UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA – PPGEEL

GABRIEL ABATTI

DESENVOLVIMENTO DE SISTEMA DE COORDENAÇÃO DE FROTAS DE ROBÔS MÓVEIS AUTÔNOMOS HETEROGÊNEOS

JOINVILLE
2023

GABRIEL ABATTI

DESENVOLVIMENTO DE SISTEMA DE COORDENAÇÃO DE FROTAS DE ROBÔS MÓVEIS AUTÔNOMOS HETEROGÊNEOS

Dissertação apresentada ao Programa de Pós–Graduação em Engenharia Elétrica do Centro de Ciências Tecnológicas da Universidade do Estado de Santa Catarina, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica.

Orientador: André Bittencourt Leal

JOINVILLE
2023

Ficha catalográfica elaborada pelo programa de geração automática da

Biblioteca Universitária Udesc,

com os dados fornecidos pelo(a) autor(a)

Abatti, Gabriel
Desenvolvimento de Sistema de Coordenação de Frota de Robôs Móveis Autônomos Heterogêneos / Gabriel Abatti. -- 2023.
99 p.

Orientador: André Bittencourt Leal
Dissertação (mestrado) -- Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Engenharia Elétrica, Joinville, 2023.

1. Sistemas Multirrobôs. 2. Robôs Móveis Autônomos. 3. Sistemas a Eventos Discretos. 4. Coordenação de Frotas de Robôs. 5. Frota de Robôs Heterogêneos. I. Leal, André Bittencourt. II. Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

GABRIEL ABATTI

DESENVOLVIMENTO DE SISTEMA DE COORDENAÇÃO DE FROTAS DE ROBÔS MÓVEIS AUTÔNOMOS HETEROGÊNEOS

Dissertação apresentada ao Programa de Pós–Graduação em Engenharia Elétrica do Centro de Ciências Tecnológicas da Universidade do Estado de Santa Catarina, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica.

Orientador: André Bittencourt Leal

BANCA EXAMINADORA:

Prof. Dr. André Bittencourt Leal
Univ. do Estado de Santa Catarina

Membros:

Prof. Dr. Edson Roberto De Pieri
Univ. Federal do Estado de Santa Catarina

Prof. Dr. Max Hering de Queiroz
Univ. Federal do Estado de Santa Catarina

Prof. Dr. Ricardo Ferreira Martins
Univ. do Estado de Santa Catarina

Prof. Dr. Yuri Kaszubowski Lopes
Univ. do Estado de Santa Catarina

Joinville, 14 de julho de 2023

AGRADECIMENTOS

Ao meu orientador, André Bittencourt Leal pela orientação e apoio no desenvolvimento do trabalho. Também aos demais professores, Douglas W. Bertol, Yuri K. Lopes e Renan Sebem pelas ideias e discussões. Agradeço também a FAPESC pela bolsa e pelo apoio durante o desenvolvimento do trabalho e pelo fomento ao Grupo de Pesquisa em Automação de Sistemas e Robótica - GASR (FAPESC 2021TR930).

Aos meus amigos e familiares pela companhia e apoio, motivando em todos os momentos. Principalmente, aos meus avós, Irio e Leida, e minha irmã, Jennifer. Agradeço também aos meus alunos e ex-alunos que me motivam a dar o meu melhor a cada desafio.

RESUMO

Com o advento da Indústria 4.0, nos últimos anos tem havido um crescimento significativo no número de aplicações de robôs móveis autônomos (AMRs) em ambientes industriais. Nesse contexto, com a disponibilidade de vários fabricantes com modelos e tipos de robôs diferentes, são observadas frotas de robôs heterogêneos, robôs com características e funções distintas. Contudo, a coordenação dessas frotas ainda é um desafio complexo, explorado por diversos trabalhos com estratégias diferentes. Esta dissertação propõe uma arquitetura de coordenação de frota de AMRs heterogêneos dividida em três módulos, cada um com uma solução específica: Tratamento de Tarefas (TT), Escolha do Tipo de Robô (ET) e Escolha do Robô (ER). A solução para o módulo TT se baseia em uma política que considera a ordem de requerimento e visa otimizar o uso da frota. Já o módulo ET é responsável por escolher o tipo mais adequado de robô, dependendo das políticas previamente determinadas, e para tanto se utiliza uma solução baseada em métodos formais. A solução de ET disponibiliza os modelos e especificações de controle em autômatos de forma genérica para gerar os supervisores que garantem o comportamento do módulo ao receber as tarefas, considerando os tipos de robôs disponíveis no momento. O último módulo, ER, determina o melhor robô para executar a tarefa, escolha determinada por meio de algoritmos de escolha. O trabalho apresenta soluções funcionais para cada módulo que garantem o comportamento esperado do sistema. Foi desenvolvida uma plataforma com capacidade de implementação da arquitetura para realização de estudos de caso e validação. A plataforma é composta por uma interface com o usuário e possui integração com uma frota de robôs disponível por meio do simulador Robotarium e com a capacidade de comunicação com robôs construídos com o kit da LEGO EV3. Foram elaborados três cenários para testes, compostos por diferentes frotas para apresentar situações e comportamentos específicos do sistema. Os testes foram realizados através do simulador de frotas, dentre esses, alguns foram testados experimentalmente. A partir dos resultados obtidos por simulação e experimentos, pode-se concluir que com a utilização da arquitetura foi garantido o comportamento da frota de robôs para realização das tarefas, considerando as políticas estabelecidas. Também foi apresentada uma flexibilidade para estabelecer novas políticas e estratégias para alocações de tarefa, conforme as necessidades do sistema.

Palavras-chave: Sistema Multirrobôs. Robôs Móveis Autônomos. Sistemas a Eventos Discretos. Coordenação de Frotas de Robôs. Frota de Robôs Heterogêneos.

ABSTRACT

In recent years, there has been a substantial surge in the utilization of autonomous mobile robots (AMRs) within industrial environments, attributed to the advent of Industry 4.0. Within this context, the availability of diverse manufacturers offering a variety of robot models and types has resulted in the presence of heterogeneous robot fleets. These fleets comprise robots with distinct characteristics and functionalities. Nonetheless, effectively coordinating such fleets remains an intricate and multifaceted challenge, which has been thoroughly explored in various scholarly works employing diverse strategies. The present dissertation proposes an architecture aimed at coordinating heterogeneous AMR fleets, divided into three distinct modules, each offering a solution: Task Handling (TT), Choice of Robot Type (ET), and Choice of Robot (ER). The TT module's solution is predicated on an optimizing policy that considers the sequential order of task requests, with the ultimate goal of maximizing fleet utilization. On the other hand, the ET module assumes the responsibility of judiciously determining the most suitable robot type, utilizing a solution grounded in formal methodologies. This particular solution provides generic control specifications and models encapsulated in automata format, thereby enabling the generation of supervisors that guarantee the desired behavior of the module when confronted with incoming tasks, taking into account the available robot types at any given moment. The final module, ER, serves the purpose of ascertaining the most optimal robot for executing a specific task, employing sophisticated choice algorithms to inform the decision-making process. Notably, this work showcases functional solutions for each module, ensuring the attainment of the anticipated system behavior. In order to facilitate rigorous case studies and validation, an innovative platform capable of seamlessly implementing the proposed architecture has been developed. This platform comprises a user interface and seamlessly integrates with a fleet of robots via the Robotarium simulator. Moreover, it exhibits seamless communication capabilities with robots constructed using the LEGO EV3 kit. To substantiate the effectiveness of the proposed architecture, three distinct test scenarios have been devised, encompassing diverse fleets, thereby providing a comprehensive evaluation of specific system circumstances and behaviors. Extensive testing has been conducted utilizing the fleet simulator, complemented by empirical experiments in select scenarios. Based on the conclusive results garnered from simulations and experiments, it can be concluded that the architecture, when employed, guarantees the desired fleet behavior during task execution, effectively aligning with the established policies. Furthermore, the architecture affords inherent flexibility, enabling the establishment of novel policies and strategies for task allocations in accordance with the ever-evolving system requirements.

Keywords: Multi-robot Systems. Autonomous Mobile Robots. Discrete Event Systems. Robot Fleet Management. Heterogeneous Robot Fleet.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AGV	*Automated Guided Vehicle*
AMR	*Autonomous Mobile Robots*
AUV	*Autonomous Underwater Vehicle*
VANT	Veículo Aéreo Não Tripulado
LiDAR	*Light Detection and Ranging*
SED	Sistemas a Eventos Discretos
IFR	*International Federation of Robotics*
MQTT	*Message Queuing Telemetry Transport*
MRTA	*Multi-robot Task Allocation*
IoT	*Internet of Things*
TCS	Teoria de Controle Supervisório
ET	Escolha do Tipo do Robô
TT	Tratamento de Tarefa
ER	Escolha do Robô

LISTA DE SÍMBOLOS

E	Autômato para a Especificação de Controle
G	Autômato para a Planta
L	Linguagem
ε	Palavra Vazia
$L(G)$	Linguagem Gerada pelo Autômato G
$L_m(G)$	Linguagem Marcada Gerada pelo Autômato G
$SupC$	Máxima Linguagem Controlável
S	Supervisor
X	Conjunto de Estados
x_0	Estado Inicial
X_m	Conjunto de Estados Marcados
Σ	Alfabeto Finito de Eventos
Σ_c	Alfabeto de Eventos Controláveis
Σ_{uc}	Alfabeto de Eventos não Controláveis
δ	Função de Transição do Autômato
G_{Local_i}	Planta Local i
S_i	Supervisor Modular Local i
ζ	Tipos de Robôs na Frota
η	Número de Tipos de Robôs para Realizar a Tarefa (Primeiro Grupo para Tarefa Colaborativa)
θ	Número de Tipos de Robôs do Segundo Grupo para Realizar a Tarefa Colaborativa
τ	Tarefa Simples Genérica
σ	Tarefa Colaborativa Genérica
G_{η}^{τ}	Modelo de Planta que Descreve a Tarefa τ Simples Genérica
$G_{\eta+\theta}^{\sigma}$	Modelo de Planta que Descreve a Tarefa σ Colaborativa Genérica
α	Primeiro Grupo de Tipos de Robôs (Colaborativa)
β	Segundo Grupo de Tipos de Robôs (Colaborativa)

SUMÁRIO

1 INTRODUÇÃO

Com o avanço da Indústria 4.0, a área da robótica vem ganhando cada vez mais espaço na medida que surgem novos desafios e aplicações desses sistemas. Dentro dessa área há um destaque para aplicações com sistemas multirrobôs, com ênfase em robôs móveis para ambientes industriais, destinados tarefas e missões de transporte e logística (VERMA; RANGA, 2021). Nesses ambientes, há o surgimento de uma nova geração de robôs móveis na última década, Robôs Móveis Autônomos (*AMR - Autonomous Mobile Robots*), capazes de se deslocar por ambientes sem a necessidade de guias, possibilitando a atribuição de uma diversidade maior de tarefas (HUANG; ZHANG; XIAO, 2019).

Grande parte das soluções para sistemas multirrobôs tem como foco grupos de robôs com as mesmas características e funções. Contudo, em diversas situações, é necessária uma frota com robôs de tipos diferentes com suas próprias ferramentas e funções (VERMA; RANGA, 2021). Esse tipo de frota estabelece novos desafios para gerenciamento, comunicação e até as interações desses robôs, dependendo das tarefas e do ambiente. Frotas compostas por robôs heterogêneos geram novos desafios para estratégias de coordenação de sistemas multirrobôs (VERMA; RANGA, 2021).

Uma das problemáticas comumente explorada para sistemas multirrobôs é a alocação de tarefas, com soluções de algoritmos e métodos que consideram as características e informações dos robôs disponíveis em frotas (SARAVANAN et al., 2020). Considerando que as tarefas alocadas aos robôs móveis podem ter suas necessidades, como robôs com características específicas ou colaborações entre vários robôs (VERMA; RANGA, 2021).

Para garantir a coordenação e gerenciamento dessas frotas, uma das soluções possíveis é o uso de uma arquitetura de coordenação modular com uma organização de suas funções e aplicações, como observado em alguns trabalhos como de Ware et al. (2021) e Ju e Son (2020). Com essa abordagem, é necessário um estudo sobre os métodos para garantir o comportamento desejado.

Um dos métodos possíveis de ser explorado é tratar o sistema com a frota de robôs e as tarefas como um Sistema a Eventos Discretos (SED), os quais são sistemas com espaço de estados discretos, cuja evolução se dá pela ocorrência de eventos discretos assíncronos no tempo (CASSANDRAS; LAFORTUNE, 2021). A Teoria de Controle Supervisório (TCS) e suas extensões, proveem um ferramental matemático para a obtenção de lógicas de controle que são minimamente restritivas e não bloqueantes (RAMADGE; WONHAM, 1989). Com o uso de supervisores é possível garantir um comportamento seguro desses sistemas, além de possibilitar o controle de sistemas mais complexos, compostos por vários subsistemas.

Considerando um sistema baseado em requerimentos de tarefas para frota de robôs móveis heterogêneos, propõe-se uma arquitetura de coordenação para gerir características diferentes. Uma arquitetura que possibilita políticas entre as tarefas requeridas, que consideram a ordem de chegada, mas que busque aumentar a eficiência da frota. O uso da TCS estabelece as políticas

de escolha do tipo de robôs móvel, definindo prioridades de escolha entre os tipos de robôs disponíveis na frota. Ao determinar o tipo de robô que realizará a tarefa, o sistema pode escolher o robô mais adequado para realizá-la, dependente das características individuais como: nível de bateria, posição e entre outras.

Ainda, no contexto da indústria 4.0 e dos sistemas ciberfísicos, a simulação de frotas de robôs em ambientes industriais consiste em importante ferramenta para validação de métodos de coordenação dessas frotas. Existem várias ferramentas de simulação de robôs, tais como o disponibilizado pelo Robotarium (TECH, 2023). Entretanto, há uma limitação entre as ferramentas de simulação disponíveis quando lidam com sistemas com frotas compostas por robôs heterogêneos. É importante também que existam plataformas capazes de realizar testes simulados e em ambientes físicos para explorar aspectos observados em ambientes reais, principalmente em ambientes industriais.

1.1 OBJETIVOS

O objetivo geral dessa dissertação é propor um sistema de coordenação de frotas de robôs móveis heterogêneos por meio de uma arquitetura dividida em módulos, capaz de solucionar vários aspectos desses sistemas, considerando que cada módulo possui uma solução própria. Assim, estabelecendo um módulo intermediário entre o responsável pela tarefa e a escolha de robôs, tendo o comportamento esperado por meio de um supervisor que garante as políticas de prioridades de tipos de robôs para execução das tarefas, buscando possibilitar uma flexibilidade para determinar dessas políticas.

Para o desenvolvimento desse trabalho, destacam-se os seguintes objetivos específicos:

- Desenvolver mecanismos e ferramentas para validação dos métodos de coordenação;
- Definir cenários e sistemas que a arquitetura pode ser implementada;
- Estabelecer uma política funcional para o tratamento de tarefas, que seja eficiente para sistemas multirrobôs;
- Propor modelos de planta e especificações de controle genéricos e escaláveis que garantam políticas de escolhas entre tipos de robôs móveis através da Teoria de Controle Supervisório;
- Apresentar alternativas de algoritmos de escolha para os robôs considerando características de posição e bateria;
- Realizar testes que comprovem o funcionamento e eficiência do sistema de coordenação.

1.2 TRABALHOS RELACIONADOS

Com o controle supervisório ganhando espaço como solução de sistemas automatizados, também ganha destaque em trabalhos de controle para sistemas robóticos. Nesse aspecto, primeiro é importante destacar trabalhos que tratam de garantir o funcionamento de robôs móveis autônomos como o de Torrico, Leal e Watanabe (2016) que visa garantir o comportamento de um robô de competição na categoria de sumô autônomo. O trabalho apresenta modelos que descrevem o comportamento livre do robô e de especificações de controle que garantem o comportamento esperado em malha fechada, considerando os objetivos de um robô sumô (TORRICO; LEAL; WATANABE, 2016).

Também há trabalhos para garantir o comportamento de sistemas com múltiplos robôs, como o trabalho de Lopes et al. (2016), que utiliza técnicas de controle supervisório de SEDs para a coordenação de enxame de robôs. No trabalho, são apresentados os modelos de planta e especificações de controle que garantem o comportamento do sistema, ainda abre a discussão entre três abordagens de supervisores diferentes: monolítica, modular e modular local (LOPES et al., 2016).

Além de coordenação de robôs terrestres, a TCS também vem sendo utilizada na coordenação de sistemas com veículos aéreos não-tripulados (VANTs), como no trabalho de Florencio et al. (2018), que trata de utilização de múltiplos VANTs para atendimento de requisições como nas aplicações de missões de resgate, monitoramento de fronteiras e sistemas de segurança residencial. Nesse trabalho é proposta uma estrutura para otimizar custos, representando a comunicação entre os agentes por meio de modelos em autômatos.

Tatsumoto et al. (2018) trataram do problema de gerenciamento de frotas de robôs em ambientes de armazéns ou estoques. No trabalho é considerado que os robôs possuem as mesmas características, recebem tarefas com informações de posição inicial, posição do objeto e posição final que objeto deve ser transportado. Todos os robôs são representados via um modelo genérico em autômato (G), para facilitar a escalabilidade no número de robôs no sistema. A eficácia da abordagem se baseia em um estudo de caso, a partir do qual os autores concluem que é eficaz no controle de sistemas complexos com vários agentes.

Dulce-Galindo et al. (2019) propõem uma abordagem para a navegação autônoma de múltiplos robôs com tarefas atribuídas por um agendador. A TCS é utilizada para obter os supervisores que irão garantir o comportamento de cada robô, se baseando na dinâmica entre duas políticas de navegação, uma focada na localização da tarefa e outra para desvio de obstáculos. Já em um trabalho mais recente dos mesmos autores (DULCE-GALINDO et al., 2022), são investigadas abordagens de supervisão distintas, uma centralizada e outra descentralizada.

Existem também trabalhos que utilizam técnicas de controle supervisório para gerenciar frotas de robôs móveis heterogêneos, dentre os quais se destaca o trabalho de Ju e Son (2023), que trata da coordenação desse tipo de frota para ambientes agrícolas. Nele é proposto o gerenciamento de dois tipos de robôs, aéreo e terrestre, e o comportamento dos robôs é garantido

com o uso de dois supervisores específicos, um para cada tipo de robô.

Ainda com frotas heterogêneas, o trabalho de Simon et al. (2023) trata de coordenação de frotas compostas por veículos aéreos não tripulados (UAV — *Unmanned Aerial Vehicle*) e veículos terrestres não tripulados (UGV — *Unmanned Ground Vehicle*) por meio de arquiteturas de coordenação. Parte dessas arquiteturas tem a solução garantida através do controle supervisório, que estabelece o comportamento esperado de cada robô. As missões são focadas para problemas de busca e salvamento por meio de sistemas com múltiplos robôs.

Além de trabalhos que propõem a coordenação de robôs, são importantes os trabalhos de Janarthanan, Gohari e Saffar (2006) e Chen e Wonham (2002), pois tratam ou mencionam políticas de prioridade para diferentes finalidades com o uso do controle supervisório. Os trabalhos estabelecem modelos ou estratégias para garantir políticas entre o comportamento total de tarefas ou ainda prioridades distintas para eventos controláveis.

1.2.1 Trabalhos com arquiteturas de coordenação

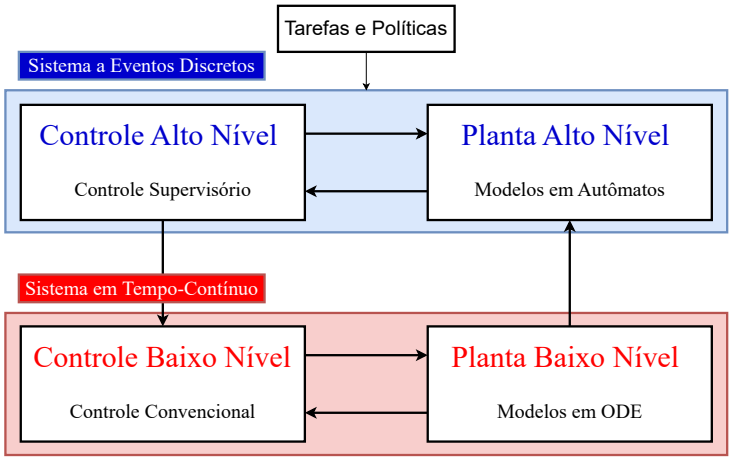
Trabalhos que abordam arquiteturas ou estruturas de coordenação para garantir o controle de frotas de robôs móveis também são importantes de serem destacados para esta dissertação. Existem materiais e trabalhos que estabelecem arquiteturas de coordenação para ambientes variados, como o descrito pela patente de Ware et al. (2021), que estabelece uma arquitetura dividida em três módulos: um para o gerenciamento da bateria, outro para o gerenciamento e tratamento das tarefas e um módulo para o planejamento dos trajetos dos robôs.

Em diversos trabalhos analisados, identificou-se um padrão adotado para a arquitetura de coordenação. Nesses trabalhos, a arquitetura é dividida em dois módulos, conforme ilustrado por Ju e Son (2020). Além disso, Ju e Son (2020) adotam uma estrutura dividida em dois níveis hierárquicos, de alto e baixo nível. A TCS é usada para obtenção de supervisores responsáveis pelo controle de alto nível, enquanto técnicas de controle de sistemas contínuos são adotadas para o controle de baixo nível. O trabalho destaca que teorias e arquiteturas de coordenação de frotas de robôs heterogêneos tradicionais podem não ser adequadas em determinados ambientes e para determinadas tarefas, principalmente em ambientes que necessitam de alta escalabilidade e/ou com características modulares.

Já no trabalho de Li et al. (2008), que trata da coordenação de veículos subaquáticos autônomos (AUV - *Autonomous Underwater Vehicle*), utiliza uma arquitetura de controle dividida em três camadas: camada de missão, camada de tarefa e camada de execução. A arquitetura proposta foi atribuída a uma série de tarefas executadas por um time de veículos homogêneos, todos com as mesmas capacidades e funcionalidades.

Outro trabalho com a solução por meio de uma arquitetura de coordenação relevante para essa dissertação é o de Liu, Ficocelli e Nejat (2015). Cujo objetivo é estabelecer políticas de alocação de tarefas com frotas de robôs heterogêneos para missões de busca e salvamento. Nele também há uma arquitetura dividida em módulos, com um específico para o controle de tarefas e outros tratando do comportamento dos robôs. Na solução de Liu, Ficocelli e Nejat (2015), a

Figura 1 – Arquitetura para coordenação de robôs em ambientes agrícolas proposta por Ju e Son (2020).

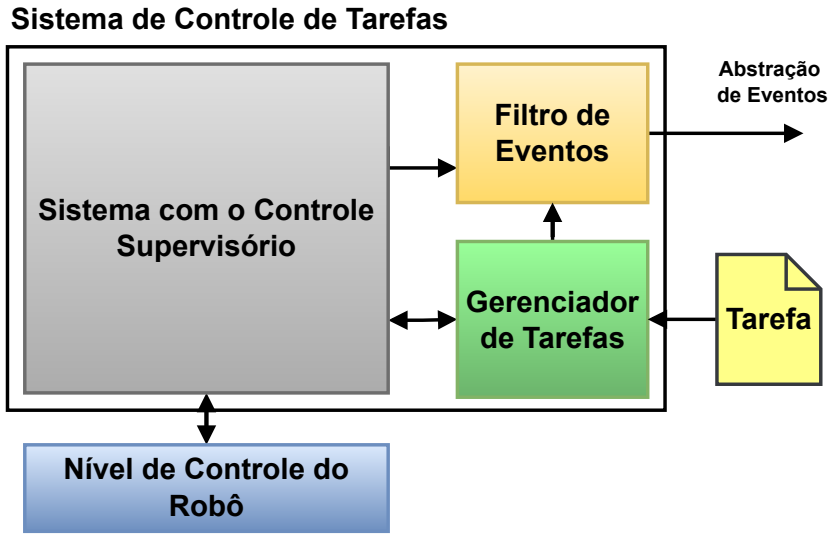


Fonte: Adaptado de Ju e Son (2020).

teoria de controle supervisório é adotada tanto para o gerenciamento e alocações das tarefas como para garantir o comportamento espero dos robôs.

A arquitetura mais próxima da que será proposta nesta dissertação é apresentada na Figura 2 (SIMON et al., 2023). Simon et al. (2023) propõem um sistema de coordenação com múltiplos robôs para missões de busca e salvamento urbano (USAR) com o uso da teoria de controle supervisório. Nele, é proposto um sistema dividido em duas camadas. A camada reativa que garante por meio de supervisores que os robôs satisfaçam situações inesperadas (como desviar de obstáculos) e a camada deliberativa que coordena o sistema multirrobôs através de heurísticas para execução das tarefas com eficiência. Os cenários possuem dois tipos de robôs, terrestres e o aéreos, capazes de execução de tarefas específicas. Essa arquitetura separa em módulos com funções para tratar as tarefas submetidas ao sistema, com módulos específicos para tratar o sistema como um Sistema a Eventos Discretos, um para controlar o robô e outro para o tratamento da tarefa submetida.

