creating Legible Robotic Motion via Local Planning*.

Taheri, H., Qiao, B., & Ghaeminezhad, N. (2015). Kinematic Model of a Four Mecanum Wheeled Mobile Robot. In *International Journal of Computer Applications* (Vol. 113, Issue 3).

TECNALIA. (2018). *Mobile dual arm robotic workers with embedded cognition for hybrid and dynamically reconfigurable manufacturing systems*.

Vitolo, F., Rega, A., di Marino, C., Pasquariello, A., Zanella, A., & Patalano, S. (2022). Mobile Robots and Cobots Integration: A Preliminary Design of a Mechatronic Interface by Using MBSE Approach. *Applied Sciences (Switzerland)*, *12*(1). https://doi.org/10.3390/app12010419

Vivas, A., & Sabater, J. M. (2021). UR5 Robot Manipulation using Matlab/Simulink and ROS. *2021 IEEE International Conference on Mechatronics and Automation, ICMA 2021*, 338–343. https://doi.org/10.1109/ICMA52036.2021.9512650

Wang, X., Moriyama, K., Brooks, L., Kameyama, S., & Matsuno, F. (2021). Real-time global path planning for mobile robots with a complex 3-D shape in large-scale 3-D environment. *Artificial Life and Robotics*, *26*(4), 494–502. https://doi.org/10.1007/s10015-021-00706-x

Wen, R., & Tong, M. (2017). *Mecanum Wheels with Astar Algorithm and Fuzzy PID Algorithm Based on Genetic Algorithm*.

Xiuzhi Li, Xingnan Liang, Jinhui Fan, & Songmin Jia. (2018). *Position-Based Visual Servo Control of Intelligent Wheelchair/bed Docking*.

Yifan Jia, Xiaodong Song, & Sendren Sheng-Dong Xu. (2013). *2013 CACS International Automatic Control Conference : CACS 2013 conference digest : December 2-4, 2013, Fleur de Chine, Sun Moon Lake, Nantou, Taiwan*.

Zhengguang Ma, Lin Zhu, Peng Wang, & Yongguo Zhao. (2019). *Proceedings, 2019 Chinese Automation Congress (CAC2019) : Nov. 22-24, 2019, Hangzhou, China.*

Zhong, X., Tian, J., Hu, H., & Peng, X. (2020). *Hybrid Path Planning Based on Safe A* Algorithm and Adaptive Window Approach for Mobile Robot in Large-Scale Dynamic Environment*. https://doi.org/10.1007/s10846-019-01112-z/Published