Figura 2 – Arquitetura para o controle de tarefas inserido em cada robô proposta por Simon et al. (2023).



Fonte: Adasptado de Simon et al. (2023).

1.3 ORGANIZAÇÃO DO DOCUMENTO

Esta dissertação é apresentada em seis capítulos. No Capítulo 2, são apresentadas as ferramentas e conceitos que contribuem para o desenvolvimento das soluções de cada módulo de coordenação ou do desenvolvimento da plataforma, destacando o estudo sobre a Teoria de Controle Supervisório e conceitos fundamentais para sistemas com robôs móveis. Já no Capítulo 3, são tratadas as especificações e definições de sistemas nos quais a arquitetura proposta pode ser implementada, propondo a arquitetura de coordenação e as soluções para cada módulo.

O Capítulo 4 apresenta a implementação do coordenador com uma plataforma, destacando os recursos e ferramentas desenvolvidas. Já o Capítulo 5, apresenta os testes simulados e no laboratório e o comportamento geral do coordenador, divididos em três estudos de casos com cenários diferentes. Por fim, no Capítulo 6, são apresentadas as conclusões, limitações e trabalhos futuros da dissertação.

2 CONCEITOS FUNDAMENTAIS

Este capítulo apresenta as ferramentas e conceitos fundamentais utilizados para o desenvolvimento das soluções de coordenação de frotas de robôs móveis e para o desenvolvimento da plataforma com a arquitetura de coordenação.

2.1 SISTEMAS COM MULTIRROBÔS MÓVEIS

Conforme a norma ISO 8373:2021, da *International Organization for Standardization*, que trata de conceitos e palavras na área da robótica, define-se o conceito de robôs autônomos como aqueles capazes de realizar tarefas sem a intervenção humana. O documento também define três áreas e aplicações de robôs: robôs industriais para ambientes industriais, robôs de serviço para a realização de tarefas em prol de pessoas e equipamentos, e robôs médicos para uso em ambientes hospitalares (ISO, 2021).

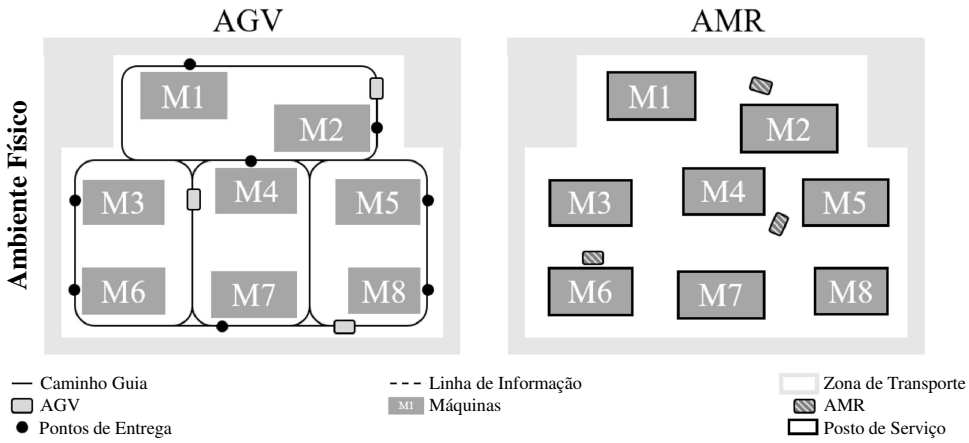
2.1.1 AMR — Robôs Móveis Autônomos

A primeira geração de robôs móveis em ambientes industriais, conhecida como Geração de Veículos Autoguiados (AGV — *Automated Guided Vehicle*), já foi explorada em diversos processos industriais e trabalhos acadêmicos. Esses robôs possuem a capacidade de transporte em ambientes, sendo conduzidos com o auxílio de guias previamente instalados (HUGH, 2001). Com o rápido desenvolvimento de robôs móveis, os Robôs Móveis Autônomos (AMR — *Autonomous Mobile Robot*) estão ganhando cada vez mais espaço, principalmente em ambientes industriais e de serviços (FRAGAPANE et al., 2021).

A Figura 3 apresenta uma comparação entre a locomoção de AGVs e AMRs, com as movimentações possíveis dos robôs (com os AGVs limitados em linhas fixas e os AMRs livres para determinar as trajetórias). Ainda sobre a Figura 3, é destacado que o AGV ainda precisa de um posto de serviço específico para executar a tarefa, enquanto o AMR há uma flexibilidade já que ele pode chegar ao posto de serviço em qualquer ponto da máquina. Fragapane et al. (2021) afirmam que o uso de AMRs oferece maior flexibilidade, por dois motivos: menor necessidade de preparo e recursos no ambiente para locomoção e facilidade de desvio de obstáculos móveis ou fixos no ambiente.

De acordo com relatórios da IFR (2021), a presença de AMRs em soluções de logística aumentou seis vezes entre 2014 e 2019 e o número de instalações de robôs móveis continua a subir a cada ano. Os AMRs possuem maior liberdade para explorar ambientes, graças a sensores mais eficientes para auxiliar na navegação, incluindo o uso de tecnologias como o *Light Detection and Ranging* (LiDAR), sistemas de visão, acelerômetros, giroscópios e encoders. Isso possibilita um melhor cálculo de trajetórias com um controle descentralizado, em que todos os robôs determinam a melhor rota a seguir para realizar a tarefa, conforme destacam Fragapane et al. (2021).

Figura 3 – Comparativo entre AGV e AMR.



Fonte: Adaptado de Fragapane et al. (2021)

Na Figura 4 são apresentados três modelos de AMR de fabricantes diferentes e, por consequência, possuem características distintas, ainda com os mesmos aspectos construtivos. Esses robôs são modelos básicos, utilizados principalmente em funções de transporte e logística. Cada modelo pode apresentar diferentes características, tais como capacidade de carga, capacidade de bateria ou velocidade máxima que podem atingir.

Figura 4 – Robôs Móveis de Três Fabricantes Distintos: a) MIR (2023), b) ABB (2023) e c) OMRON (2023b)

a) b) c)

Fonte: O autor (2023)

Em ambientes industriais, são encontrados AMRs com manipuladores para auxiliar e manusear ferramentas, equipamentos e até produtos. Na Figura 5 são apresentados alguns modelos de AMRs com manipuladores, todos em seus ambientes específicos. O robô da MIR (2023) da Figura 5 a) é usado para transporte de cargas pesadas, já o da ABB (2023) (b) é

adotado em manuseios em laboratórios. O robô da OMRON (2023b) (c) com um manipulador para transporte de pequenas peças ou objetos em ambientes industriais.

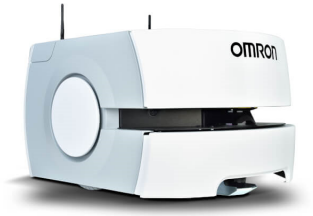
Figura 5 – Robôs Móveis com Manipulador dos fabricantes a) MIR (2023), b) ABB (2023) e c) OMRON (2023b) respectivamente.

a) b) c)

Fonte: O autor (2023)

Até em modelos simples para transportes, há uma variedade de características e capacidades em modelos de AMRs. Na Figura 6 são apresentados dois modelos distintos de AMR do mesmo fabricante, contudo suas capacidades diferem: o LD-90 possui a capacidade de transporte de carga de 90 kg com velocidade máxima de 1,35 m/s (4,86 km/h), enquanto o HD-1500 possui capacidade de transporte de 1500 kg em uma velocidade de 1,8 m/s (6,48 km/h).

Figura 6 – Robôs móveis do mesmo fabricante, mas com características distintas: o da esquerda possui menor capacidade de carga, menores velocidades e menor peso que o da direita.



LD – 90	
Capacidade de Carga	90 kg
Max. Velocidade	1,35m/s
Max. Velocidade Rotação	180°/s
Peso do robô	62 kg

HD – 1500	
Capacidade de Carga	1500 kg
Max. Velocidade	1,8m/s
Max. Velocidade Rotação	60°/s
Peso do robô	506,5 kg

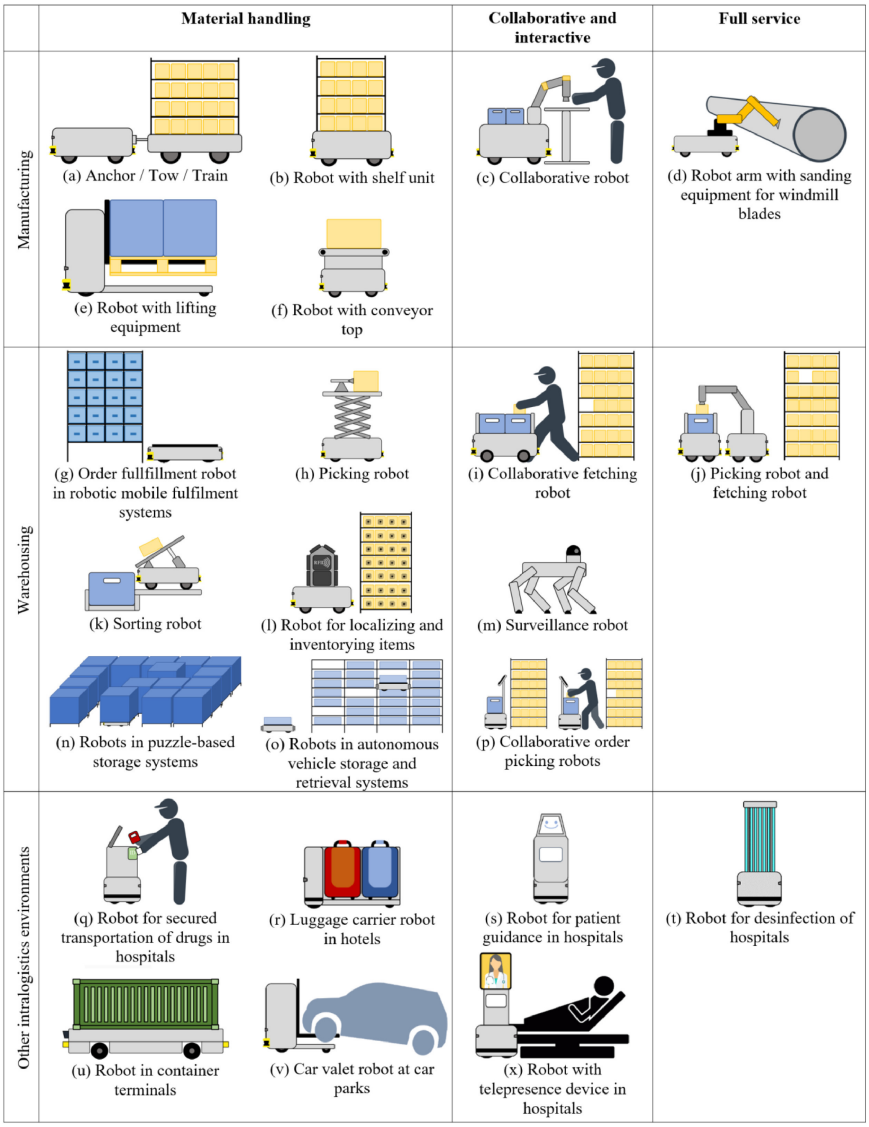
Fonte: O autor (2023)

2.1.2 Tarefas para robôs móveis

Analisando o relatório da IFR (2021) observa-se que os AMRs estão ganhando destaque devido ao crescimento no número de ambientes onde esses robôs são instalados. Dependendo desse ambiente, as tarefas realizadas por robôs móveis podem ser distintas, existindo uma ênfase maior para tarefas de transporte de peças e equipamentos devido à popularidade e às vantagens desse tipo de automação em processos de logística.

Conforme ilustrado na Figura 7 (FRAGAPANE et al., 2021), os AMRs são aplicados em diversas tarefas, desde o simples transporte de peças até o acoplamento de equipamentos com funções específicas. É importante destacar, na Figura 7, a variação das tarefas mesmo entre aquelas de transporte. Algumas são realizadas com o auxílio de um operador (exemplos: (c), (i), (p) e (q)), outras com outro robô móvel ou ainda realizadas completamente por um único robô (exemplos: a), (d), (g), (h), (v), etc.).

Figura 7 – Exemplos de aplicações com AMRs.



Fonte: Fragapane et al. (2021)

2.1.3 Coordenação de frotas de robôs móveis

Para esta dissertação, foi importante estudar diversos produtos e identificar quais aspectos e ferramentas são importantes para o gerenciamento de robôs móveis. Considerando esse crescimento no número de robôs móveis, há um movimento de fabricantes na elaboração de ferramentas para coordenar frotas de robôs. Dos softwares analisados se destacam: o *MobilePlanner* da Omron (2023a), o *MiR Fleet* da MiR (2023) e o *OTTO Fleet Manager* da OttoMotors (2023).

Na literatura, encontram-se trabalhos focados em coordenação de robôs móveis, como o artigo de Cai (2020) que aborda o uso de sistemas multirrobôs em ambientes de armazéns e galpões. Outro trabalho que aborda a coordenação é o de Dai et al. (2022), o qual trata especificamente de robôs em ambientes industriais, em trabalhos com estruturas grandes e complexas, propondo também uma solução flexível para adaptações em métodos de tomada de decisões.

São identificados diversos desafios ao tratar de sistemas dessa natureza. Dentre elas, muitos desses desafios com base na abordagem para a coordenação de grupos de robôs, como: a escalabilidade, alocação de tarefas, arquitetura de controle, comunicação e robustez (VERMA; RANGA, 2021).

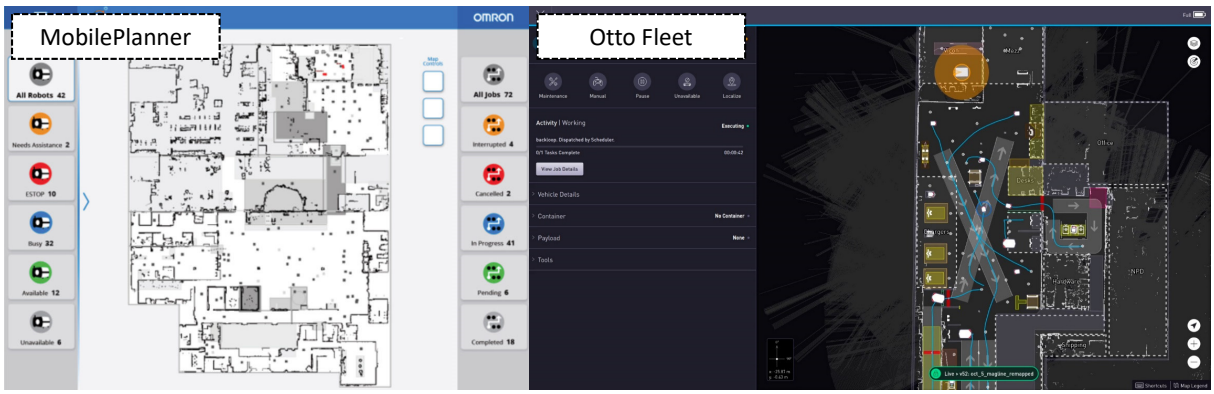
2.1.3.1 Plataformas de coordenação comerciais

Ao analisar as plataformas de coordenação dos três fabricantes citados, é possível destacar a importância da interface e da interação com o usuário. Algumas plataformas apresentam uma interface capaz de receber informações, como tarefas, áreas proibidas e registros de robôs, e também disponibilizam dados do sistema para o usuário.

Na Figura 8, são apresentadas duas das plataformas usadas como referência para esta dissertação, pois apresentavam recursos similares da proposta e disponibilidade de informações para serem analisadas. Um aspecto importante nas plataformas da Figura 8 é a representação do mapa do ambiente e a posição de cada robô móvel nele. No contexto da utilização de robôs móveis autônomos, esses mapas são criados e atualizados pelos próprios robôs, identificando a presença de paredes e obstáculos. No caso do *Otto Fleet Manager*, o trajeto do robô móvel é representado no mapa, o que possibilita ao usuário analisar o trânsito dos robôs em um ambiente.

Todas as plataformas seguem um padrão predefinido para as informações individuais dos robôs, sendo alguns desses dados disponibilizados diretamente na interface, como o nível de bateria e a posição atual do AMR. Além dessas informações, cada plataforma pode criar e estabelecer um padrão para tratar da disponibilidade dos robôs. De modo geral, um robô pode estar disponível (aguardando e esperando o requerimento de uma tarefa) ou indisponível (realizando uma tarefa, com nível muito baixo de bateria, precisando de assistência, não conectado, etc.). Com todos os dados apresentados na interface, é possível atribuir uma tarefa a qualquer robô disponível a uma tarefa.

Figura 8 – Exemplos de plataformas de coordenação de frota de robôs, disponibilizados por Omron (2023a) e OttoMotors (2023).



Fonte: O autor (2023)

Algumas limitações foram observadas analisando as informações dessas plataformas. Uma delas é possível determinar apenas algumas políticas de escolha, como critérios de menor distância até o ponto de uma tarefa. Outra limitação importante é a dificuldade de acrescentar e visualizar robôs com características diferentes ou de outros fabricantes. Logo, nas duas plataformas apresentadas há limitações ao tratar de frotas de robôs heterogêneas, principalmente ao tratar de robôs de outros fabricantes e novas funcionalidades.

2.1.3.2 Coordenação de frotas de robôs móveis heterogêneos

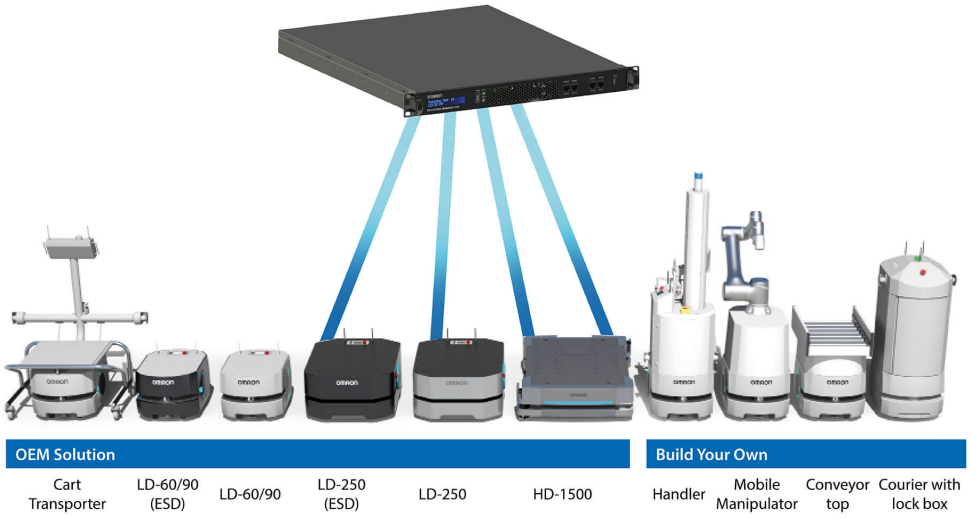
Com a evolução da coordenação de frotas de AMRs, estabelece-se o próximo desafio para esse tipo de gerenciamento: a coordenação de frotas de robôs móveis com vários tipos de robôs. A maioria dos trabalhos relacionados a sistemas multirrobôs são elaborados considerando apenas robôs do mesmo tipo (homogêneos), sendo destacado a importância do desafio de coordenação para robôs heterogêneos para o avanço da área (VERMA; RANGA, 2021). Esse desafio se baseia em dois parâmetros importantes: o primeiro é a colaboração entre robôs e o segundo são as tarefas que podem ser realizadas dependendo das características de cada robô (QUERALTA et al., 2020).

Nesse contexto, os relatórios disponibilizados pelo IFR (2021) e Sostero (2020) mostram em diferentes aspectos o crescimento no número de robôs presentes em diversos ambientes no mundo todo, também apontam a diversidade de tipos de robôs que podem pertencer a um único ambiente. Com isso, é observada a necessidade de técnicas e ferramentas de coordenação de robôs heterogêneos em frotas.

Alguns fabricantes de AMRs disponibilizam tecnologias para a coordenação de frotas de AMRs, considerando grupo de robôs heterogêneos, como apresentado na Figura 9. Esse sistema de coordenação visa melhorar o desempenho de frotas de AMRs em ambientes industriais, gerenciando o trânsito e tomada de decisões da frota com a mínima intervenção humana.

Entretanto, considerando que diferentes modelos e tipos de AMRs podem ser utilizados

Figura 9 – Exemplo de coordenador de frotas por solução de um fabricante.



Fonte: Omron (2023a).

em uma mesma aplicação, é fundamental o desenvolvimento de arquiteturas e estruturas de coordenação capazes de gerenciar frotas. Esse desafio não se restringe apenas a aplicações industriais, mas se estende a diversos ambientes. Nesse sentido, há trabalhos que abordam essa problemática para ambientes específicos.

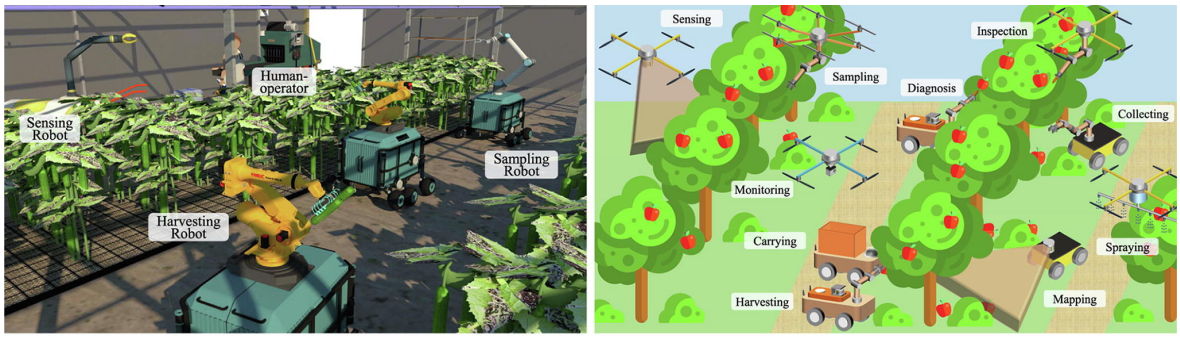
Alguns trabalhos, como o de Pereira et al. (2023) e Cechinel et al. (2021), têm como proposta principal a coordenação de robôs móveis autônomos em ambientes industriais considerando as diferenças entre as características dos robôs. No trabalho de Pereira et al. (2023) é adotado um coordenador para estabelecer as trajetórias otimizadas e seguras para os robôs móveis. Já no de Cechinel et al. (2021), é abordada a alocação de tarefas, considerando características individuais dos robôs.

No trabalho de Ju e Son (2021) aborda as frotas ilustradas na Figura 10, são apresentados dois exemplos de frotas com vários tipos de robôs com atribuições em ambientes agrícolas. Nesse caso, há o desafio de coordenar a frota com as muitas características e funções distintas dos robôs, missões e tarefas que podem até depender umas das outras (por exemplo: na Figura 10 b, o robô com função de inspeção precisa realizar sua tarefa primeiro, para depois o coletor executar a sua).

2.1.3.3 Alocação de tarefas

Considerando a coordenação de robôs móveis, a alocação de tarefas tem sido adotada em diversas aplicações, principalmente na realização de tarefas específicas em ambientes dinâmicos. Uma alocação eficiente de robôs móveis tem com objetivo proporcionar benefícios para o funcionamento de processos, como o aumento na produtividade e redução de custos. Essa eficiência pode ser alcançada com o uso de diversas estratégias, técnicas e/ou algoritmos de otimização (SARAVANAN et al., 2020). Esse problema também é chamado de alocação de

Figura 10 – Frotas de robôs heterogêneos em ambientes agrícolas.



(a) (b)

Fonte: Ju e Son (2021).

tarefas para sistemas multirrobôs(do inglês *MRTA — Multi-robot Task Allocation*).

O problema de alocação de tarefas ganha destaque para aplicações em ambientes industriais, agrícolas e principalmente aqueles focados para transporte e logística. Nesse aspecto, alguns trabalhos buscam otimizar as soluções desse problema por meio de determinados algoritmos. Um exemplo é o trabalho de Cechinel et al. (2021), que trata de alocação de tarefas para AMRs considerando características individuais como a capacidade de carga, velocidade, posição e bateria. Esse trabalho utiliza o algoritmo genético com modelos de ilhas para melhorar a eficiência na escolha do robô que realizará uma determinada tarefa.

Analisando alguns trabalhos já estabelecidos e artigos de revisão de literatura, como os de Saravanan et al. (2020) e Khamis, Hussein e Elmogy (2015), encontram-se diversas técnicas para alocação de tarefas e suas características. Dentre elas, algumas que se destacam são: otimização por enxame de partículas, algoritmo guloso e algoritmo genético.

Em alguns trabalhos, são utilizados vários algoritmos de escolha e otimização, comparando suas eficiências. No trabalho de Li et al. (2020) é feita a comparação entre o algoritmo genético e o de enxame de partícula para ambientes de armazéns, dando ênfase no algoritmo genético, que utiliza princípios de evolução biológica para otimização.

Um dos primeiros algoritmos utilizados para estabelecer um mecanismo de escolha é o **algoritmo guloso**. Esse algoritmo também é muito utilizado com o intuito de otimizar o uso de robôs em sistemas com frotas e, pela facilidade de sua implementação e flexibilidade, foi o algoritmo adotado nesta dissertação.

Com o sistema elaborado, o nome do algoritmo “guloso” é atribuído devido ao seu objetivo e comportamento: a escolha do candidato se baseia simplesmente no menor custo naquele momento, sem considerar escolhas futuras, logo é escolhido sempre o melhor candidato a cada requisição (BRASSARD; BRATLEY, 1996).

De modo geral, a solução se baseia em um problema no qual há vários candidatos a serem escolhidos, considerando que cada um pode possuir características e/ou custos diferentes (BRASSARD; BRATLEY, 1996). Para o desenvolvimento desse algoritmo são necessárias as considerações:

- É necessária a criação de uma lista com os candidatos e os custos de cada um;
- No processo, o algoritmo deve analisar os custos de todos os candidatos;
- Apenas um candidato será aceito pelo sistema, enquanto todos os outros serão rejeitados;

Algorithm 1 Algoritmo Guloso Genérico (BRASSARD; BRATLEY, 1996)

```
/* C é o conjunto de soluções                                    */
/* S é o conjunto que irá conter a solução                       */
def AlgoritmoGuloso(C: conjunto):
    S ← ∅
    while C ≠ ∅ e não existe S do
        x ← seleciona_gul (C);
        C ← C \ {x};
        if VIABILIDADE(S ∪ {x}) then
            S ← S ∪ {x};
        end
    end
    if SOLUÇÃO(S) then
        return S
    else
        return Não existe solução
    end
```

O Algoritmo 1 contém a solução genérica para um algoritmo guloso, a partir do qual se pode analisar o funcionamento da estratégia adotada. A princípio, o conjunto S, que representa os candidatos escolhidos, está vazio, então, a cada passo é utilizada a função para determinar qual o melhor candidato. Ao adicionar um candidato à solução, ele não será mais considerado nos passos futuros (BRASSARD; BRATLEY, 1996).

Esse algoritmo possui aplicações em diversos cenários com problemas distintos, estando já estabelecido também em algumas aplicações de alocação de tarefas para sistemas de múltiplos robôs, sempre determinando os robôs como candidatos para execução de tarefas e missões. Como no trabalho de Kong et al. (2019), onde é apresentado um estudo sobre o uso do algoritmo guloso e a otimização por enxame de partículas para *MRTA*, visando melhorar a eficiência na escolha do robô mais adequado a executar uma tarefa, considerandos parâmetros como a sua posição no ambiente.

2.1.4 Recursos e simuladores

Uma grande parte dos estudos de técnicas para *MRTA* fazem suas validações por intermédio de simulações e experimentos que simulam em tempo real (SARAVANAN et al., 2020). Dessa forma, é importante fazer uma análise das plataformas de simulação voltadas para propostas de coordenação de multirrobôs. Há alguns critérios para avaliar qual plataforma deve ser

utilizada, dependendo da estratégia proposta pelo trabalho são relevantes características como: flexibilidade, escalabilidade no número de robôs, interface com o usuário, se comporta tarefa simples ou complexas, etc.

Algumas das propostas de coordenação são validadas em duas etapas: mediante uma plataforma de simulação e por intermédio de experimentos com robôs físicos em ambientes previamente estabelecidos. No trabalho de An et al. (2023) são apresentadas algumas dessas plataformas, tanto as exclusivas com o uso de simuladores como outras plataformas com robôs reais. Para o desenvolvimento da plataforma desenvolvida no contexto desta dissertação, foi optado pela preferência de *softwares* que possuem desenvolvimento em Python, pela facilidade, flexibilidade e todo o material já estabelecido nessa linguagem de programação.

2.1.4.1 Interfaces gráficas

Uma interface entre o usuário e o sistema adequada é fundamental para estabelecer trabalhos sobre MRTA, pois é a ferramenta que permite a interação humana com o sistema. No contexto de plataformas de simulação, uma interface com o usuário estabelece a comunicação do usuário com o sistema, podendo o usuário informar e requisitar novas tarefas ou ainda receber informações sobre as necessidades dos testes.

Interfaces com finalidade de supervisionar operações com robôs móveis são adotadas em diversos ambientes. Os fabricantes buscam estabelecer interfaces para suas plataformas de coordenação, considerando as características dos robôs que produzem. Também há trabalhos que analisam diversos critérios de interfaces, como o de Chen, Barnes e Harper-Sciarini (2011), o qual trata principalmente da interação humana com o sistema e a interface com usuário.

Um recurso que traz uma facilidade para o desenvolvimento de interfaces gráficas é o PyQt, uma biblioteca para Python que permite esse desenvolvimento de maneira rápida, flexível e eficiente. Além disso, com os recursos disponíveis pela biblioteca é possível o desenvolvimento de plataformas complexas com estilos personalizados e suportes a *multithreading* (COMPUTING, 2023).

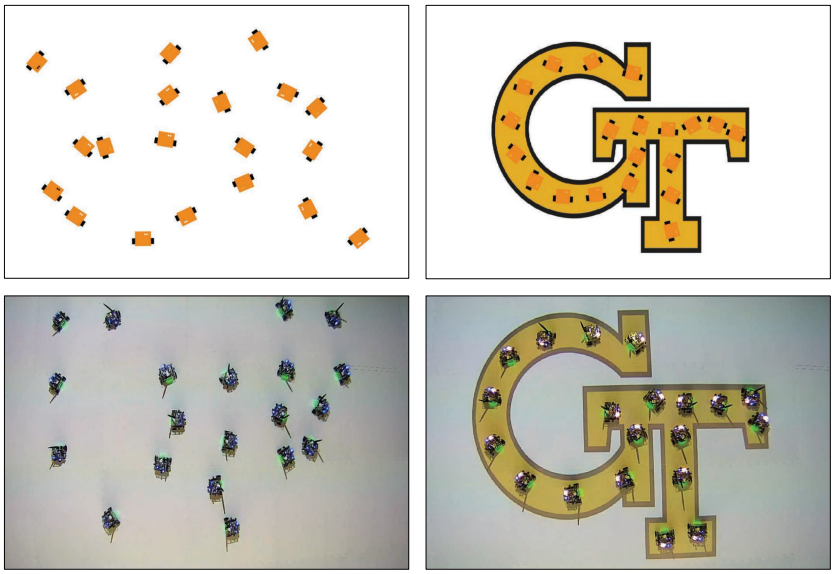
2.1.4.2 Robotarium

Além da interface com o usuário, para a validação de estratégias e políticas desenvolvidas, é necessária uma ferramenta que possibilite a realização de testes com frotas de robôs. Essa validação pode ser feita por meio de um simulador. O Robotarium (PICKEM et al., 2017) é uma plataforma composta por um simulador e um laboratório remoto desenvolvido pela GeorgiaTech, que permite aplicações de sistemas com múltiplos robôs. Esse projeto permite que pesquisadores de todo o mundo possam desenvolver, testar e analisar algoritmos de controle de robôs reais em ambiente controlado.

Nessa Figura 11 ilustra-se mais do que o ambiente real disponibilizado pelo Robotarium. Além de disponibilizar um ambiente real para testes e análises, o laboratório também disponi-

biliza ferramentas de simulação desenvolvidos em Python ou em Matlab (TECH, 2023). São disponibilizados recursos como algoritmos de trajetória e desvio de obstáculos, formando assim um ambiente de simulação ideal para representar um robô móvel autônomo.

Figura 11 – Simulador e Plataforma Robotarium.



Fonte: Wilson et al. (2020)

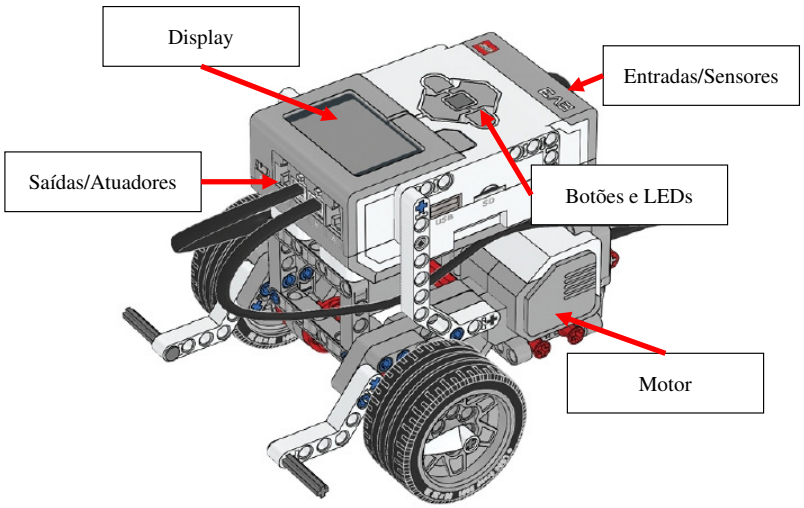
Segundo Wilson et al. (2020), a maioria das aplicações do Robotarium é proveniente de projetos educacionais, controle de formação de frotas e projetos de seguidor de linha, mas também são destaques planejamentos de trajetória e sistemas de *MRTA*. Com as vantagens na facilidade de escalabilidade no número de robôs e estabelecer um ambiente para execução de tarefas simples, o simulador Robotarium é colocado como uma boa alternativa para validação de técnicas e estratégias de coordenação de robôs.

2.1.4.3 LEGO Mindstorms EV3

No intuito de validar de forma experimental a arquitetura de coordenação proposta e a ferramenta de coordenação desenvolvida nesta dissertação, optou-se por utilizar robôs móveis montados com kits de robótica LEGO Mindstorms EV3. Esses kits possuem todos os elementos necessários para a montagem de robôs móveis autônomos e o seu bloco de controle suporta linguagens de programação mais recentes, como o *MicroPython* (GROUP, 2023), com possibilidades de desenvolvimentos adequados para o que se precisa neste trabalho.

Na Figura 12 é apresentado um robô base como exemplo, indicando a posição do display, botões, LEDs, conexões de entradas e saídas. Para esta dissertação, o robô foi construído com os dois motores e os sensores de distância e giroscópio. Outros trabalhos já validaram estratégias de coordenação de robôs móveis através da plataforma LEGO EV3. Um exemplo é o trabalho já citado de Tatsumoto et al. (2018), que utilizou esses robôs para validar a proposta de estratégia de coordenação para armazéns automatizados.

Figura 12 – Robô básico com o Kit da LEGO.



Fonte: O autor (2023)

2.1.4.4 Protocolo MQTT

Outro critério importante para a escolha de qual plataforma deverá ser usada para testes práticos com robôs móveis são as opções de comunicação com o coordenador. Com o uso do *MicroPython*, o LEGO EV3 possui facilidade de comunicação *Wireless* através do protocolo de comunicação MQTT - *Message Queuing Telemetry Transport*, acrescentando o módulo de comunicação *Wireless* (PEREIRA, 2023). O *MQTT* é um protocolo de comunicação de mensagens leves, adequado especificamente para dispositivos com processamento limitado, que ganhou ainda mais popularidade com o uso em *Internet of Things (IoT)* (MQTT, 2023).

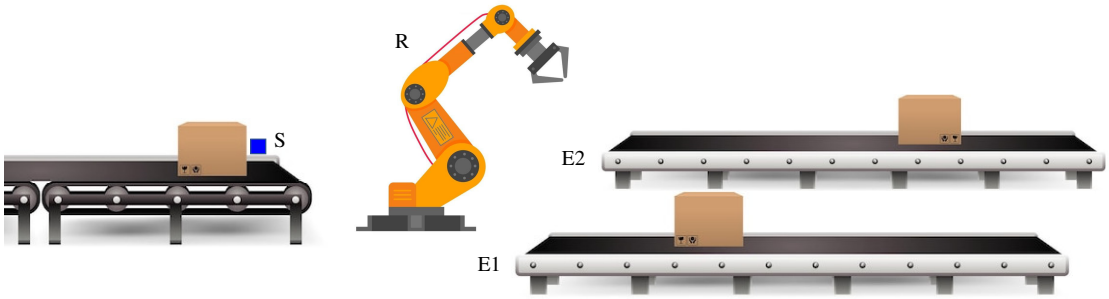
Além de disponível para dispositivos com baixa capacidade, uma das principais vantagens no uso do *MQTT* é a fácil escalabilidade de dispositivos conectados. Ele funciona com um modelo de comunicação de publicação e assinatura, no qual os dispositivos se comunicam por meio de um *broker* central. O *broker* recebe as mensagens de todos os dispositivos e distribui em tópicos que apenas os dispositivos interessados possuem assinaturas (MQTT, 2023). O formato *Json* é usado para as mensagens do protocolo, que se destaca por ser flexível e fácil de interpretar e manusear os dados a serem encaminhados e/ou recebidos. Dessa forma, para um sistema de coordenação de robôs, é importante definir os tópicos para as assinaturas dos robôs e onde estará implementado o *broker*.

2.2 SISTEMAS A EVENTOS DISCRETOS

Sistemas a Eventos Discretos (SEDs) são sistemas cujo espaço de estados assume valores simbólicos discretos e a transição entres os seus estados se dá pela ocorrência de eventos discretos assíncronos no tempo (CASSANDRAS; LAFORTUNE, 2021). Normalmente os SEDs são compostos por diversos subsistemas, os quais interagem com o intuito de alcançar determinado

objetivo. Para exemplificar, considere o sistema ilustrado na Figura 13, o qual é composto por um robô (R), uma esteira de entrada com um sensor (S) capaz de reconhecer o tipo de peça, e duas esteiras de saía (E1 e E2). Os estados do subsistema robô podem ser *em repouso* e *transportando peça* e os eventos relacionados a ele podem ser *início de transporte para E1*, *início de transporte para E2* e *fim de transporte*.

Figura 13 – Ilustração de um Robô (R) separador de peças por um sensor de peso (S).



Fonte: O autor (2023).

Estabelecendo os estados e eventos, é importante elaborar modos de descrever o sistema por meio de modelos específicos, podendo representar o comportamento do sistema de modo global ou ainda de partes dele (CASSANDRAS; LAFORTUNE, 2021). Dois paradigmas usados para representar o comportamento de SEDs são apresentados a seguir, quais sejam: linguagens formais e autômatos.

2.2.1 Linguagens como modelos para SEDs

Como a dinâmica dos SEDs é regida pela ocorrência de eventos, é possível representar o comportamento desses sistemas por sequências de eventos (símbolos) definidos sobre um alfabeto Σ. Essas sequências finitas de eventos (concatenação de símbolos) formam palavras e um conjunto de palavras forma uma linguagem. Como exemplo de linguagem, para o alfabeto $\Sigma = \{a, b, c\}$, pode-se formar a linguagem $L_1 = \{\varepsilon, a, bc, abb\}$. Uma palavra que não contém eventos é chamada de palavra vazia e é representada por ε.

Tendo introduzida a noção de linguagem, é importante apresentar algumas operações normalmente usadas sobre essas linguagens. Seja Σ um alfabeto e $s = uvw$ uma cadeia, com $u, v, w \in \Sigma^*$ (Σ^* define a linguagem máxima com todas as palavras possíveis com o alfabeto Σ). Então, diz-se que: u é um prefixo de s, v é uma subcadeia de s, e w é um sufixo de s.

Uma operação sobre linguagens que merece destaque é a de concatenação. Considerando que $L_1, L_2 \subseteq \Sigma^*$, a concatenação de L_1 e L_2 é definida como $L_1L_2 := \{s \in \Sigma^* : (s = s_1s_2)$ e $(s_1 \in L_1)$ e $(s_2 \in L_2)\}$. Já a operação *Fecho de Kleene*, denotada por *, é formalmente definida como $L^* := \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$. Assim, o conjunto de todas as palavras de comprimento finito que podem ser formadas com os símbolos de um alfabeto Σ, incluindo a palavra vazia ε, é denotado por Σ^* (CASSANDRAS; LAFORTUNE, 2021). Por exemplo, para $\Sigma = \{a, b, c\}$, tem-se que $\Sigma^* = \{\varepsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, \ldots\}$.

A operação denominada de prefixo-fechamento resulta em uma linguagem com todas as cadeias de Σ^* que são prefixos da linguagem usada na operação, i.e., para $L \subseteq \Sigma^*$, tem-se que $\overline{L} := \{s \in \Sigma^* : (\exists t \in \Sigma^* [st \in L])\}$.

2.2.2 Autômatos como modelos para SEDs

O comportamento de um SED também pode ser representado por autômatos de estados finitos, aos qual chamaremos apenas de *autômatos*. Um autômato pode representado por meio de uma quíntupla $G = (X, \Sigma, \delta, x_0, X_m)$, sendo que (CASSANDRAS; LAFORTUNE, 2021):

- X é o conjunto finito de estados do autômato;
- Σ é o conjunto com todos os eventos que definem o alfabeto;
- $\delta : X \times \Sigma \to X$ é a função de transição;
- x_0 é o estado inicial do autômato;
- X_m é o conjunto de estados marcados ou finais, com $X_m \subseteq X$.

Uma característica importante da função de transição considerada na teoria de SEDs é a de que ela é possivelmente parcial, ou seja, podem não haver transições de alguns estados com alguns eventos do alfabeto. Outro aspecto importante a ser destacado é a noção de estados marcados, que na teoria de SEDs representam os estados nos quais uma tarefa do sistema é concluída (CASSANDRAS; LAFORTUNE, 2021).

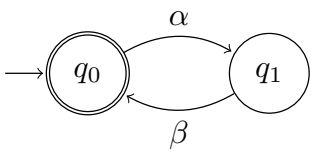
Sendo assim, um autômato G pode ser associado a duas linguagens $L(G) := \{s \in \Sigma^* : \delta(x_0, s)$ é definido$\}$ e $L_m(G) := \{s \in \Sigma^* : \delta(x_0, s) \in X_m\}$. Chamada de linguagem gerada, $L(G)$ representa todos os comportamentos fisicamente possíveis de acontecer no SED G. Já a linguagem marcada $L_m(G)$ corresponde aos comportamentos de G que representam a finalização de tarefas, de modo que $L_m(G) \subseteq L(G) \subseteq \Sigma^*$.

Os autômatos podem ser representados de forma gráfica por diagramas de transição de estados, nos quais círculos são usados para representar os estados e arcos ligando dois estados denotam as transições entre eles. O estado inicial é simbolizado por uma seta apontando para o referido estado, sem um estado de origem. Já os estados marcados são representados por círculos concêntricos. Na Figura 14 ilustra-se um autômato com dois estados e duas transições, sendo que $X = \{q_0, q_1\}$, $\Sigma = \{\alpha, \beta\}$, $x_0 = q_0$, $X_m = \{q_0\}$, $\delta(q_0, \alpha) = q_1$ e $\delta(q_1, \beta) = q_0$.

Ainda usando o autômato da Figura 14 como exemplo, as linguagens gerada e marcada por ele são: $L(G) = \{\varepsilon, \alpha, \alpha\beta, \alpha\beta\alpha, ...\}$ e $L_m(G) = \{\varepsilon, \alpha\beta, \alpha\beta\alpha\beta, ...\}$. Logo, um sistema pode ser modelado e representado por um autômato, ou, equivalentemente, por duas linguagens que apresentam o mesmo comportamento.

Com o autômato determinado também é possível estabelecer definições e operações com autômatos.

Figura 14 – Exemplo de Autômato G.



Fonte: O autor (2023).

Acessibilidade: um estado x é dito acessível se $x = \delta(x_0, u)$ para alguma cadeia $u \in \Sigma^*$. Ou seja, o estado x pode ser atingido a partir do estado inicial do autômato. Se todos os estados de um autômato são acessíveis, diz-se que o autômato é acessível. Podemos deletar de G todos os estados que não são alcançáveis de x_0 por alguma cadeia em $L(G)$ sem afetar as linguagens geradas e marcadas por G. Quando deletamos um estado, também deletamos todas as transições que estão ligados a esse estado. Essa operação é denotada por $Ac(G)$.

Coacessibilidade: um estado x é dito coacessível se a partir dele é possível alcançar um estado marcado. Se todos os estados de G são coacessíveis, então G é dito ser coacessível. Denotamos a operação de deletar todos os estados de G que não são coacessíveis por $CoAc(G)$.

Trim: Um autômato que é acessível e coacessível é chamado de *trim*. A operação Trim é definida como $Trim(G) := CoAc[Ac(G)] = Ac[CoAc(G)]$.

Bloqueio: um autômato G é dito bloqueante caso tenha algum estado que seja alcançável a partir do estado inicial, mas que, a partir dele, não seja possível atingir um estado marcado. Note que a operação $Trim(G)$ remove de G todos os seus estados bloqueantes. Um autômato G é não bloqueante se $\overline{L_m(G)} = L(G)$.

Autômato não Determinístico: um autômato é dito ser não determinístico caso exista um estado com duas ou mais transições de saída etiquetadas com o mesmo evento e que levam a estados distintos.

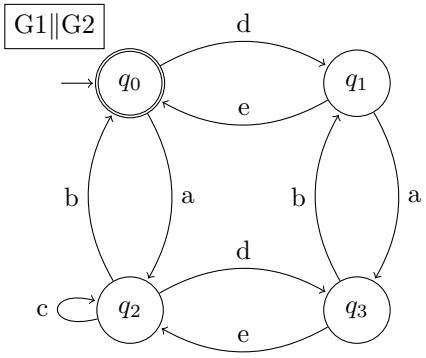
Composição Síncrona de Autômatos: uma das operações mais importantes sobre autômatos é a composição síncrona, denotada por $\|$. Nessa operação, todos os eventos em comum entre os autômatos devem ser executados sincronamente, enquanto os demais ocorrem assincronamente. Sejam por exemplo os autômatos $G1$ e $G2$ ilustrados na Figura 15, cujos alfabetos são $\Sigma_1 = \{a, b, c\}$ e $\Sigma_2 = \{c, d, e\}$, respectivamente.

Figura 15 – Autômatos para exemplo da composição síncrona.

Fonte: O autor (2023).

O autômato resultante da composição síncrona $G = G1 \| G2$ é apresentado na Figura 16.

Figura 16 – Autômato resultante da composição síncrona entre $G1$ e $G2$.



Fonte: O autor (2023).

2.2.3 Teoria de Controle Supervisório

A Teoria de Controle Supervisório (TCS) distingue o sistema a ser controlado, comumente chamado de planta e denotado por G, do elemento de controle, denominado de *supervisor* e denotado por S (RAMADGE; WONHAM, 1987; RAMADGE; WONHAM, 1989). Além disso, na TCS Σ é dividido em dois subconjuntos disjuntos, $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$ (a união é disjunta é denotada por $\dot{\cup}$), de acordo com a controlabilidade dos eventos. Assim, Σ_c é o conjunto de eventos controláveis, que corresponde aos eventos que podem ser impedidos de ocorrer por um agente externo, e Σ_u é o conjunto de eventos não controláveis, que não podem ser impedidos ou desabilitados pelo agente externo (CASSANDRAS; LAFORTUNE, 2021).

O papel do supervisor consiste em observar os eventos gerados pela planta e, a partir dessa observação, desabilitar eventos controláveis para garantir que o comportamento do sistema em malha fechada esteja de acordo com o desejado. Na etapa de projeto, esse comportamento desejado é estabelecido por intermédio das especificações de controle, também chamadas de restrições de controle.

Na TCS, tanto o comportamento da planta quanto das especificações de controle são modelados por autômatos. Quando um SED é composto por vários subsistemas, é possível modelar individualmente cada um deles por intermédio de um autômato G_i, com $i = 1, 2, ..., n$, sendo n o número de subsistemas que compõem a planta. De forma semelhante, pode-se optar por obter um único modelo que represente a especificação global de controle, denotada por E, ou pode-se optar por subdividir as especificações de controle em diversos modelos E_j, com $j = 1, 2, ..., m$, sendo m o número de especificações de controle.

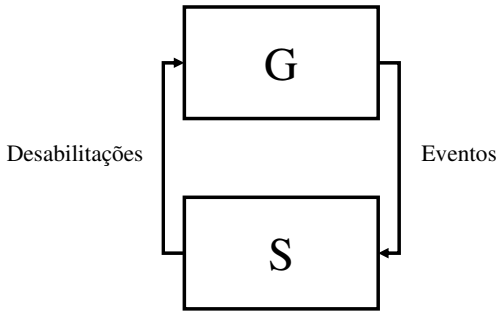
A seguir, apresentam-se algumas abordagens existentes para a síntese de supervisores.

2.2.4 Abordagem monolítica de síntese de supervisores

Na chamada abordagem monolítica de síntese de supervisores, o objetivo é obter um único supervisor que contemple todas as restrições de controle impostas no projeto (RAMADGE; WONHAM, 1987). Assim, a estrutura de controle é composta por uma planta G e um supervisor

S interagindo numa estrutura de malha fechada, conforme ilustrado na Figura 17.

Figura 17 – Estrutura de Controle da Abordagem Monolítica.



Fonte: Cassandras e Lafortune (2021).

Nesse caso, é necessário estabelecer um único modelo que represente o comportamento global do sistema em malha aberta e, caso o projetista tenha modelado os subsistemas isoladamente, utiliza-se a composição síncrona de todos os modelos para obter esse modelo global, i.e., faz-se $G = G_1 \| G_2 \| ... \| G_n$, com n sendo o número de subsistemas. Da mesma forma, todos os autômatos usados para modelar as especificações de controle são compostos para obter um autômato que representa a especificação global de controle, ou seja, faz-se $E = E_1 \| E_2 \| ... \| E_m$, em que m representa o número de especificações de controle.

Com G e E definidos, calcula-se o autômato $R = G \| E$, cuja linguagem $K = L_m(R)$ representa o comportamento fisicamente possível da planta G e que atende às especificações de controle E, sendo conhecida também por *linguagem-alvo*. A linguagem K é dita controlável em relação a $L(G)$ e Σ_u, se $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$. Em palavras, K é controlável se para toda cadeia que pertence ao seu prefixo-fechamento, e para todo evento não controlável $\sigma \in \Sigma_u$ que pode ocorrer na planta após essa cadeia, tal ocorrência leva a um comportamento previso na linguagem-alvo.

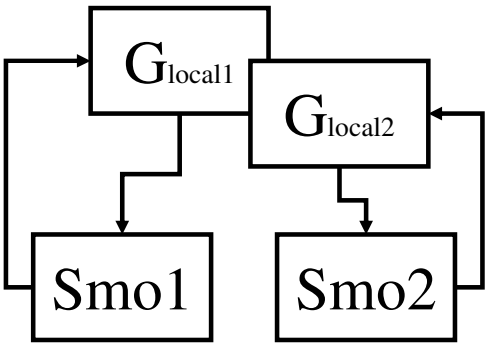
De acordo com Ramadge e Wonham (1989), se K é controlável, então existe um supervisor S, minimamente restritivo e não bloqueante para G, tal que $L_m(S/G) = K$, sendo que S/G representa o comportamento da planta sob a ação do supervisor. Ainda, se K não é controlável, pode-se calcular a máxima sublinguagem de K que é controlável em relação à $L(G)$, denotada por $SupC(G,K)$.

2.2.5 Abordagem modular local de síntese de supervisores

No intuito de explorar a modularidade das especificações de controle e, com isso, evitar a explosão combinatória de estados que ocorre na abordagem monolítica, Queiroz e Cury (2000) propuseram a *abordagem modular local* de síntese de supervisores. Nessa abordagem são usados diversos supervisores que, ao atuarem em conjunto sobre a planta, garantem que o comportamento do sistema em malha fechada obedeça as especificações de controle. A estrutura de controle modular local proposta por Queiroz e Cury (2000) é ilustrada, para o caso hipotético de dois supervisores, por intermédio da Figura 18. Nessa estrutura, cada supervisor observa

parte do conjunto de eventos da planta e, com base nisso, desabilita um conjunto de eventos controláveis no intuito de garantir o cumprimento de uma especificação de controle local. Assim, a cada instante, apenas os eventos controláveis que não foram desabilitados por nenhum dos supervisores é que podem ocorrer na planta.

Figura 18 – Estrutura de Controle Modular Local.



Fonte: Modificado de Queiroz e Cury (2002).

Nessa abordagem, ao invés de calcular um modelo global para a planta G, para cada especificação de controle E_j calcula-se uma planta local G_{local_j} compondo-se os modelos dos subsistemas que são afetados por essa especificação. Após, calcula-se $R_j = G_{local_j} \| E_j$ para $j = 1,2,...,m$, sendo m o número de especificações de controle. Vale destacar que a linguagem-alvo local é dada por $K_j = L_m(R_j)$. De forma análoga à feita antes, calcula-se a máxima sublinguagem de K_j que é controlável em relação à G_{local_j} fazendo $SupC(G_{local_j}, K_j)$ e o supervisor que representa esse comportamento é dado por S_{local_j}.

Após calcular os supervisores modulares, é importante realizar o teste de modularidade (ou de não conflito) entre eles, o que consiste na verificação de que a atuação conjunta dos supervisores não levará a uma ação bloqueante. Para esse teste, pode-se obter um autômato resultante da composição síncrona de todos os supervisores locais e então verificar se nesse autômato não existe nenhum estado bloqueante. Se não existir tal estado, os supervisores são não conflitantes e podem ser empregados de forma modular, com a garantia de que a ação conjunta será minimamente restritiva e não bloqueante. Caso haja conflito, ele deve ser solucionado, o que algumas vezes pode ser feito pela inclusão de uma especificação de controle voltada para esse fim (TEIXEIRA; CURY; QUEIROZ, 2011).

Considerando o caso de dois supervisores não conflitantes, tem-se que $L_m(S_{local_1}/G_{local_1}) \| L_m(S_{local_2}/G_{local_2}) = L_m(S/G)$, sendo $L_m(S/G)$ o resultado obtido na abordagem monolítica (QUEIROZ; CURY, 2000).

Para ambas as abordagens, monolítica e modular local, todos os cálculos para obtenção dos supervisores podem ser realizados com auxílio da ferramenta Nadzoru (PINHEIRO et al., 2015). Esse *software* disponibiliza ferramentas e uma interface gráfica para o desenvolvimento de modelos em autômatos, síntese de supervisores, além de outras funções para a simulação de autômatos, análise de problemas de implementação e geração de código.

2.2.6 Aspectos relacionados à implementação da estrutura de Controle Supervisório

Um ponto importante para a aplicação prática da TCS é a implementação da estrutura de controle supervisório. Dependendo da abordagem de síntese de supervisores adotada, é possível utilizar diferentes alternativas para essa implementação. Para isso, é importante analisar alguns desafios para um funcionamento adequado dos supervisores. No trabalho de Fabian e Hellgren (1998), que aborda sobre a implementação voltada para Controladores Lógicos Programáveis (CLPs), são apresentados alguns problemas que podem ocorrer na implementação da estrutura de controle supervisório, alguns ainda pertinentes para implementação em outros dispositivos de controle.

Efeito Avalanche: esse efeito se manifesta quando a ocorrência de um evento provoca mais de uma transição de estado na lógica de controle, problema esse que é mais comumente identificado na linguagem *Ladder* para CLP.

Problema da Escolha: em estados do supervisor em que há mais de um evento controlável habilitado, é necessária a determinação de qual desses eventos deverá ocorrer naquele estado. Dependendo da estrutura do autômato, essa escolha pode levar o sistema à um comportamento bloqueante, mesmo com o supervisor não bloqueante.

Dentre esses problemas, há outros relacionados principalmente com implementação em CLP, que ainda podem estar presentes dependendo da estratégia de implementação em outros dispositivos, como o problema de simultaneidade (FABIAN; HELLGREN, 1998). Para as implementações dessa dissertação, foi usado como base o trabalho de Lopes et al. (2012), que trata da implementação de supervisores em microcontroladores.

A implementação apresentada por Lopes et al. (2012) se baseia na elaboração análoga a uma estrutura de dados do tipo dicionário, construindo através da função transferência do supervisor (δ). Essa proposta busca uma alternativa para implementação de supervisores para o mínimo de esforço no processamento de microcontroladores. Após a construção do dicionário, é possível analisar apenas os eventos presentes no estado atual do sistema para realizar uma nova transição.

3 ARQUITETURA E PROJETO DE SISTEMA DE COORDENAÇÃO DE FROTAS DE ROBÔS

Este capítulo apresenta os requisitos dos sistemas que a arquitetura visa atuar, demonstrando qual o funcionamento esperado e os tipos de tarefas que tem em vista tratar. Apresenta também uma proposta de arquitetura de coordenação de frotas de robôs móveis autônomos heterogêneos. Apresenta, por fim, propostas de soluções para cada módulo da arquitetura de coordenação proposta.

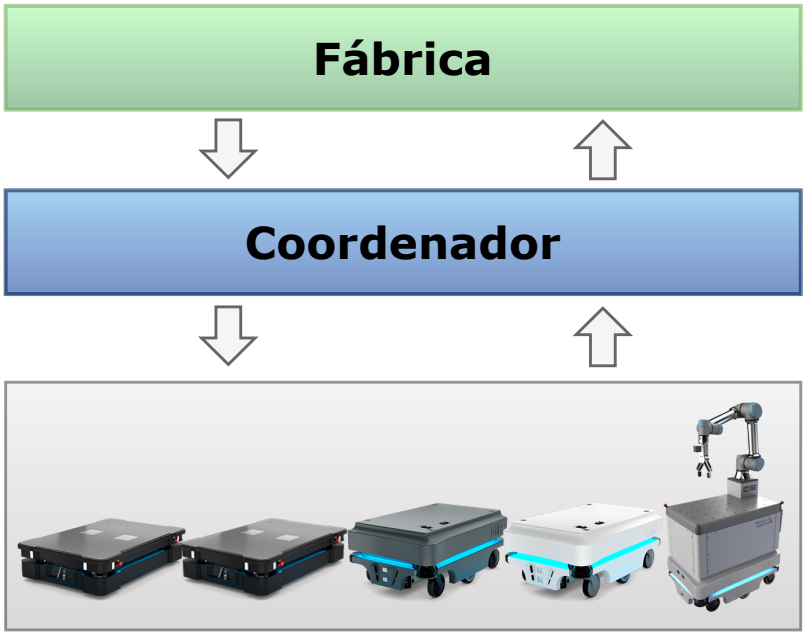
3.1 ESTRUTURA DO SISTEMA FABRIL COM ROBÔS MÓVEIS

Em sistemas fabris, os robôs móveis são utilizados principalmente para tarefas de transporte e logística em pontos específicos, podendo ser em máquinas, depósitos ou posições estabelecidas pela fábrica. Com isso, é necessário um sistema capaz de receber as demandas da fábrica, como tarefas e missões em locais específicos, e, a partir delas, alocar robôs para atender essas demandas.

Esses sistemas fabris podem apresentar dois principais agentes, intermediados pelo coordenador. O primeiro agente é a fábrica, responsável por fornecer e requerer tarefas. Além disso, por meio de um projetista, a fábrica define as políticas das tarefas e quais robôs devem realizá-las. O programador de tarefas é o funcionário ou o usuário da fábrica responsável por preparar as tarefas no sistema, conforme ISO (2021). Por fim, em uma frota de robôs móveis, cada um possui autonomia de locomoção e capacidade de desvio de obstáculos. Além disso, pode haver mais de uma unidade de cada tipo de robô, ou seja, a frota pode ser composta por vários grupos de robôs separados por tipos.

Na Figura 19 é apresentada a arquitetura na qual se insere o coordenador de frotas de robôs em sistemas fabris. O coordenador deve receber as tarefas, as informações da fábrica e as políticas escolhidas pelo projetista, e, com isso, deve atuar a fim de garantir a excussão conforme as necessidades do sistema, distribuindo as tarefas para as frotas de robôs considerando as políticas estabelecidas e visando obter a melhor eficiência possível.

Figura 19 – Sistema Fabril com Frota de Robôs Móveis.



Fonte: O autor (2023).

3.2 TIPOS DE TAREFAS

Dentre as tarefas atribuídas aos robôs móveis, as propostas e exemplos tratados neste trabalho se baseiam em tarefas de transporte e logística. Com isso, são propostos dois tipos genéricos de tarefas, sem detalhar quais produtos são transportados ou o que cada robô realiza nos pontos de interesse, mas com foco na alocação dos robôs para a execução das tarefas.

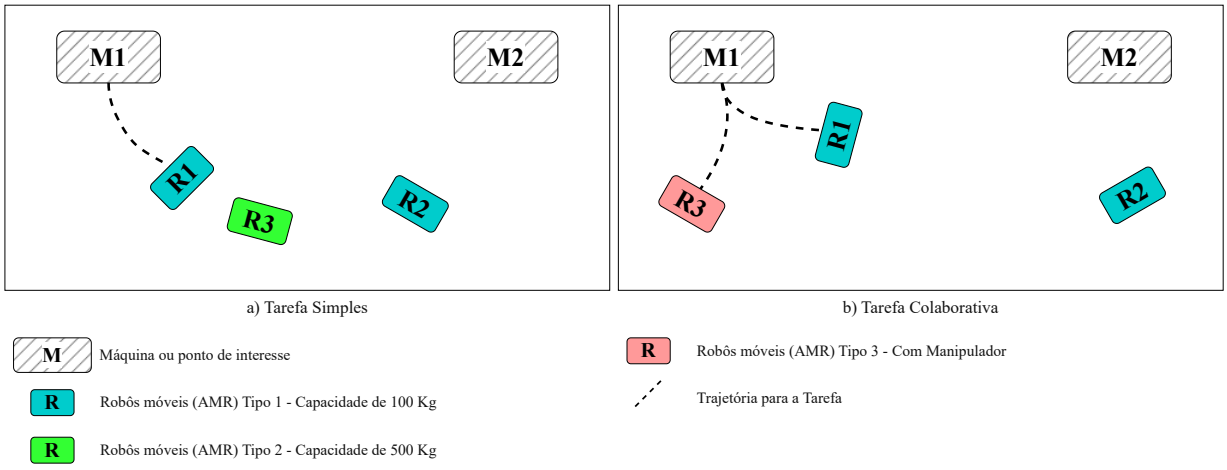
Na Figura 20 são apresentados dois cenários propícios para tarefas com AMRs. Os pontos de interesse para as tarefas, que podem ser máquinas ou repositórios, são representados pela letra "M". Já os robôs são representados pela letra "R". Consideram-se três tipos de robôs, sendo os do tipo 1 (em azul) e tipo 2 (em verde) robôs com capacidade de carga diferentes e o do tipo 3 (em vermelho) um robô equipado com um manipulador. Considera-se ainda que podem haver vários robôs de cada tipo.

3.2.1 Tarefa simples com prioridade

Considera-se como tarefa simples aquela que pode ser realizada por um único AMR. No entanto, é possível que a tarefa possa ser realizada por mais de um tipo de robô existente, sendo necessário estabelecer uma política de prioridade entre os tipos de robôs disponíveis. Essas ordens de prioridade devem ser estabelecidas conforme a necessidade da fábrica, considerando a disponibilidade e as características de cada tipo de robô. Um exemplo prático para esse cenário seria um ambiente com vários robôs de transporte de modelos diferentes, com capacidades de carga distintas.

Por exemplo, na Figura 20 a) ilustra-se um cenário contendo três robôs, dois deles do

Figura 20 – Exemplo de cenário para tarefas, a) robôs de tipos diferentes podendo realizar uma tarefa específica e b) robôs de tipos diferentes realizando uma tarefa em conjunto.



Fonte: O autor (2023).

tipo 1, destacados em azul (R1 e R2, com capacidade de carga de 100 kg) e um, em verde, do tipo 2 (R3, com capacidade de carga de 500 kg). Ao receber uma tarefa simples de transporte de um produto de 50 kg, qualquer um dos robôs teria capacidade para realizá-la. No entanto, o programador de tarefas pode estabelecer uma prioridade para os robôs do tipo 1, de modo que os robôs do tipo 2 só sejam atribuídos à essa tarefa se não houver disponibilidade de robôs do tipo 1. Para esse exemplo, a tarefa só será rejeitada caso não exista disponibilidade de ambos os tipos de robôs.

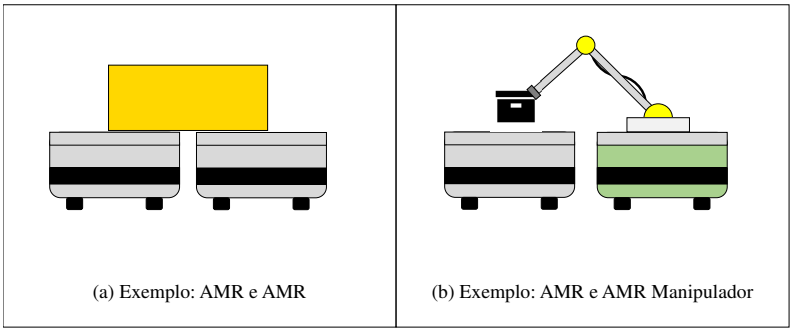
Na Figura 20 b), é apresentado um cenário com dois tipos de robôs diferentes, um para transporte (tipo 1) e outro para manipulação (tipo 3), destacado em vermelho claro. Nesse caso, para realizar uma tarefa colaborativa, é necessária a colaboração entre um robô do tipo 1 (robôs R1 e R2) e um robô do tipo 3 (robô 3). Na referida Figura 20 b), ilustra-se a realização da tarefa pelos robôs R1 e R3. Se não houver disponibilidade de um desses tipos de robôs, a tarefa não pode ser realizada naquele momento.

3.2.2 Tarefa colaborativa

Além das tarefas de transporte simples, em muitas aplicações a colaboração entre dois robôs é essencial para realizar uma tarefa específica. Por exemplo, a combinação de um robô manipulador com outro equipado com uma base de transporte possibilita a realização de uma tarefa na qual o robô com manipulador possa carregar produtos no robô de transporte para descarregá-los em outro local.

Diferentes tarefas colaborativas podem ser enquadradas nesse tipo, como as representadas na Figura 21. Em uma situação como a representada na Figura 21(a), são necessários dois AMRs para realizar o transporte de uma carga mais pesada. Já na Figura 21(b), quem empilha cargas sobre o AMR de transporte é outro AMR que possui um manipulador.

Figura 21 – Exemplos de Tarefas Colaborativas.



(a) Exemplo: AMR e AMR

(b) Exemplo: AMR e AMR Manipulador

Fonte: O autor (2023).

3.3 PROPOSTA DE ARQUITETURA DE COORDENAÇÃO

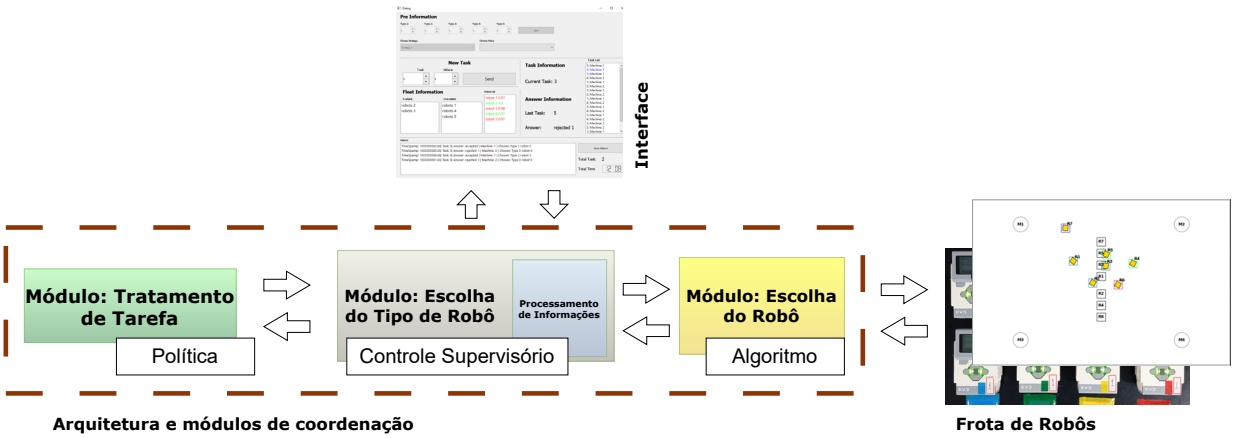
A maioria dos trabalhos encontrados na literatura sobre coordenação de robôs móveis divide a arquitetura em dois ou três módulos, todos focados em aplicações específicas. Dentre os trabalhos com essa característica, podem-se citar os de Ju e Son (2020), Liu, Ficocelli e Nejat (2015) e Li et al. (2008). Em todas as arquiteturas analisadas nos trabalhos relacionados, há pelo menos um módulo específico para o tratamento de tarefas (com objetivo de garantir a execução da tarefa ou ainda determinar qual será submetida ao sistema), alguns com soluções através do controle supervisório, com destaque para os trabalhos de Ju e Son (2020), Liu, Ficocelli e Nejat (2015) e Simon et al. (2023).

Nesta dissertação, propõe-se uma arquitetura dividida em três módulos de coordenação: Tratamento de Tarefas; Escolha do Tipo de Robô; e Escolha do Robô. Cada módulo tem seu objetivo e solução próprios para a garantir o comportamento esperado de todo o sistema. As diferenças entre essa proposta com os trabalhos já destacados, são as funções dadas para cada módulo de coordenação, principalmente especificando um módulo para determinar qual tipo de robô executará a tarefa. Além disso, essa arquitetura propõe tratar a demanda de tarefas com uma totalidade, desde o requerimento de várias tarefas e o gerenciamento da frota para a execução delas.

Na Figura 22 é apresentada a arquitetura completa com a interação entre os módulos e as propostas de soluções deste trabalho. A tarefa é inicialmente encaminhada para o módulo de Tratamento de Tarefas (*Task Handling*), onde é estabelecida uma lista com todas as tarefas submetidas ao sistema, passando para a Escolha do Tipo de Robô (*Choice of Robot Type*) que determina, pela política de prioridade escolhida, qual ou quais os tipos de robôs devem executar a tarefa. Por fim, a Escolha do Robô (*Choice of Robot*) determina qual ou quais os robôs mais adequados para realizarem a tarefa, considerando a posição e/ou o nível de carga da bateria de cada robô.

Os três módulos que compõem o coordenador devem operar em conjunto para garantir o funcionamento do sistema. Na Figura 23 são detalhadas as interações entre os módulos. O módulo de Escolha do Tipo de Robô (ET) é o que estabelece a interação entre todos os módulos,

Figura 22 – Arquitetura de coordenação proposta e suas interações com a interface e frota de robôs.

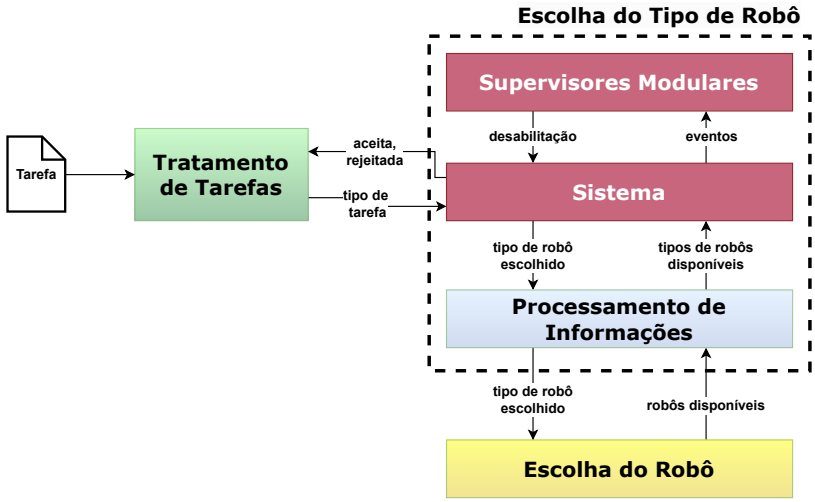


Fonte: O autor (2023).

recebendo a tarefa e estabelecendo se é aceita ou rejeitada para o Tratamento de Tarefas (TT) e, se aceita, encaminha a tarefa ao módulo Escolha do Robô (ER).

Na Figura 23, detalha-se o módulo de Escolha do Tipo de Robô, o qual é composto por três blocos, seguindo a estrutura de controle supervisório proposta proposto por (QUEIROZ; CURY, 2002). No bloco dos Supervisores Modulares são implementados os supervisores que estabelecem as políticas de controle de prioridade entre os tipos de robôs que executam cada tarefa. Já o bloco denominado de Sistema, destacado em vermelho, é destinado à implementação de código para representar o comportamento dos subsistemas que compõem a planta, com todas as tarefas e possibilidades de escolha dos tipos de robôs. Já em azul, destaca-se uma etapa de processamento de informações, a qual é usada para adaptar as informações de disponibilidade dos robôs para estabelecer as abstrações dos eventos necessários para estabelecer o sistema em SED.

Figura 23 – Interação entre os Módulos da Arquitetura.



Fonte: O autor (2023).

3.3.1 Módulo de Tratamento de Tarefas - TT

O módulo de Tratamento de Tarefas é responsável por receber as requisições de tarefas enviadas pela fábrica e por compor uma lista dessas tarefas. Assim, o módulo determina qual será a tarefa submetida ao sistema e, a partir da resposta se foi aceita ou rejeitada, determina as ações específicas dependendo da resposta.

Nesse módulo é possível adotar diversas políticas de tratamento, para esse trabalho buscou-se tratar as tarefas por ordem de chegada, mas considerando as respostas dadas anteriormente. Assim, ao receber uma nova tarefa, é adicionada ao final da lista de tarefas, com um dos comportamentos possíveis sendo o de fila, sendo analisada uma de cada vez e podendo ser aceita ou rejeitada pelo módulo de Escolha do Tipo de Robô. Nesse caso, é possível adotar diversas políticas, tais como: prioridades para determinado tipo de tarefa sobre as demais; se uma tarefa for rejeitada, o sistema testa a tarefa seguinte; entre outras propostas. Essas propostas de soluções, buscam atender a determinados critérios que possam ser adotados pelo usuário/fábrica. Os critérios podem ser adotados visando, por exemplo, o aumento na eficiência do uso da frota, buscando não deixar os robôs ociosos.

3.3.2 Módulo de Escolha do Tipo de Robô - ET

O módulo de Escolha do Tipo de Robô é responsável por determinar qual (ou quais, no caso de uma tarefa colaborativa) tipo(s) de robô(s) deverá(ão) executar a tarefa. Ao tratar de uma tarefa que pode ser realizada por mais de um tipo de robô, esse módulo deve considerar os critérios de prioridades entre os tipos de robôs especificados pelo projetista. Essas políticas e prioridades devem ser definidas para cada tarefa existente no sistema, considerando as tarefas simples e colaborativas.

O módulo ET é uma das principais contribuições desta dissertação para a coordenação de frotas de robôs heterogêneas. Assim, ao ser proposto um módulo intermediário entre o gerenciamento de tarefas e a escolha do robô, esse módulo consegue distinguir e tratar os vários tipos de robôs que podem existir em uma frota de robôs.

Ainda considerando frotas heterogêneas, o responsável por elaborar as tarefas (usuário, projetista ou programador de tarefas) pode adotar diversas políticas de prioridade de escolhas de tipos de robôs para a realização da tarefa. Com isso, o módulo de Escolha do Tipo de Robô determinará qual tipo de robô deve realizar cada tarefa, dependendo das políticas previamente escolhidas.

Vale destacar que, conforme ilustrado nas Figuras 23 e 22, existe uma etapa de processamento de informações para esse módulo. Essa etapa existe devido à necessidade de abstração de algumas informações para a geração de determinados eventos. Por exemplo, há eventos relacionados à disponibilidade dos tipos de robôs, não especificamente dos robôs individuais.

3.3.3 Módulo de Escolha do Robô — ER

O módulo de Escolha do Robô é responsável por determinar, dentre os diversos robôs de um determinado tipo, qual robô é o mais adequado para realizar a tarefa requisitada. Para essa escolha, o módulo deve considerar as características individuais dos robôs. Ao tratar de tarefas colaborativas, o módulo escolhe dois robôs dentre os tipos de robôs determinados previamente.

Uma vez que o módulo de ET tenha definido qual tipo de robô deve executar a tarefa, o módulo de Escolha do Robô analisa todos os robôs desse tipo para determinar qual deles é o mais adequado para a execução da tarefa. Esse módulo pode utilizar diversos algoritmos de otimização e de escolha, como o algoritmo guloso, podendo considerar características individuais como a posição e o nível de bateria de cada robô.

3.3.4 Estruturas de Sistemas de coordenação

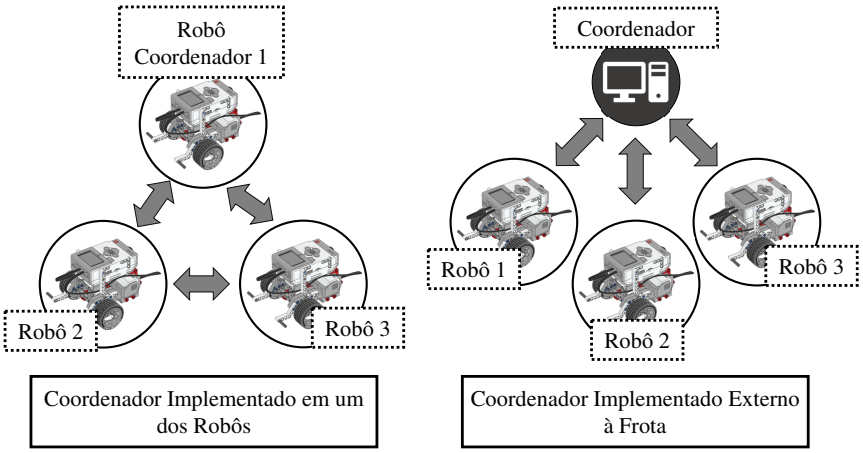
Segundo Verma e Ranga (2021), o gerenciamento de sistemas multirrobôs pode ser garantido com uma estrutura de coordenação centralizada, decentralizada ou hibrida. Com o coordenador centralizado, a tomada de decisão parte de um único robô ou dispositivo coordenador, enquanto que na estrutura descentralizada o sistema é composto por robôs independentes para tomar as decisões entre si. Ainda, para alguns sistemas são adotadas ambas estruturas de coordenação (centralizada e descentralizada), com partes do sistema garantidas de modo centralizado e outras tomadas de decisão de modo individuais, essas estratégias são denominadas híbridas.

Ao considerar a proposta de sistema de coordenação já apresentada, será adotado uma estrutura híbrida para implementação. Com o coordenador centralizado tendo a responsabilidade de receber e alocar as tarefas para os robôs da frota. Enquanto cada robô possui a autonomia para determinar a suas trajetória no ambiente e a realização da tarefa.

O local onde o coordenador é implementado pode definir características importantes ao sistema. Na Figura 24 são apresentadas duas estratégias de implementação do coordenador, sendo que na primeira delas o coordenador está em um dos robôs móveis pertencentes à frota. Essa abordagem traz a vantagem de não precisar de um dispositivo específico para tratar o coordenador. Já na segunda abordagem, o coordenador é implementado em um dispositivo específico.

Para essa dissertação foi adotada a estratégia do coordenador implementado externo a frota, em um computador dedicado às atribuições, visto que com um coordenador centralizado possui a facilidade para estabelecer a comunicação e as interações com a frota de robôs e as tarefas requisitadas.

Figura 24 – Estratégias de Implementação do Coordenador.



Fonte: O autor (2023).

3.4 SOLUÇÕES PARA OS MÓDULOS DE COORDENAÇÃO

Tendo definido a estrutura do coordenador, pode-se agora tratar das soluções para os seus módulos, adotando políticas, estratégias ou algoritmos de tomada de decisão próprias para cada um deles.

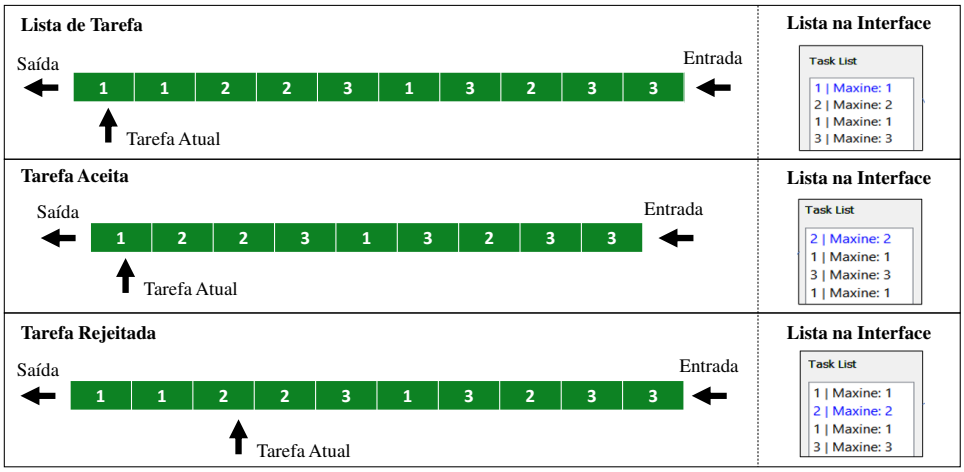
3.4.1 Políticas para o módulo de Tratamento de Tarefas

Para a proposta do módulo TT consideram-se as responsabilidades e funcionamento estabelecidos anteriormente. A solução apresentada nessa dissertação se baseia na ordem de chegada à fila de tarefas e uma política definida para quando uma tarefa é aceita ou rejeitada, visando otimizar o uso da frota de robôs, evitando deixar robôs ociosos.

Na Figura 25 é apresentada a política adotada e implementada na plataforma para o gerenciamento das tarefas aceitas e rejeitadas. Na parte superior da figura, ilustra-se a lista de tarefas, ao centro ilustra-se a situação de uma tarefa aceita e, por fim, na parte inferior da figura ilustra-se o caso de uma tarefa rejeitada. Ao aceitar uma tarefa, já definido qual o tipo de robô irá executá-la, ela é retirada da lista de tarefas e atribuída ao módulo de Escolha do Robô. Quando uma tarefa é rejeitada, passa-se a analisar uma tarefa de tipo diferente, ainda respeitando a ordem de chegada no sistema. Por exemplo, na parte inferior da figura ilustra-se a situação na qual, ao ser rejeitada a tarefa 1, ao invés de tratar a tarefa seguinte, que é do mesmo tipo, é feita a análise da tarefa 2, respeitando a ordem na fila. Se nessa análise a tarefa 2 é aceita, volta-se para a primeira tarefa da fila (tarefa 1), caso contrário, segue-se para a próxima tarefa de tipo diferente (tarefa 3), e assim sucessivamente.

Ao testar todos os tipos de tarefas do sistema ou analisar a última tarefa da lista, retorna-se para o início da fila e recomeça-se a análise, desconsiderando as informações anteriores sobre as tarefas analisadas. Com isso, é garantido o funcionamento contínuo da plataforma considerando a lista de tarefas requeridas. Outro aspecto importante é que o módulo faz essa análise na lista

Figura 25 – Política para o Tratamento de Tarefas.



Fonte: O autor (2023).

de tarefas somente se houver pelo menos um tipo de robô disponível. Caso a frota de robôs disponível no momento não tenha capacidade de executar uma das tarefas da lista, o coordenador espera algum dos outros tipos de robôs retornar a disponibilidade para retomar a analisar as tarefas.

3.4.2 Supervisores para o módulo de Escolha do Tipo de Robô

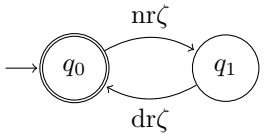
Com os tipos de tarefas considerados, foram definidas duas tarefas-base a serem tratadas: Tarefa Simples com Prioridade e a Tarefa Colaborativa. O módulo ET tem como missão garantir o funcionamento de políticas estabelecidas sobre a prioridade de escolha entre os tipos de robôs existentes na frota. É considerado que o usuário/empresa poderá adicionar novas tarefas ou remover tarefas dependentes das situações desejadas, característica essa que traz uma dificuldade para garantir a confiabilidade para soluções com adição dinâmica de tarefas no sistema. Assim, a corretude do comportamento lógico do módulo é garantida por intermédio do emprego da Teoria de Controle Supervisório para a sua obtenção.

Para a proposta de solução desse módulo são apresentados modelos genéricos de planta e de especificações de controle, considerando os dois tipos de tarefas. A cada tarefa nova do sistema, seja ela simples ou colaborativa, é adotado um novo conjunto de autômatos composto por todos os modelos de subsistemas e especificações, aos moldes dos modelos genéricos apresentados. Para a elaboração desses modelos, foi importante a análise de alguns trabalhos relacionados ao tema, como os de Florencio et al. (2018), Lopes et al. (2016) e Dulce-Galindo et al. (2019), com as propostas em autômatos detalhados nos trabalhos relacionados. A TCS é utilizada com o intuito de garantir o funcionamento do sistema sem bloqueios, considerando todas as tarefas que podem existir e suas políticas previamente definidas.

Os primeiros modelos a serem apresentados para esse problema são os que representam a disponibilidade do tipo de robô. Na Figura 26 apresenta-se esse modelo genérico, com ζ

especificando qual tipo de robô está sendo representado, sendo que ζ assume valores sobre o conjunto $\zeta \in \{A, B, C, ...\}$.

Figura 26 – Modelo Genérico de Disponibilidade para Tipos de Robôs.



Fonte: O autor (2023).

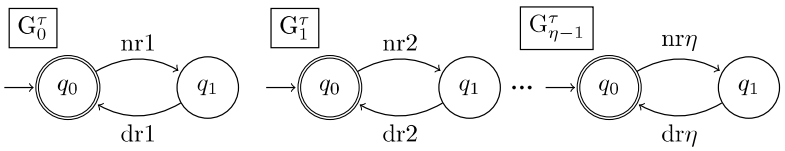
Para todos os modelos de disponibilidade, q_0 representa o estado no qual existe ao menos um robô daquele tipo disponível e o estado q_1 representa a situação de indisponibilidade de robôs do tipo ζ. Já o eventos "drζ" indica que ao menos um robô do tipo ζ ficou disponível e "nrζ" indica que todos os robôs desse tipo ficaram indisponíveis.

3.4.2.1 Solução genérica para tarefas simples

As tarefas simples, que são realizadas por um único robô, normalmente podem ser executadas por mais de um tipo de robô. Assim, é importante estabelecer um sistema de prioridades na escolha de qual tipo de robô deve ser escolhido para cada tipo de tarefa, permitindo assim que seja feito um uso eficiente dos recursos disponíveis. No intuito de apresentar uma solução genérica para a priorização, adota-se a notação $1 > 2 > 3 > ...$ para representar que o tipo 1 de robô tem prioridade sobre o 2 e assim sucessivamente. Mais detalhes sobre essa questão da priorização são apresentados após a introdução dos modelos para as especificações de controle.

O número de autômatos responsáveis em descrever a disponibilidade dos robôs na tarefa será o mesmo número de tipos de robôs capazes de realizá-la. Na Figura 27 são apresentados os modelos, de G_0^τ até $G_{\eta-1}^\tau$, que representam a disponibilidade de todos os tipos de robôs que podem realizar a tarefa τ, sendo que o índice η representa o número de tipos de robôs que podem realizar a tarefa τ. Adota-se aqui a prioridade decrescente estabelecida pelo índice $1, 2, ..., \eta$.

Figura 27 – Modelos genéricos de disponibilidade para Tarefa Simples, com a prioridade decrescente sendo estabelecida pelo índice $1, 2, ..., \eta$.

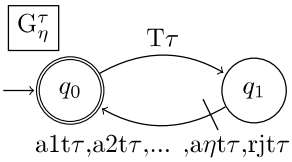


Fonte: O autor (2023).

Na Figura 28 apresenta-se o autômato "G_η^τ", que representa o modelo genérico que descreve o comportamento de uma tarefa simples. As transições etiquetadas com nomes iniciados com a letra "*a*" denotam eventos controláveis que representam a escolha (por parte do módulo ET) de um determinado tipo de robô. Assim, ao receber uma tarefa (evento Tτ), o módulo

escolhe um tipo de robô para realizá-la (eventos a1tτ, ..., aηtτ), ou então a rejeita (evento rjtτ). Esses eventos seguem um padrão na nomenclatura que corresponde a ordem de prioridade estabelecida pelo projetista, ou seja, a1tτ > ... > aηtτ.

Figura 28 – Modelo genérico de que descreve o comportamento de uma Tarefa Simples.



Fonte: O autor (2023).

Na Tabela 1 é apresentada a descrição dos eventos usados na modelagem referente às tarefas simples. Os eventos relacionados à disponibilidade são representados pelos eventos do tipo com menor prioridade (drη e nrη), assim como os eventos de escolha (aηtτ).

Tabela 1 – Descrição dos Eventos de Tarefas Simples com Prioridade.

Eventos	Descrição	Contr.	Obs
drη	Robôs com prioridade η Ficam Disp.	Não	Sim
nrη	Robôs com prioridade η Ficam Indisp.	Não	Sim
Tτ	Recebeu a Tarefa τ	Não	Sim
aηtτ	Escolhe o tipo com prioridade η para a Tarefa	Sim	Sim
rjtτ	Rejeita a Tarefa τ	Sim	Sim

Fonte: O autor (2023).

Para determinar as especificações de controle é importante que elas respeitem determinadas condições. De modo geral, todos os modelos seguem uma estratégia similar:

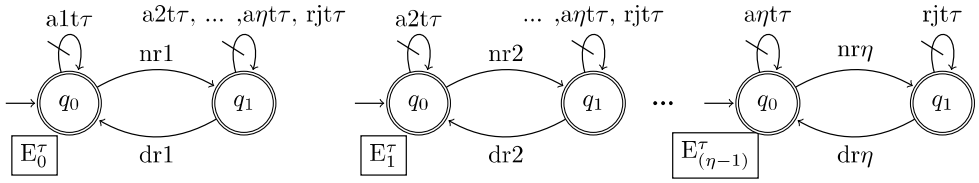
1. Um tipo de robô só pode ser escolhido se ele estiver disponível (se houver ao menos um robô deste tipo disponível);

2. Somente quando não houver disponibilidade de um tipo de robô é que se pode escolher um tipo de robô de prioridade menor; ainda, somente nessa situação uma tarefa pode ser rejeitada.

Os modelos apresentados para as especificações de controle (ver Figura 29) estabelecem uma ordem de prioridade de escolha para determinar qual tipo de robô deverá realizar a tarefa. A prioridade é estabelecida considerando todas as especificações de controle da Figura 29 tratadas em conjunto. Essa prioridade é definida pelo projetista, definindo qual o tipo ganhará a maior prioridade (prioridade 1) e qual terá a menor (prioridade η).

Com esses modelos, é estabelecido um método para o projetista criar as tarefas que deseja no sistema. Para as tarefas simples, primeiro se deve determinar quais os tipos de robôs podem executar essa tarefa, para então determinar os modelos de disponibilidade. Seja por exemplo um

Figura 29 – Especificações de Controle para a Tarefa Simples.



Fonte: O autor (2023).

cenário com dois tipos de robôs (B e C) com capacidade de realizar uma tarefa, com a prioridade para robôs do tipo B. Nesse caso, a partir dos modelos genéricos, haverá três modelos de planta (G0, G1 e GT3) e duas especificações de controle (E0 e E1). Para estabelecer a devida ordem de prioridade, os eventos genéricos "nr1" e "dr1", que denotam maior prioridade, serão associados (mudarão a nomenclatura) aos eventos "nrB" e "drB" do robô tipo B, enquanto "nr2" e "dr2", que indicam menor prioridade, serão associados aos eventos "nrC" e "drC" do robô tipo C.

Assim, o projetista determina a ordem de prioridade relacionando modelos de disponibilidade dos tipos de robôs (nrA, drA,...) aos de disponibilidade por prioridade (nr1, dr1, ...), para então, posteriormente, com os modelos e especificações de controle, realizar as operações necessárias para obter os supervisores.

3.4.2.2 Solução genérica para tarefa colaborativa

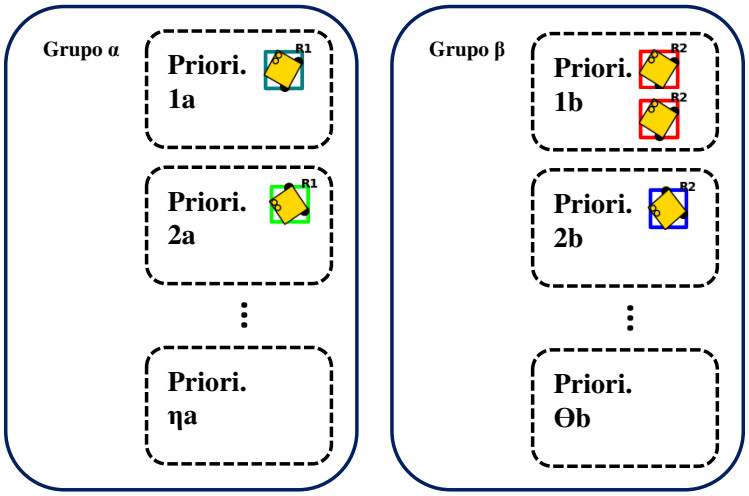
No que tange às tarefas colaborativas, considera-se que elas podem ser realizadas por dois grupos de robôs, denotados por α e β, sendo que cada um deles pode ser formado por robôs de diferentes tipos, conforme ilustrado na Figura 30. Denota-se por η o número de tipos de robôs pertencentes ao grupo α, enquanto θ representa o número de robôs do grupo β. Nesse caso, também é necessário estabelecer uma ordem de prioridade entre os tipos de robôs que fazem parte de um mesmo grupo. De forma análoga à questão de priorização adotada para as tarefas simples, adota-se agora a notação $1a > 2a > ... > \eta a$ e $1b > 2b > ... > \theta b$ para representar a prioridade dos tipos de robôs dentro dos grupos α e β, respectivamente. Assim, robôs do tipo $1a$ têm prioridade para realizar parte da tarefa colaborativa e robôs do tipo $1b$ têm prioridade para realizar a outra parte da mesma tarefa colaborativa.

Na Figura 31 são apresentados dois conjuntos de autômatos que modelam, respectivamente, a disponibilidade de robôs dos grupos α e β.

Nesses modelos genéricos, os eventos de disponibilidade são relacionados com as prioridades entre os tipos de robôs de um mesmo grupo, logo $nr1a$ e $dr1a$ está relacionado ao tipo de robô com maior prioridade para realizar a tarefa do grupo α. Isso é elaborado com a ordem de prioridade estabelecida pelo projetista, até chegar em $nr\eta a$ e $dr\eta a$, sendo o tipo de menor prioridade para o grupo α. O mesmo ocorre para os eventos relacionados a disponibilidade do grupo β.

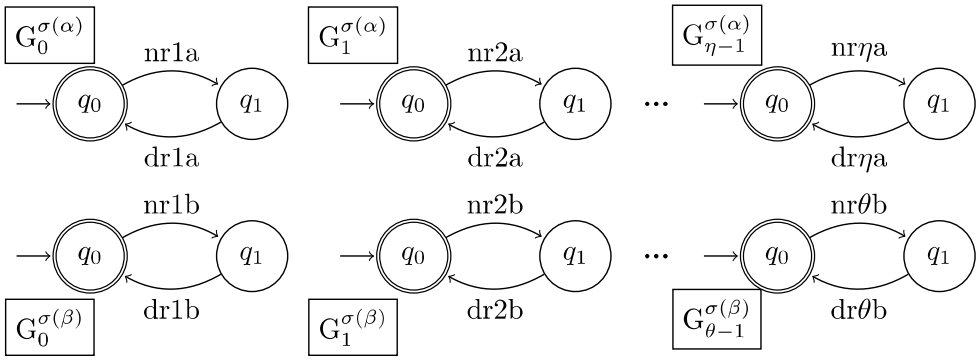
Considere, por exemplo, um cenário com uma tarefa colaborativa que, dentro do grupo α,

Figura 30 – Divisões em grupos (α e β) dos tipos de robôs que podem executar a Tarefa Colaborativa.



Fonte: O autor (2023).

Figura 31 – Modelos genéricos de disponibilidade para Tarefa Colaborativa, com a prioridade decrescente sendo estabelecida pelos índices $1a, 2a, ..., \eta a$ para o grupo α e $1b, 2b, ..., \eta b$ para o grupo β.



Fonte: O autor (2023).

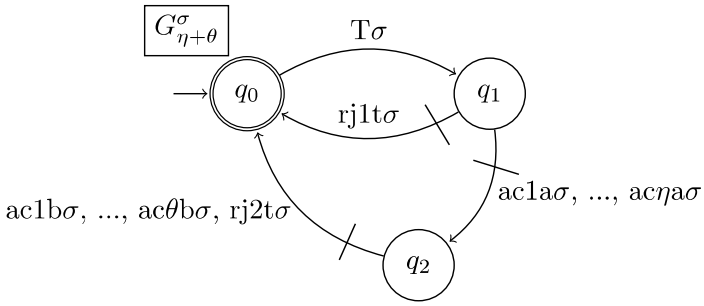
deve ser escolhido um robô do tipo A ou do tipo B (tipo A sendo prioritário) e, dentro do grupo β só existem robôs do tipo C. Então, $nr1a$ e $dr1a$ serão renomeados para os eventos do modelo de autômato do robô tipo A (nrA e drA). Ainda, os eventos $nr2a$ e $dr2a$ serão renomeados para os eventos do modelo de autômato do robô tipo B (nrB e drB) e, por fim, os eventos $nr1b$ e $dr1b$ serão renomeados para nrC e drC, respectivamente.

Na Figura 32 é apresentado o autômato $G^{\sigma}_{\eta+\theta}$, que descreve uma tarefa colaborativa σ a ser realizada por dois grupos de robôs, α e β, nessa ordem. Os eventos similares a "acηa" são os eventos de escolha de tipos de robôs do grupo α, com η sendo o número de tipos de robôs desse grupo. Enquanto os similares à "acθb" são os eventos de escolha para β, com "θ" sendo o número total de robôs que compõem o grupo.

Diferentemente da tarefa simples com prioridade, para essa tarefa ser aceita é necessário

passar por duas etapas: ter robôs disponíveis do primeiro grupo α e do segundo grupo β.

Figura 32 – Modelo que descreve uma Tarefa Colaborativa.



Fonte: O autor (2023).

Na Tabela 2 é apresentada a descrição dos eventos usados na modelagem para tarefas colaborativas. Destacando quais eventos são controláveis e não controláveis. Nos demais eventos de disponibilidade dos tipos de robôs do grupo α possuem descrições similares aos de "drηa" e "drηa", assim como "drθb" e "drθb" para o grupo β. Parecidamente, os eventos de escolha também possuem a descrições similares aos "acηaσ" e "acθbσ"

Tabela 2 – Descrição dos Eventos para Tarefas Colaborativas

Eventos	Descrição	Contr.
drηa	Robôs com prioridade η ficam disponíveis	Não
nrηa	Robôs com prioridade η ficam indisponíveis	Não
drθb	Robôs com prioridade θ ficam disponíveis	Não
nrθb	Robôs com prioridade θ Ficam indisponíveis	Não
Tσ	Recebeu a Tarefa α	Não
acηaσ	Escolhe o tipo com prioridade η para a Tarefa	Sim
rj1tσ	Rejeita a tarefa por falta de robôs em α	Sim
acθbσ	Escolhe o tipo com prioridade θ para a Tarefa	Sim
rj2tσ	Rejeita a tarefa por falta de robôs em β	Sim

Fonte: O autor (2023).

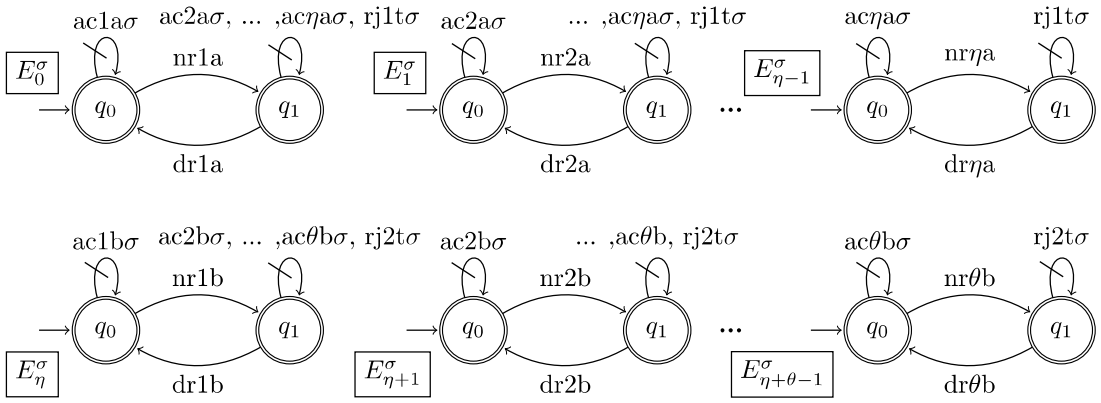
Com a definição dos modelos de planta, é necessário estabelecer as especificações de controle para a tarefa colaborativa. O número de especificações de controle para garantir o comportamento do sistema dependerá da quantidade de tipos de robôs. Assim, o número total de modelos de especificações será igual a η+θ.

1. Um tipo de robô só pode ser escolhido se ele estiver disponível (se houver ao menos um robô deste tipo disponível);

2. Quando não houver robô de um tipo disponível, deve ser possível a escolha de outros tipos de robôs com prioridade menor e que a tarefa seja rejeitada.

As condições são as mesmas de tarefas simples com prioridade, mas as especificações de controle são divididas para os dois grupos e cada um seguindo as mesmas condições estabelecidas. A Figura 33 apresenta todas as especificações.

Figura 33 – Especificações de controle genéricas para Tarefa Colaborativa.



Fonte: O autor (2023).

Para as especificações de controle, é adotada a mesma proposta para lidar com a prioridade de tipos de robôs nos grupos α e β de modo individual, conforme apresentado na Figura 33. Com o conjunto de especificações de controle, é garantido que o sistema respeite as prioridades estabelecidas entre os tipos de robôs dentro de cada grupo α e β.

3.4.3 Solução do módulo de Escolha do Robô

Para a solução do módulo ER foram analisados os algoritmos utilizados para otimização de escolha, os algoritmos desse trabalho se basearam no algoritmo guloso pela facilidade de implementação. Ao receber a tarefa e a informação de qual o tipo de robô deve executar, o módulo de Escolha do Robô analisa qual o robô mais adequado naquele momento. Para a escolha foram estabelecidas duas informações para a determinar o robô mais adequado: bateria e a distância do robô até a tarefa.

$$G_n = P1 * bateria_n + P2 * \frac{1}{distancia_n} \tag{1}$$

Para a aplicação de algoritmos de tomada de decisão, para cada robô é atribuído um ganho que dependerá do valor da bateria e da posição do robô. Esse ganho é determinado pela Equação 1, com as informações individuais de cada robô ($bateria_n$ e $distancia_n$) e os parâmetros "P1" e "P2" que determinam o peso dado entre as características individuais no determinado momento.

Ainda sobre a Equação 1, os parâmetros "P1" e "P2" foram obtidos de maneira empírica, considerando políticas que possam ser adotadas. Nesse ínterim, é possível estabelecer algoritmos que priorizem o valor da bateria dos robôs em execução ou os robôs mais próximos à tarefa

requisitada. Na Tabela 3 é apresentado um exemplo de valores atribuídos a quatro robôs diferentes, considerando valoreis unitários para os parâmetros “P1” e “P2”.

Tabela 3 – Exemplo de Ganhos Atribuídos à Quatro Robôs.

Robô	Distância	Bateria	Ganho	Probabilidade
R1	$0,844$	$0,954$	$0,0^*$	$0,0\%$
R2	$1,109$	$0,954$	$18,60$	$31,95\%$
R3	$0,909$	$0,954$	$20,55$	$35,30\%$
R4	$1,103$	$0,999$	$19,06$	$32,74\%$

* Robô 1 está indisponível no momento

Fonte: O autor (2023).

3.4.3.1 Algoritmo guloso e algoritmo com probabilidade

Com o algoritmo guloso, é sempre escolhido o robô com o maior ganho absoluto no momento. Para o caso apresentado na Tabela 3, o robô escolhido é R3 de maneira direta, já que possui o maior ganho (20,55). Em uma situação de empate, em que dois robôs apresentam o mesmo ganho, o algoritmo escolhe o último robô analisado e atribuído o valor.

Mesmo apresentando o funcionamento esperado com o algoritmo guloso para a escolha do robô, há situações e ambientes que podem exigir condições diferentes, como em uma política na qual é dada prioridade a um robô com menor valor de bateria, para consumir a bateria de uma vez e assim retornar ao carregamento. Pensando nessas situações, foi proposto mais um algoritmo, desta vez baseado na probabilidade de um robô ser escolhido para a tarefa.

Nesse segundo algoritmo, a probabilidade é definida pelo ganho atribuído aos robôs com relação à soma de todos os ganhos dos robôs disponíveis no grupo. Logo, é observado pela Tabela 3 que “R3” tem sim a maior probabilidade de ser escolhido para a tarefa, contudo, os robôs “R2” e “R4” ainda assim podem ser escolhidos para a execução da tarefa.

3.4.3.2 Variações dos algoritmos

Determinando os algoritmos e estabelecidos a implementação, é possível escolher algumas características para essa escolha. Com o intuito de aumentar a liberdade para o programador de tarefas é criado características através dos parâmetros “$P1$” e “$P2$”. Como os parâmetros estão associados diretamente à bateria e à distância do robô até a tarefa, é possível elaborar alternativas que deem mais prioridades a uma característica com relação a outra.

Com esse aspecto, há três variações para cada algoritmo:

- Pesos iguais para as características ($P1 \approx P2$);
- Preferência para robôs com nível de bateria mais elevados ($P1 >> P2$).
- Preferência para robôs mais próximos à tarefa ($P1 << P2$);

Essas variações estabelecem uma proposta na qual a política mais equilibrada para a escolha pode sugerir um funcionamento mais eficiente em mais situações, já a política de priorizar o nível de bateria pode favorecer estratégias que visam garantir sempre um nível de bateria no funcionamento do sistema. Por último, sobre a preferência de distância do robô, de modo mais claro, garante estratégias para os robôs percorrerem o menor percurso e realizando o máximo de tarefas possíveis, enquanto possuírem mais que o mínimo de bateria.

3.5 CONCLUSÃO DO CAPÍTULO

Nesse capítulo foram apresentadas as soluções e alternativas para cada módulo da arquitetura proposta. Para o Tratamento de Tarefas, uma proposta com condições que consideram a ordem de requerimento e a resposta dada pelo coordenador. Enquanto para a Escolha do Tipo de Robô, são propostos modelos genéricos para gerenciar os tipos de robôs conforme as tarefas necessárias do sistema. Por fim, apresentados dois algoritmos para a Escolha do Tipo de Robô, um baseado em algoritmo guloso (escolhendo o robô com maior valor no momento) e um algoritmo com probabilidade conforme as características momentâneas do robô.

4 IMPLEMENTAÇÃO DA PLATAFORMA

Este capítulo apresenta a implementação do coordenador por meio de uma plataforma, destacando todas as ferramentas implementadas e as possibilidades de testes com o uso corretamente. A plataforma possui três ramificações: a interface com o usuário, o cenário com o simulador Robotarium e o cenário com robô LEGO EV3.

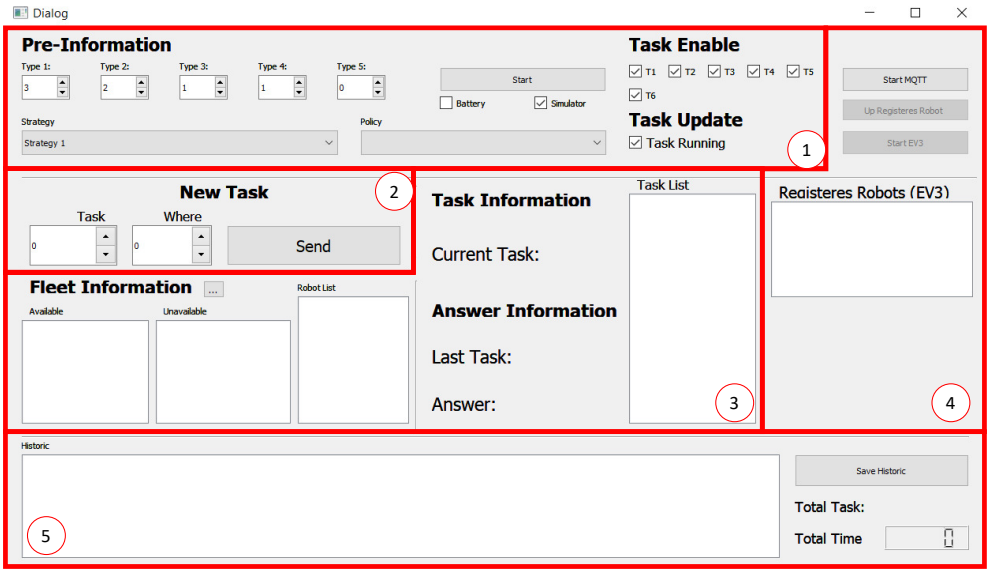
4.1 INTERFACE

Para existir uma interação do usuário com o sistema, a plataforma se faz o uso do PyQT5, o QT é um conjunto de bibliotecas em alto nível que possibilitam desenvolvimento de aplicativos e bibliotecas em diversas plataformas, o PyQT são os recursos do QT para desenvolvimento em Python (COMPUTING, 2023).

4.1.1 Interface gráfica da plataforma

Na Figura 34 é apresentada a interface desenvolvida, mostrando todas as ferramentas e informações. Essa interface serve como entrada de informações do sistema, como: o número de tipos de robôs, número de robôs e acrescentar uma tarefa nova a lista, além de mostras as informações de resposta e estado do sistema.

Figura 34 – Interface com Usuário Desenvolvida.



Fonte: O autor (2023).

Ainda na Figura 34 é apresentada uma divisão da interface em cinco áreas diferentes. Cada área representa um conjunto de ferramentas com determinadas funções.

- **Área 1:** Definições em pré-simulação;
- **Área 2:** Adicionar uma nova tarefa à lista;

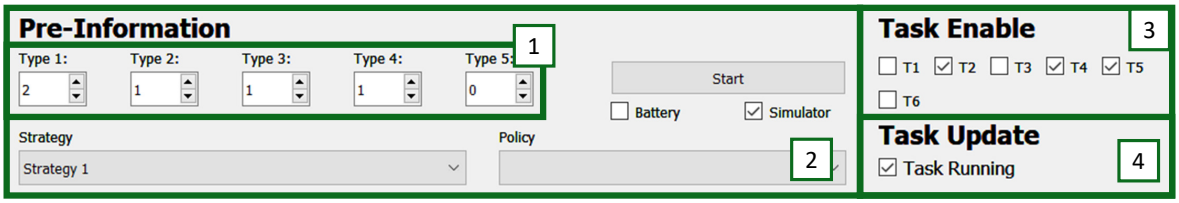
- **Área 3:** Informações atuais do sistema;
- **Área 4:** Ferramentas para robôs EV3;
- **Área 5:** Histórico de respostas do coordenador e informações das Tarefas.

A interface deve disponibilizar recursos para o uso de todas as ferramentas propostas da arquitetura de coordenação, habilitando ou desabilitando conforme as necessidades dos testes a serem realizados. Também deve disponibilizar todos os dados de testes e execuções para futuras análises. Os dados do histórico são salvos em uma documentação no formato de "n^o da Tarefa | *TimeStamp* | Tarefa | Resposta | Lugar | Robô Escolhido".

Na Figura 35 é detalhado os elementos de pré-simulação, no item 1 o usuário poderá estabelecer o número de robôs de cada tipo, podendo escolher a quantidade de robôs disponíveis no cenário. No item 2, há ferramentas para determinar qual estratégia de alocação de tarefas será adotada na implementação, também se será considerada a bateria de cada robô no sistema.

Ainda sobre a Figura 35, no item 3 o usuário tem a liberdade de habilitar somente as tarefas que deseja submeter, entre seis tarefas já implementadas previamente, sendo que quatro delas são do tipo simples com prioridade e outras duas colaborativas. Por fim, o item 4 é a opção dada ao usuário de pausar a alocação das tarefas quando desejar, ideal para observar momentos específicos do sistema.

Figura 35 – Área de Pré-Simulação com as ferramentas dispostas para o usuário.

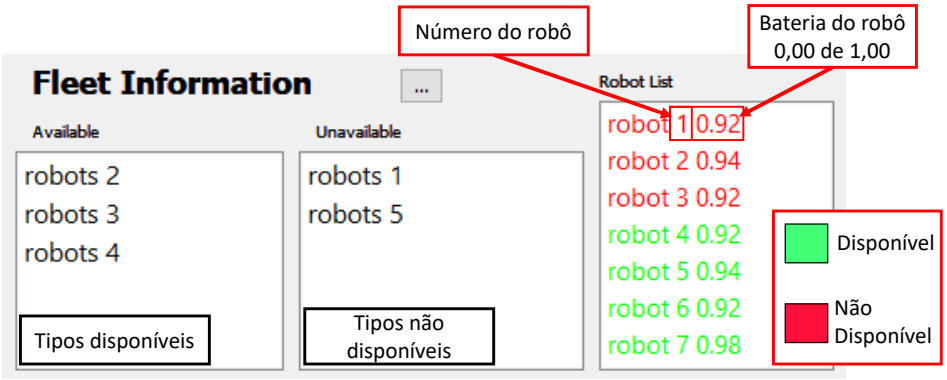


Fonte: O autor (2023).

A Figura 36 apresenta o funcionamento da plataforma para informar sobre a disponibilidade da frota de robôs no momento da análise da tarefa. Como é observado, há três listas diferentes: tipos de robôs disponíveis, tipos de robôs não disponíveis e a lista de robôs. Caso um tipo de robô tenha pelo menos um robô disponível, ele entra na lista de tipos de robôs disponíveis, se não tiver, é adicionado a lista de tipos indisponíveis. Enquanto na lista de robôs estão apresentadas as disponibilidades induviais de cada robô presente na frota (verde está disponível e vermelho indisponível) e o nível de bateria de cada um (0,00 — 0% e 1,00 — 100%).

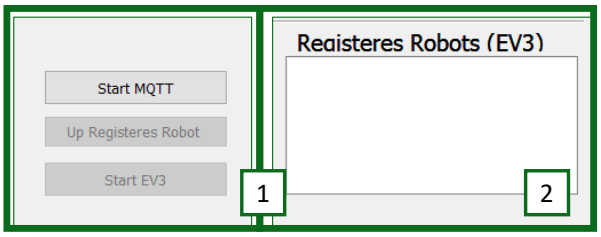
Dentre as outras ferramentas, na Figura 37 apresenta as ferramentas implementadas referente aos robôs da LEGO EV3. No item 1, são botões necessários para o teste com robôs reais, com: inicialização do protocolo de comunicação *MQTT*, visualização dos robôs conectados ao coordenador e a inicialização do sistema respectivamente. Já o item 2, é uma lista com todos os robôs já conectados e prontos para receberem tarefas.

Figura 36 – Informações momentâneas da frota, sobre a disponibilidade dos tipos e informações individuais dos robôs.



Fonte: O autor (2023).

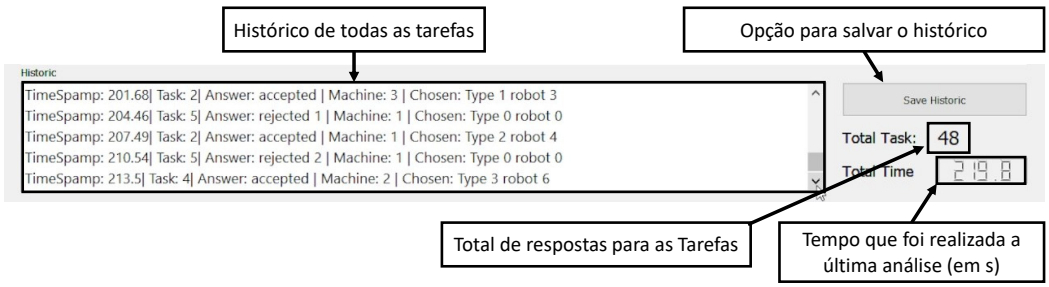
Figura 37 – Área de Ferramentas para a Comunicação MQTT.



Fonte: O autor (2023).

A Figura 38 apresenta todas as informações sobre o histórico de tarefas submetidas ao sistema. A interface disponibiliza as informações de número total de respostas dadas pelo coordenador e o tempo que realizada a última análise. Além de possibilitar que o usuário salve as informações sobre o sistema a qualquer momento.

Figura 38 – Área de Ferramentas para o histórico de respostas para as tarefas.



Fonte: O autor (2023).

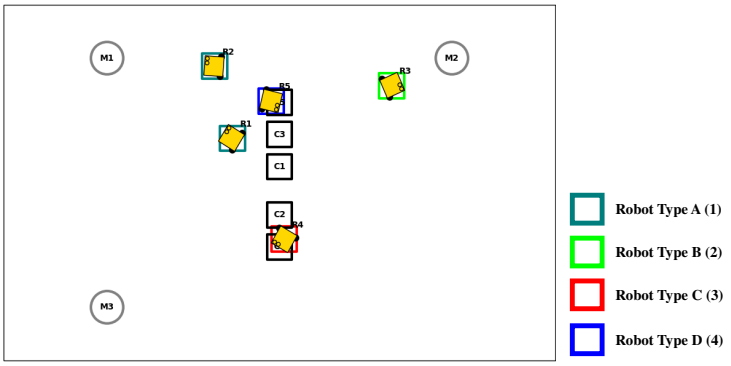
4.2 CENÁRIO NO ROBOTARIUM

O Robotarium (PICKEM et al., 2017) é um simulador desenvolvido para Python e Matlab, disponibiliza algoritmos e ferramentas para controle de frotas de robôs controlando trajetória ou

formação dos robôs, apresentando resultados também por meios gráficos. Os desenvolvedores do simulador também possuem um ambiente real o qual é possível realizar os testes com robôs físicos, os recursos do simulador e informações sobre o ambiente estão disponíveis no site oficial do Robotarium (TECH, 2023).

Na Figura 39 é apresentado o cenário adaptado com o uso do simulador. O cenário é composto pelos robôs, as bases de carregamento e as máquinas (representando os pontos onde são realizadas as tarefas). No cenário estão presentes cinco robôs móveis divididos em quatro tipos diferentes de robôs, contudo a plataforma comporta um número genéricos de robôs de cada tipos.

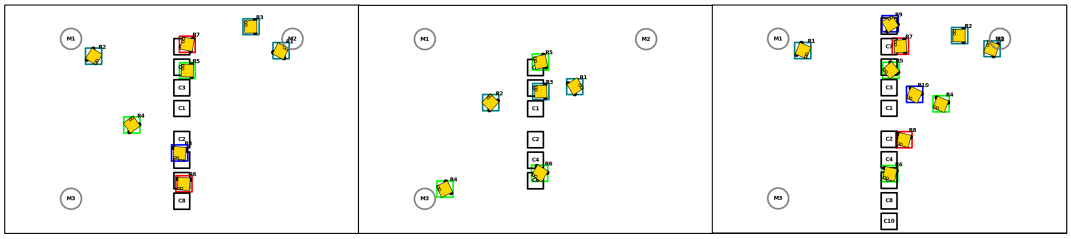
Figura 39 – Cenário do Sistema no Robotarium.



Fonte: O autor (2023).

Um aspecto importante é a flexibilidade de uma plataforma de simulação quanto ao número de robôs disponíveis no sistema. Na Figura 40 apresenta alguns cenários com variações no número e nos tipos de robôs disponíveis no sistema: Figura 40 a) com oito robôs distribuídos em quatro tipos diferentes, Figura 40 b) seis robôs em apenas dois tipos de robôs e Figura 40 c) total de dez robôs distribuídos nos quatro tipos. Com isso, a plataforma ganha a flexibilidade na simulação da frota de robôs, com a possibilidade de escolha de quantos robôs há disponíveis e a distribuição em quatro tipos de robôs diferentes.

Figura 40 – Número de Tipos e de Robôs no Simulador.



a) 8 robôs em 4 tipos b) 6 robôs em 2 tipos c) 10 robôs em 4 tipos

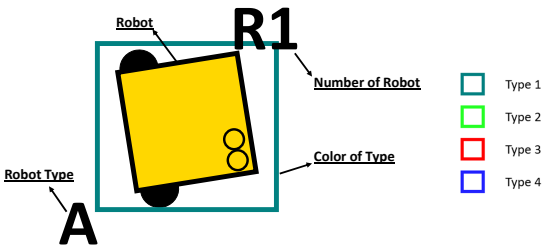
Fonte: O autor (2023).

Todos os cenários do Robotarium seguem determinado padrão para a identificação dos elementos estáticos de cenário. As circunferências identificadas com “M” representam pontos

onde são realizadas as tarefas e os quadrados com identificação “R” é a base de carregamento para cada robô móvel do sistema.

Já os robôs são identificados em amarelo, com legendas de identificação de número e seu tipo. Na Figura 41 é apresentado um robô exemplo, pela cor do quadrado externo e na legenda da parte inferior esquerda é identificado qual tipo pertence, podendo apresentar cores: azul-claro (tipo A), verde (tibo B), vermelho (tipo C) e azul-escuro (D). Também é padronizada a numeração dada a cada robô, os primeiros números (R1, R2, . . .) são dados aos robôs do tipo A, seguindo para o tipo B, C e D, sequencialmente.

Figura 41 – Exemplo e Identificação de Robôs no Robotarium.



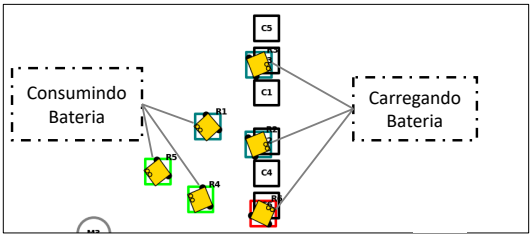
Fonte: O autor (2023).

4.2.1 Simulação da bateria

Para gerar mais informações a serem avaliadas para a escolha de qual robô realizará a tarefa, é necessário o desenvolvimento de aspectos que aproximem de um cenário real, como o estado da bateria e a posição do robô no momento da requisição de tarefa. Com essa necessidade, na plataforma também é simulada a bateria do robô, que pode estar consumindo ou carregando em momentos distintos.

Há dois momentos distintos: o robô pode estar se movimentando e consumindo a bateria ou na base carregando a bateria. Na Figura 42 há um cenário composto por três robôs em carregamento e três consumindo a bateria. Para a simulação da bateria, tanto o carregamento quanto o descarregamento da bateria baseado em um modelo linear, contudo considerando que o tempo que o robô leva para consumir a bateria é menor que o de carregamento.

Figura 42 – Momentos que a Bateria está sendo consumida ou carregada no Simulador.



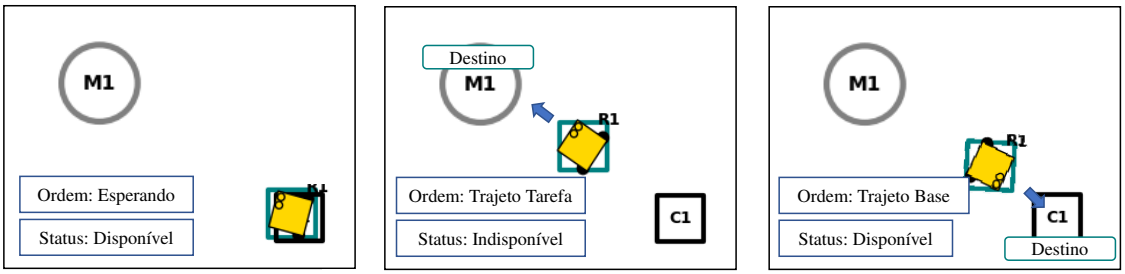
Fonte: O autor (2023).

4.2.2 Tarefas no simulador

Cada robô disponível no simulador possui seus atributos individuais, inclusive a situação do robô: disponível ou indisponível. Estando disponível, o robô pode ser atribuído a uma nova tarefa, independente se está em movimento ou na base. Já indisponível, o robô pode estar executando uma tarefa, em trajeto para a realização de uma ou com bateria insuficiente para executar. As tarefas simuladas se baseiam em atividades de transporte, onde o robô parte de sua posição inicial e se desloca até o ponto de interesse.

Na Figura 43 é apresentada a execução de uma tarefa no simulador, nela o robô "R1" se locomove até o ponto M1 e retorna à sua base. Assim que recebe a tarefa, o robô se apresenta como indisponível até finalizar a tarefa. Na trajetória de M1 até o retorno para a base C1, o robô já fica disponível para novas atribuições.

Figura 43 – Realização de uma tarefa no simulador, com momento antes de receber, a caminho e após realizar a tarefa.



Fonte: O autor (2023).

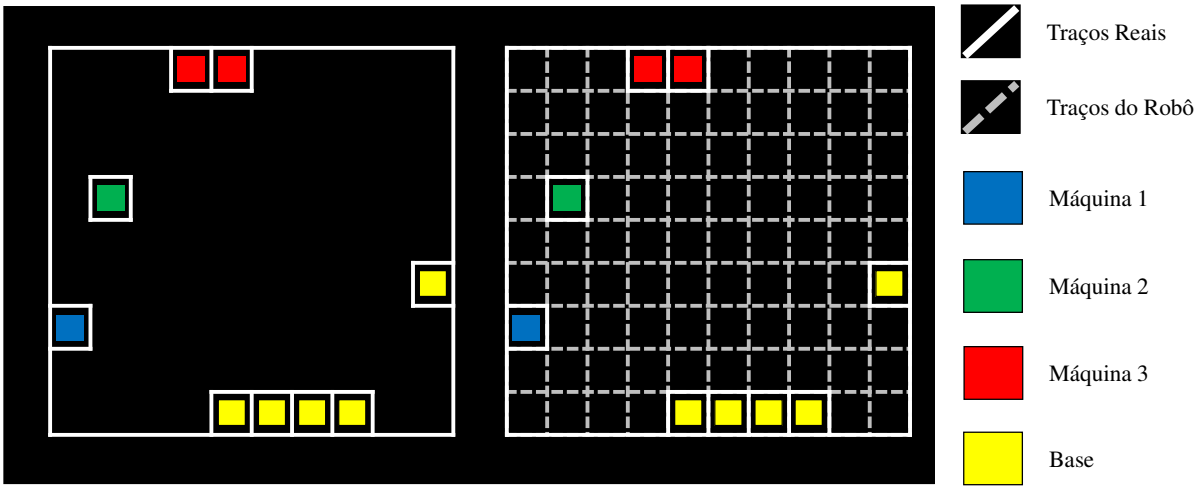
4.3 CENÁRIO COM ROBÔS LEGO

Para o ambiente para testes experimentais, foi adotada a utilização dos robôs da LEGO EV3 Mindstorms. Todos os robôs possuem a mesma construção, logo para a distinção entre os tipos de robôs foi optado pela identificação de cores para determinados testes, simulando a formação de grupos de robôs de tipos diferentes.

Na Figura 44 é apresentada a ideia de ambiente dos testes experimentais com os robôs da LEGO, nele as bases dos robôs (em amarelo) e as máquinas (em azul, verde e vermelho) dispersas no ambiente composto de posições. Todos os testes experimentais possuem o mesmo padrão de elementos, podendo ter mapas maiores e com distribuições diferentes. Para o desenvolvimento dessa dissertação, foi construído um mapa que respeita uma matriz de 9 colunas e 10 linhas, totalizando 90 posições no mapa. O tamanho dessas posições respeita as dimensões do robô, considerando a medida total do ambiente.

A Figura 44 a) é a representação do cenário real montado em laboratório, onde é possível observar os deslocamentos e comportamentos dos robôs. Já a Figura 44 b) é a representação dada ao analisar a perspectiva do robô, com as linhas em cinza sendo as posições no mapa para a determinação da trajetória a ser percorrida para retornar à base ou executar uma tarefa.

Figura 44 – a) Representação do cenário construído em laboratório e b) representação da perspectiva dos robôs.



a) Cenário Real

b) Cenário com as demarcações dos robôs

Fonte: O autor (2023).

Outra diferença é dada para a tarefa colaborativa, para os testes em laboratório foi optado para a realização dessa tarefa, exclusivamente sendo na máquina 3. Nessa máquina é possível analisar na Figura 44 que há duas posições previamente demarcadas. De modo geral, para a realização dessa tarefa, ambos os robôs devem alcançar essas posições, um em cada.

O cenário apresentado na Figura 44 a), construído fisicamente em laboratório, tem o intuito de analisar o comportamento de um sistema coordenado com a arquitetura fora de um ambiente simulado em computador. Para a construção foram utilizados lençóis de borracha formando um ambiente de dimensões 2,50×2,04 m e espessura de 3 mm e fitas brancas para demarcar as limitações, enquanto para as bases, máquinas e obstáculos foram utilizados papéis coloridos para facilitar a identificação.

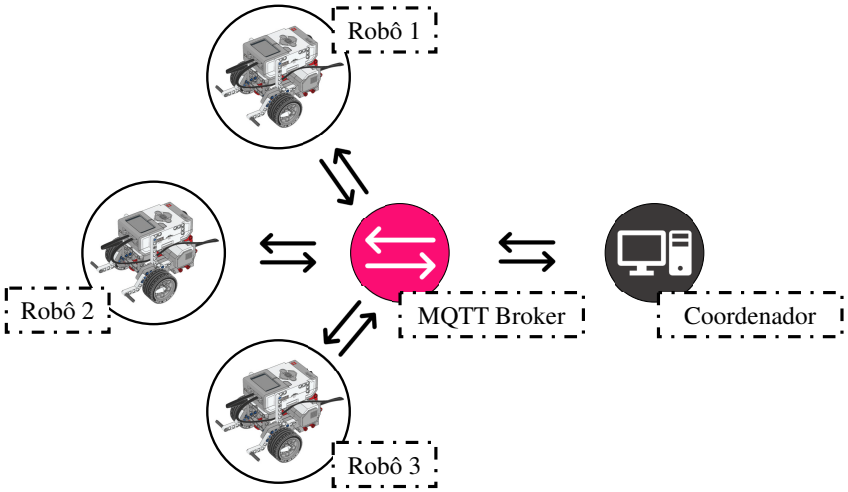
A plataforma foi desenvolvida para ter seu funcionamento garantido em qualquer computador, assim como qualquer rede *Wireless* com capacidade de estabelecer o protocolo de comunicação. Além dos materiais utilizados para montar o ambiente de testes, é importante destacar os dispositivos para a realização dos testes, foram utilizados: um computador com processador Inter Core 5 e sistema Windows 10, um roteador TP-Link TL-WR829N de 300Mbps e 5 kits da LEGO EV3.

4.3.1 Estrutura com protocolo MQTT

Uma das justificativas para o uso do kit da LEGO Mindstorms é a facilidade na implementação do protocolo *MQTT — Message Queuing Telemetry Transport* que possibilita a troca de informações entre o coordenador e os robôs. Foi adotado esse protocolo pela flexibilidade em utilizações em dispositivos com baixa capacidade computacional e a facilidade do uso com multiagentes.

O *Broker* do *MQTT* foi integrado com o coordenador, sendo inicializado também pela plataforma. Para a execução dos testes, todos os robôs e o coordenador foram conectados na rede *Wireless* (pelo protocolo, é possível que os robôs estejam conectados em diferentes redes, contudo para esses testes foi utilizada apenas uma rede *Wireless* tendo em vista o ambiente de testes), para então iniciar a comunicação, essa interação é representada na Figura 45. Todos os robôs devem possuir o mesmo canal de comunicação, que tentam conectar assim que são inicializados. Ao se conectar, o robô recebe do coordenador sua identificação: número do robô e seu tipo (A, B, C ou D). Sabendo sua identificação, o robô é cadastrado somente nos tópicos de seu interesse, como o seu tipo. Com a comunicação estabelecida, o coordenador informa ao robô escolhido qual a tarefa a ser executada e os robôs passam informações para o coordenador: posição, disponibilidade e bateria.

Figura 45 – Comunicação entre o Coordenador e os Robôs através do MQTT



Fonte: O autor (2023).

Antes dos testes com a arquitetura completa, primeiro foi realizado um teste para visualizar o funcionamento da comunicação entre o coordenador e a frota de robôs. Esse teste foi realizado apenas com os blocos programáveis da LEGO, com o intuito de assegurar o funcionamento da interação, como apresentado na Figura 46.

Para o teste da Figura 46, primeiro é iniciada a comunicação pelo coordenador, então são todos os robôs são conectados individualmente. Com todos os robôs conectados, o sistema é inicializado com algumas tarefas a serem realizadas e na interface da plataforma são apresentadas as informações de disponibilidade dos robôs individualmente.

4.3.2 Tarefas com robôs LEGO

As tarefas com os robôs LEGO seguem o mesmo objetivo das tarefas simuladas no Robotarium, a realização de transportes e logística. Logo, o robô deve ser capaz de se locomover pelo ambiente e determinar sua rota até o destino da tarefa sem o auxílio do coordenador. Assim,

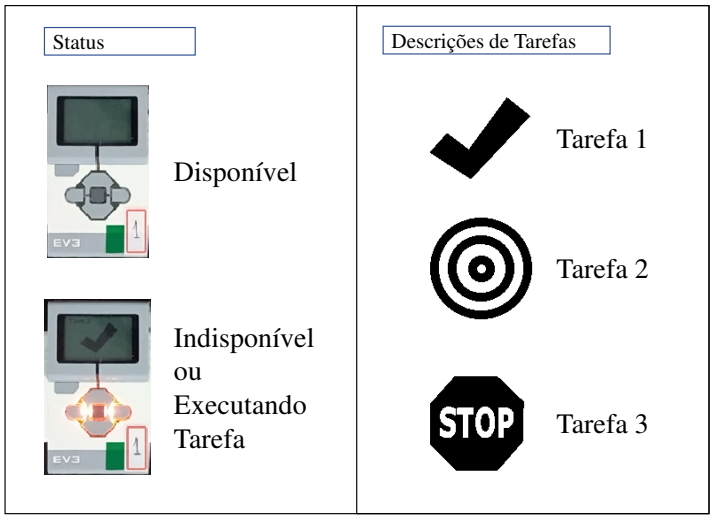
Figura 46 – Testes de comunicação com o MQTT

Fonte: O autor (2023).

para essa dissertação é considerado que o robô possui a autonomia para determinar sua trajetória ou desviar de obstáculos no ambiente, focando no coordenador somente as suas políticas de gerenciamento e alocação das tarefas.

Ao receber uma tarefa, o robô deve identificar qual tarefa deve ser realizada e percorrer a trajetória de sua posição atual até o local da tarefa. Para o cenário experimental apresentado na Figura 46 foi adotado o padrão observado na Figura 47. Para todos os testes da LEGO, foram propostas três tarefas previamente (duas simples com prioridades e uma colaborativa), todas relacionadas aos transportes, mas em posições diferentes no mapa.

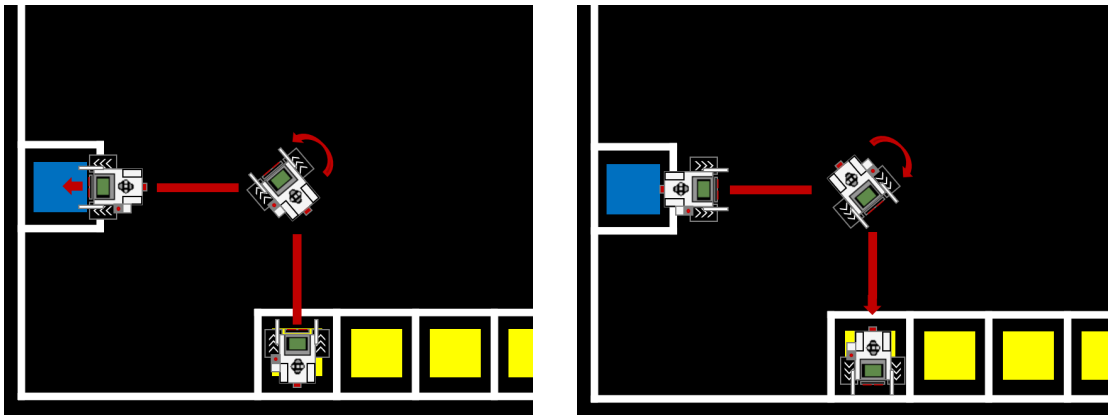
Figura 47 – Padrões para as Tarefas com o Robô LEGO.



Fonte: O autor (2023).

A Figura 48 representa um robô realizando uma tarefa no cenário experimental, com a Figura 48 a) representando o momento inicial que o robô recebe a tarefa e realiza a trajetória até o ponto de execução. Enquanto a Figura 48 b), o robô já realizou a tarefa e retorna à sua base. Observando o comportamento dos robôs no cenário experimental, possui o mesmo padrão estabelecido no simulador.

Figura 48 – Representação do robô da LEGO realizando uma tarefa: a) o momento que recebe a tarefa e se encaminha até o ponto; e b) ao finalizar a tarefa, retorna para sua base.



a) Trajetória até a tarefa b) Trajetória de retorno a base

Fonte: O autor (2023).

4.3.3 Movimentação dos robôs

A movimentação dos robôs são dadas individualmente, cada um determinando sua trajetória a ser percorrida para a execução da tarefa. Para alcançar a máquina para executar a tarefa, o robô percorre pelas posições possíveis no cenário e suas demarcações, desviando de posições com obstáculos e considerando a distância nos caminhos. Para determinar a trajetória, foi utilizado o algoritmo A* (SANTOS et al., 2021), o qual é uma versão simplificada do algoritmo Dijkstra. O algoritmo A* é utilizado por diversos trabalhos para determinar trajetórias para robôs móveis, como no trabalho de Santos et al. (2021).

Cada robô tinha o conhecimento do quanto era necessário se locomoverem para se mover de uma posição para outra, através do uso dos encoders dos motores. Para se direcionar no mapa, foi utilizado o sensor giroscópio para determinar o angulo que deve se locomover. Enquanto para evitar colisões entre robôs e obstáculos, o uso do sensor de distância em frente ao robô determina se há um objeto em sua frente.

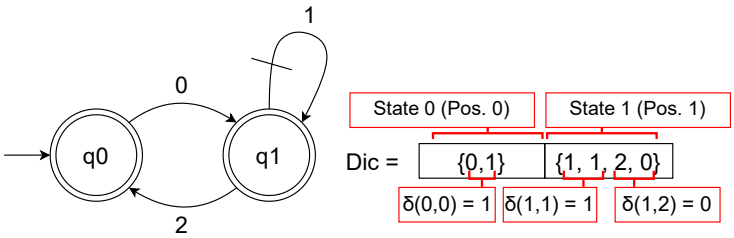
Não foram apresentados grandes detalhes sobre essa implementação nos robôs móveis, pois o objetivo dessa dissertação está na arquitetura de coordenação, que considera que os robôs possuem sua autonomia.

4.4 IMPLEMENTAÇÃO DA ESTRUTURA DE CONTROLE SUPERVISÓRIO NA PLATAFORMA

Um aspecto importante para a plataforma é a capacidade de implementar o controle supervisório com as políticas de gerenciamento dentre os tipos de robôs. Para a implementação das abordagens da Teoria de Controle Supervisório, foi adotada uma estratégia baseada na implementação de supervisores em microcontroladores do trabalho de Lopes et al. (2012), mas adaptado para Python.

Na Figura 49 é apresentado um exemplo da geração de uma estrutura de dados análoga à um dicionário, com um autômato simples composto de dois estados (q0 e q1) e três eventos (0, 1 e 2). Cada elemento do dicionário corresponde a um estado do autômato, com uma lista de todas as transições possíveis nesse estado. Esse processo de geração é aplicado para todos os estados do supervisor, se o supervisor for composto por "n" estados, o dicionário deverá ter "n" elementos. Assim, com o uso de dicionário é analisado as transições presentes no estado atual, melhorando o desempenho do sistema e da plataforma, em comparação a uma estratégia de analisar todas as transições.

Figura 49 – Exemplo de Geração de Dicionário.



Fonte: O autor (2023).

Ainda no exemplo da Figura 49, no estado "q1" o autômato possui duas transições, uma com o evento 1 que mantêm no mesmo estado "q1" e outra com o evento 2 que leva ao estado "q0". Assim, o elemento correspondente ao estado "q1" recebe uma lista {1,1,2,0} que apresenta as transições possíveis do estado. Para a plataforma é necessário ter a disposição um arquivo em formato ".csv" com todas as transições possíveis do autômato que represente o supervisor.

Na Tabela 4 é apresentado um exemplo de arquivo para a geração do dicionário, o arquivo possui o total de três transições que correspondem ao autômato apresentado no exemplo da Figura 49. Com o dicionário pronto, a plataforma pode atualizar o supervisor ao receber uma nova requisição.

Tabela 4 – Exemplo de Arquivo para Implementação Supervisores.

Atual, Evento, Destino
0, 0, 1
1, 1, 1
1, 2, 0

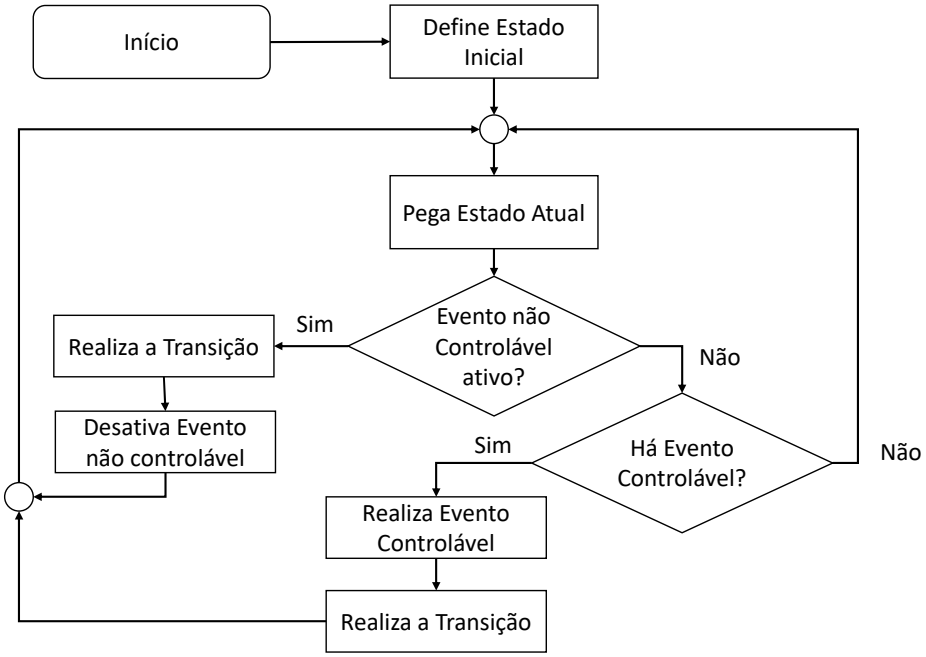
Fonte: O autor (2023).

Também para a implementação é necessário considerar alguns problemas apresentados por Fabian e Hellgren (1998), com o trabalho focado em aplicações em Controladores Lógicos Programáveis (CLP). Dentre os principais problemas, alguns foram estudados e contornados na plataforma como: efeito avalanche e problema da escolha.

Na Figura 50 é apresentado o fluxograma para a atualização do supervisor considerando o estado atual e os eventos ativos. Primeiro são analisados todos os eventos não controláveis

presentes no estado, se algum estiver ativo é realizada a transição para o estado de destino. Caso não tenha eventos não controláveis ativos e exista algum evento controlável, permite a ocorrência do evento controlável e realiza a transição.

Figura 50 – Atualização do Supervisor.



Fonte: O autor (2023).

Ainda na Figura 50, é possível a realização de apenas uma transição por atualização do supervisor e ao realizar uma transição com evento não controlável, o evento é desativado, evitando assim o problema de avalanche. E sobre o problema de escolha, para duas transições de eventos controláveis no mesmo estado, não importa qual transição será realizada primeiro, pois o outro evento será considerado em outra atualização.

A plataforma irá analisar as possíveis transições para atualizar os supervisores apenas se identificar que há tipos de robôs disponíveis suficientes para a realização de uma das tarefas do sistema. Esse critério é importante para evitar que o coordenador rejeite em excesso as tarefas e considere se existe a possibilidade de realizar alguma das tarefas habilitadas.

4.5 CONCLUSÃO DO CAPÍTULO

No capítulo foi apresentada a plataforma e seus recursos para simulação e testes da arquitetura de coordenação proposta nesta dissertação. Estabelecidos a interface com usuário, um simulador de frotas e um modo de implementação de supervisores, é possível propor estudos de casos com finalidade de analisar o funcionamento dos módulos de coordenação.

5 TESTES E VALIDAÇÃO

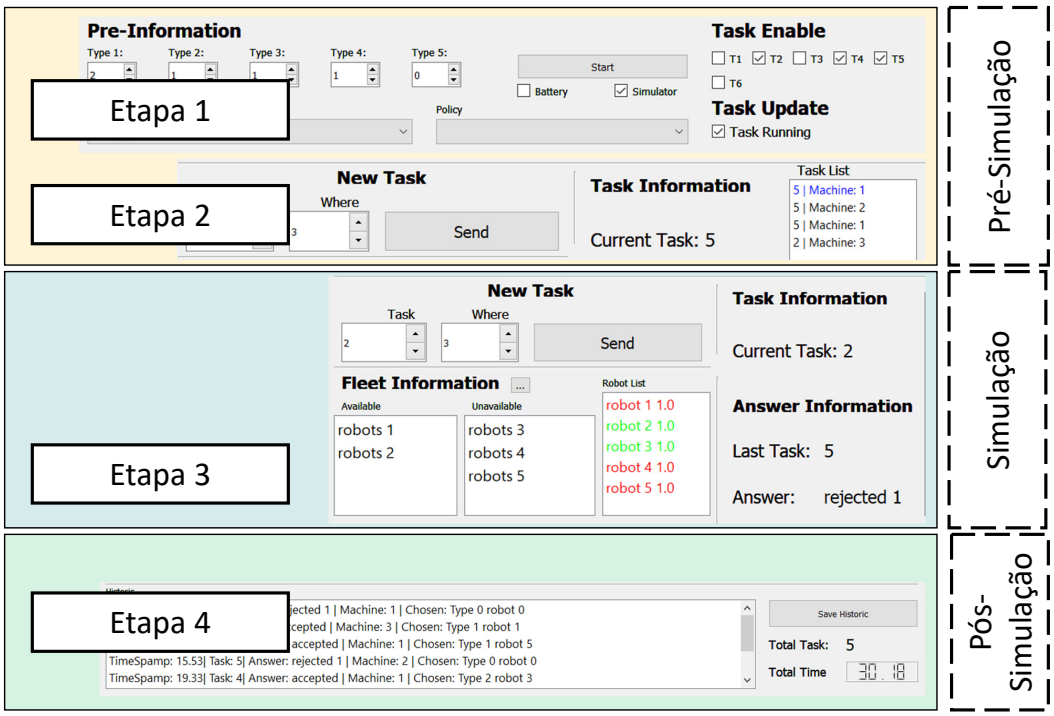
Neste capítulo são apresentados todos os testes realizados para a validação do sistema de coordenação e apresentar seu funcionamento. Para analisar o comportamento com as soluções, são propostos três estudos de casos, alguns realizados somente através do simulador e um deles em conjunto com a realização de testes experimentais com robôs da LEGO.

5.1 PROCESSO PARA REALIZAÇÃO DOS TESTES

É importante definir o método para a realização dos testes. A plataforma disponibiliza recursos para diversos testes com objetivos diferentes, com isso é importante definir os procedimentos para a realização desses testes.

Na Figura 51 é abordado sobre o método adotado para a realização de cada teste. Nesse contexto, há etapas que devem ocorrer antes, durante e após a realização do teste. Com isso, os testes respeitam a execução com quatro etapas.

Figura 51 – Passos para a realização dos testes.



Fonte: O autor (2023).

As Etapas 1 e 2, correspondem à inserção de informações sobre como será o teste. Primeiro, informando as quantidades de robôs disponíveis de cada tipo ou a conexão com a frota física, seguindo para a habilitação das tarefas que serão submetidas. Também determinando os parâmetros e estratégias para o teste. Além de realizar o requerimento das tarefas e seus respectivos locais de execução, inserindo em ordem conforme a sequência de tarefas prevista.

Já na Etapa 3, é realizada durante a simulação, estudando as informações da frota como os tipos disponíveis e a tarefa que está sendo analisada. Essa observação garante que o sistema está funcionando conforme o comportamento esperado e respeitando a ordem de prioridade das tarefas.

Por fim, a Etapa 4 apresentada por último na Figura 51 informa os dados importantes de todas as tarefas submetidas ao sistema. Nessa etapa, é salvo o histórico com todas as informações das respostas do coordenador para a sequência submetida.

5.2 PRIMEIRO ESTUDO DE CASO

Para elaborar os cenários de testes para submeter a arquitetura de coordenação, é primeiramente definido o objetivo do estudo de caso. O primeiro estudo de caso pretende a visualização do funcionamento do sistema na totalidade, as decisões e possibilidades dos três módulos.

5.2.1 Cenário Base I

Para atender o objetivo do estudo de caso, é elaborado um cenário base para a arquitetura com três tarefas, duas referentes à prioridade entre os tipos de robôs e uma colaborativa. Para a execução dessas tarefas é atribuída uma frota de robôs com quatro tipos de robôs diferentes: A, B, C e D. Para uma validação não foi necessário entrar em detalhes sobre as diferenças entre os tipos.

Tabela 5 – Execução e prioridades das tarefas.

Tarefa	Representação
T1	$AMR_A + AMR_B$
T2	$AMR_A + AMR_B + AMR_C$
T3	$AMR_D \cdot AMR_A$

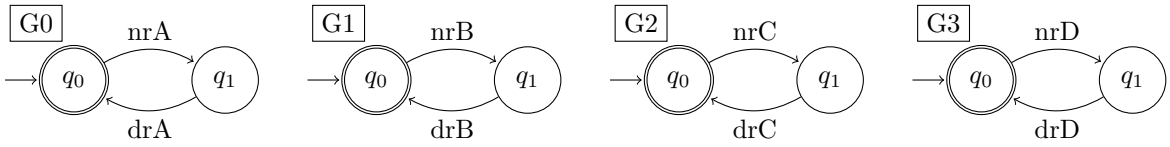
Fonte: O autor (2023).

Na Tabela 5 são apresentadas as ordens de execuções das três tarefas. Tarefa T1 pode ser executada pelos robôs A ou B, tendo o A prioridade sobre os robôs do tipo B ($A > B$). Já T2 deve ser realizada com a ordem de prioridade $A > B > C$. E por fim, T3 deve ser realizada em conjunto com um robô do tipo A e um de D. Para os testes em simulação e experimentais, foram adotados: dois robôs do tipo A, um robô de cada um dos tipos B, C e D.

5.2.2 Soluções módulo Escolha do Tipo de Robô no Cenário I

Para as soluções para as tarefas com o uso da Teoria de Controle Supervisório, foram usados como base os modelos genéricos para tarefas simples com prioridade e tarefas colaborativas, apresentados no capítulo 4.

Figura 52 – Modelos de disponibilidade dos tipos de robôs.



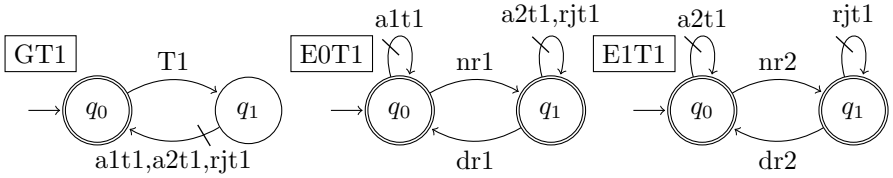
Fonte: O autor (2023).

De modo geral, seguindo os modelos genéricos é possível estabelecer quatro modelos de plantas específicos para a disponibilidade dos quatro tipos de robôs (*A*, *B*, *C* e *D*) apresentados na Figura 52. Desse modo, para as soluções dos supervisores modulares, serão usados apenas os modelos relacionados aquela tarefa específica.

5.2.2.1 Tarefa Simples com Dois Tipos (T1)

A tarefa (T1) com dois tipos de robôs com prioridade de execução. Pelos modelos genéricos apresentados pela dissertação, são propostos os modelos e especificações de controle para respeitar a prioridade 1 (tipo A — *G*0) sobre o com prioridade 2 (tipo B — *G*1). Na Figura 53, apresentam-se os modelos de planta e as especificações de controle determinadas pelos modelos genéricos e ainda com os eventos relacionados a prioridade dos tipos.

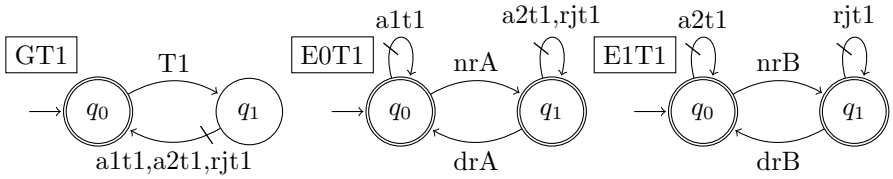
Figura 53 – Modelo (GT1) e especificações (E0T1 e E1T1) para a tarefa simples de dois tipos de robôs, com a ordem de prioridade.



Fonte: O autor (2023).

Na Figura 54, observa-se que os eventos de disponibilidade nas especificações de controle são renomeadas para os eventos de disponibilidade referentes aos tipos A e B.

Figura 54 – Modelo (GT1) e especificações (E0T1 e E1T1) para a tarefa simples de dois tipos de robôs.



Fonte: O autor (2023).

Com os modelos presentes na Figura 54, no modelo de planta $GT1$ são definidas as opções ao receber a tarefa T1, podendo ser escolhido tipo A (a1t1), escolhido tipo B (a2t1) ou ser rejeitada (rjt1). As especificações $E0T1$ e $E1T1$ estabelecem a resposta do sistema a tarefa considerando a disponibilidade de robôs A e B.

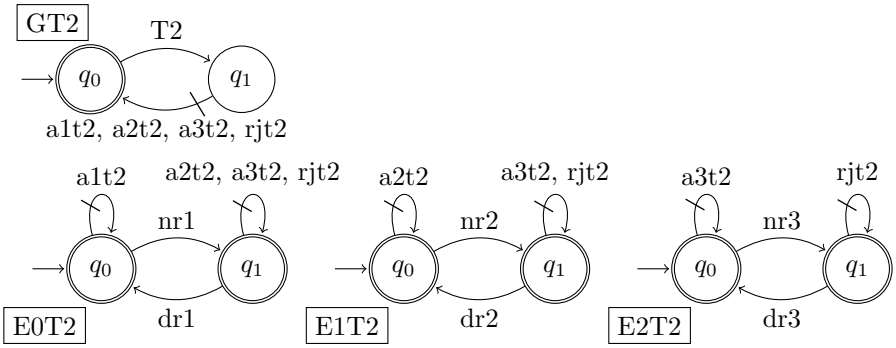
5.2.2.2 Tarefa Simples com Três Tipos (T2)

A tarefa (T2) com três tipos de robôs com prioridades de execução ($P1 > P2 > P3$). De modo geral, o sistema pode dar as opções: escolher robôs com prioridade 1 (tipo A — $G0$), escolher com prioridade 2 (tipo B — $G1$), escolher com prioridade 3 (tipo C — $G2$) ou ainda pode ser rejeitada. Considerando:

- a1t2 é escolhido robôs do tipo A
- a2t2 é escolhido robôs do tipo B
- a3t2 é escolhido robôs do tipo C

Ao usar os modelos genéricos para essa tarefa, obtêm-se o modelo e especificações de controle da Figura 55. Contudo, ainda com os eventos de disponibilidade pela prioridade.

Figura 55 – Modelo (GT2) e especificações (E0T2, E1T2, e E2T2) para a tarefa simples de três tipos de robôs, com a ordem de prioridades.

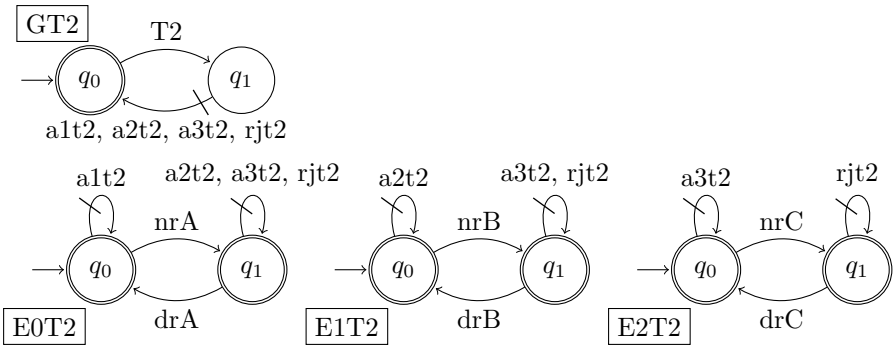


Fonte: O autor (2023).

Com esses modelos, são renomeados os eventos para a disponibilidade dos tipos de robôs para a tarefa, como demonstra a Figura 56.

Para o modelo de planta $GT2$, é apresentado da mesma forma que o modelo $GT1$ para T1, representando todo o comportamento do sistema ao receber a tarefa T2. Dessa vez, são estabelecidos três especificações de controle ($E0_{T2}$, $E1_{T2}$ e $E2_{T2}$), que garantem a ordem de prioridade, assim pelos eventos controláveis: ac1 > ac2 > ac3 > rjt2.

Figura 56 – Modelo (GT2) e especificações (E0T2, E1T2, e E2T2) para a tarefa simples de três tipos de robôs.

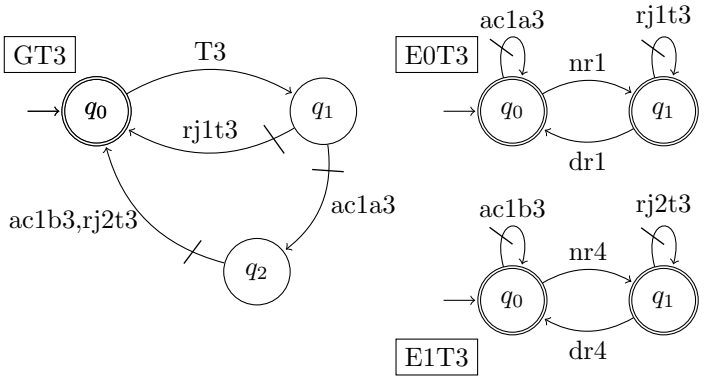


Fonte: O autor (2023).

5.2.2.3 Tarefa Colaborativa com Dois Tipos (T3)

Por fim, a tarefa colaborativa (T3) com dois tipos de robôs diferentes. As opções do sistema para essa tarefa: escolher os dois tipos de robôs (tipo A — $G0$ sendo o grupo α e D — $G3$ o grupo β), rejeitar a tarefa por falta de robôs do primeiro tipo ou rejeitar por falta do segundo tipo. Usando os modelos genéricos, obtêm-se os modelos da Figura 57.

Figura 57 – Modelo (GT3) e especificações (E0T3 e E1T3) para a tarefa colaborativa de dois tipos de robôs, com a ordem de prioridade.



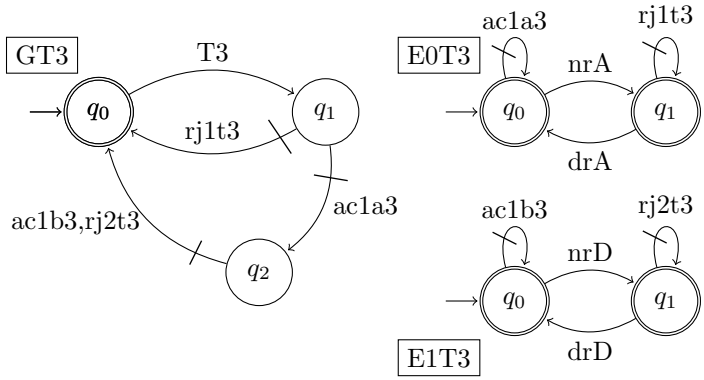
Fonte: O autor (2023).

Através dos modelos da Figura 57, são identificados os tipos de robôs relacionados a cada prioridade, para então renomeá-los e associar aos tipos de robôs (tipo A e tibo D), assim apresentado na Figura 58.

Com os eventos controláveis representando:

- ac1a3 escolhe robôs do tipo A de α
- ac1b3 escolhe robôs do tipo D de β
- rj1t3 rejeita tarefa por falta de robôs do grupo α

Figura 58 – Modelo (GT3) e especificações (E0T3 e E1T3) para a tarefa colaborativa de dois tipos de robôs.



Fonte: O autor (2023).

- rj2t3 rejeita tarefa por falta de robôs do grupo β

O autômato GT3 descreve todos os comportamentos das tarefas, ao receber a tarefa (T3), nesse caso, deve ter a disponibilidade de todos os tipos de robôs para a tarefa ser executada. As especificações de controle E0T3 e E1T3 tratam de determinar qual a resposta do sistema para a tarefa. Com essas especificações, é possível determinar qual o motivo para a tarefa ser rejeitada.

5.2.2.4 Supervisor monolítico para Cenário I

Determinando os modelos de planta ($G's$) e as especificações de controle ($E's$), para a abordagem monolítica é necessário determinar o modelo de planta global (G) e a especificação de controle que represente as demais (E). Para obtenção desses modelos é necessário o uso da operação de composição síncrona.

$$G = G0 \| G1 \| G2 \| G3 \| GT1 \| GT2 \| GT3$$
$$E = E0T1 \| E1T1 \| E0T2 \| E1T2 \| E2T2 \| E0T3 \| E1T3$$

Assim foi obtido o modelo de planta G com o comportamento total do sistema, com todas as alternativas. Já a especificação E que descreve todas as restrições e políticas para o sistema. Agora é necessário obter a linguagem K, considerando o desenvolvimento e modelos em autômatos, as operações também são realizados em autômatos, logo é estabelecido o autômato R, que logo $K = L(R)$.

$$R = G \| E$$

Tendo o R obtido por meio da composição síncrona de G e E. Para o próximo passo é necessário o uso da ferramenta *supC*, considerando os modelos R e G.

$S = supC(G,R)$ **(192 estados, 1344 transições)**.

O supervisor monolítico S é estabelecido com 192 estados com 1344 transições divididas em todos os eventos do sistema, considerando todos os tipos de robôs e as tarefas. Para os cálculos do supervisor foi utilizado a ferramenta computacional Nadzoru (PINHEIRO et al., 2015).

5.2.2.5 Supervisores modulares locais para Cenário I

Já considerando a abordagem modular local, a estratégia escolhida por essa dissertação foi estabelecer um supervisor modular local para cada tarefa. Com esse critério, primeiro é importante determinar as especificações de controle relacionadas a cada tarefa e realizando a composição síncrona em todas as especificações relacionadas.

$E_{Local_1} = E0T1 \| E1T1$ **(4 estados)**;
$E_{Local_2} = E0T2 \| E1T2 \| E2T2$ **(8 estados)**;
$E_{Local_3} = E0T3 \| E1T3$ **(4 estados)**.

Definindo isso, é possível determinar os modelos de plantas locais. Para serem obtidas, são analisados todos os modelos de planta afetados por cada especificação de controle. Para então, realizar a composição dos modelos de disponibilidade dos robôs e o modelo da tarefa. Com E_{Local_1} afetando os modelos $G0, G1$ e $GT1$, E_{Local_2} os modelos $G0, G1, G2$ e $GT2$ e a especificação E_{Local_3} afetando $G0, G2$ e $GT3$.

$G_{Local_1} = G0 \| G1 \| GT1$ **(8 estados)**;
$G_{Local_2} = G0 \| G1 \| G2 \| GT2$ **(16 estados)**;
$G_{Local_3} = G0 \| G2 \| GT3$ **(12 estados)**.

Mais uma vez é necessário o uso da composição para estabelecer todos os autômatos $R's$ do sistema. Lembrando que terá um R para cada planta localizada. Posteriormente é calculado cada um dos supervisores através da ferramenta *supC*, atribuindo cada G_{local} com o seu autômato R_{local}.

$R_i = G_{Local_i} \| E_{Local_i}$ *i = 1, 2 e 3*;
$S_i = SupC(G_{Local_i}, R_i)$ *i = 1, 2 e 3*.

Após os cálculos dos supervisores, é possível estabelecer as informações de estados e transições do sistema.

- S_1: 8 estados, 24 transições;
- S_2: 16 estados, 64 transições;
- S_3: 12 estados, 36 transições.

Com os supervisores modulares locais, é necessário realizar o teste para analisar a interferência dos três supervisores em conjunto. Para esse teste é realizada a composição síncrona de todos os supervisores modulares ($S_{123} = S_1 \| S_2 \| S_3$). Ao realizar essa operação, o autômato S_{123} resultante possui 192 estados e 1344 transições. Com esse resultado, é observado que S_{123} não possui estados bloqueantes, assim é dada a garantia que os supervisores modulares locais funcionam sem conflito entre eles.

5.2.2.6 Discussão sobre as abordagens

Analisando as duas abordagens já tratadas, para a implementação do supervisor monolítico é necessária a implementação de um autômato de 192 estados e 1344 transições. Enquanto para a modular local, a implementação é realizada com três supervisores, no pior dos casos o maior deles é de 16 estados com 64 transições.

Para a estratégia de implementação de supervisores por meio da elaboração de dicionários, para a implementação modular local é necessária a criação de três dicionários. Analisando apenas os eventos presentes no estado atual do sistema, no pior caso entre todos os dicionários elaborados é analisado quatro eventos em um estado. Logo, a vantagem da abordagem modular local é dada, então, adotada na implementação e testes para a validação da solução do módulo de Escolha do Tipo de Robô.

Com a abordagem modular local ainda há outra alternativa que poderia ser adotada, utilizando a redução de autômatos para estabelecer modelos menores. Com os supervisores reduzidos, é possível adotar uma implementação que ocupe menos memória para tratar esses supervisores. Para os cenários da dissertação, não foi adotada essa alternativa, estabelecendo supervisores maiores (com o critério de determinar um supervisor para cada tipo de tarefa).

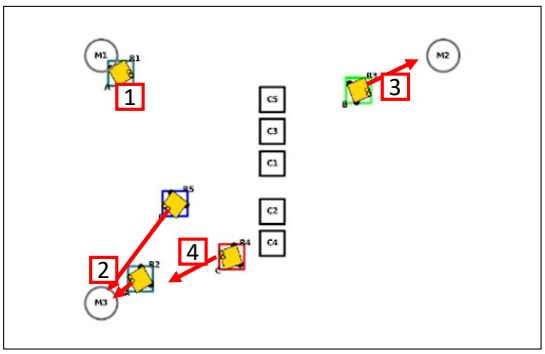
5.2.3 Comportamento do coordenador no Cenário I com o simulador

Para analisar o comportamento do coordenador é importante observar para as respostas para as tarefas, foram propostas duas sequências: uma para a visualização do funcionamento do sistema e outro a prioridade de escolha entre os tipos de robôs. Para visualizar o sistema funcionando foram submetidas as tarefas: T1, T3, T1 e T2, observando as tarefas com prioridade e uma colaborativa. Enquanto para a sequência para analisar a prioridade são submetidas ao sistema: T2, T2, T2 e T2.

Com o teste utilizando o Robotarium é possível realizar uma primeira análise preliminar. Observando o comportamento do sistema, com informações para definir se a arquitetura

de fato possibilita a flexibilidade e garante o funcionamento do sistema com as políticas de gerenciamento propostas.

Figura 59 – Comportamento no Cenário I no simulador com a primeira sequência, indicando a movimentação dos robôs e a ordem de requerimento das tarefas.



N°	Tarefa	Robôs
1	T1	R1
2	T3	R2+R5
3	T1	R3
4	T2	R4

Fonte: O autor (2023).

- n^o **1:** Tarefa T1, robôs do tipo A e B disponíveis. Escolhido o robô R1 do tipo A;
- n^o **2:** Tarefa T3, robôs do tipo A e D disponíveis. Escolhido o robô R2 do tipo A e R5 do B;
- n^o **3:** Tarefa T1, apenas robôs do tipo B disponíveis. Escolhido o robô R3 do tipo B;
- n^o **4:** Tarefa T2, apenas robôs do tipo C disponíveis. Escolhido o robô R4 do tipo C.

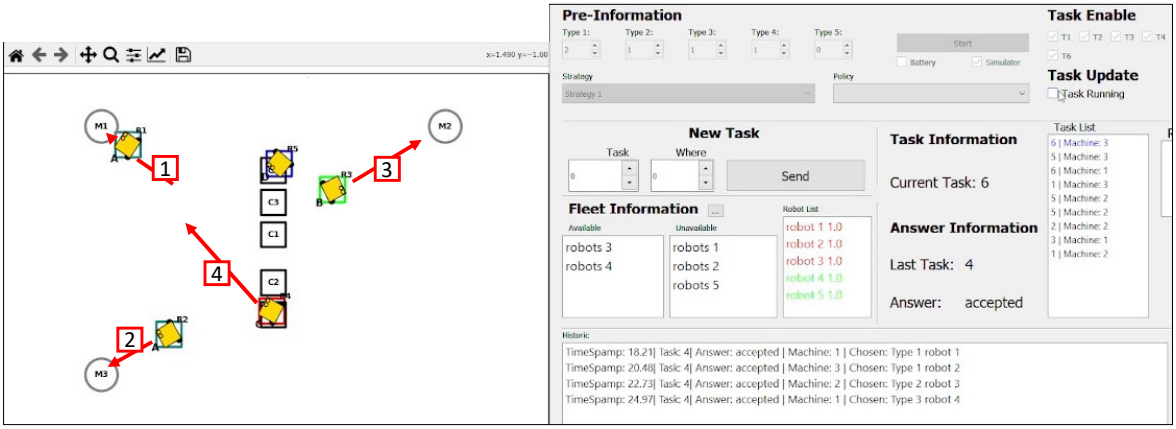
Na Figura 59 é apresentado o comportamento do sistema para a primeira sequência no cenário I. Nesse teste é possível observar a política de prioridade em T1, de modo que para o robô R3, que pertence ao tipo B, ser escolhido é necessário não ter mais disponíveis robôs A ($R1$ e $R2$). Também foi observado o comportamento para a tarefa colaborativa, escolhidos os robôs $R2$ e $R5$ onde ambos devem ir para $M3$ executar a tarefa.

Já ao submeter ao sistema uma sequência de tarefas T2, é possível analisar a ordem de prioridade de escolha entre os três tipos de robôs. Essa sequência é observada na Figura 60, onde é visível que os robôs do tipo A são os primeiros a serem escolhidos para a realização das tarefas e o do tipo B é escolhido somente quando não há do tipo A disponível. Enquanto para o robô do tipo C é escolhido apenas quando não há mais robôs dos tipos A e B.

5.2.4 Comportamento do coordenador no Cenário I com robôs da LEGO

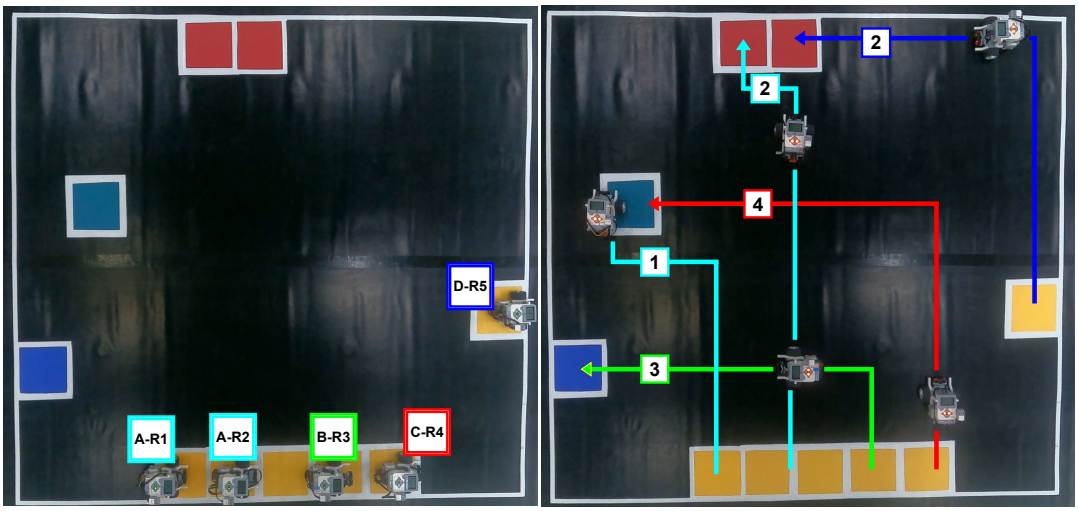
Além dos testes em ambiente simulado, as tarefas desse cenário foram submetidas em ambiente físicos com os robôs da LEGO. Também com a mesma sequência de tarefas já apresentada, para observar o funcionamento geral do sistema e suas prioridades fora de um ambiente simulado por computador.

Figura 60 – Comportamento no Cenário I no simulador com a segunda sequência, indicando a movimentação dos robôs e as informações na interface.



Fonte: O autor (2023).

Figura 61 – Comportamento no Cenário I com robôs da LEGO com a primeira sequência, indicando o momento inicial da frota e a movimentação dos robôs.

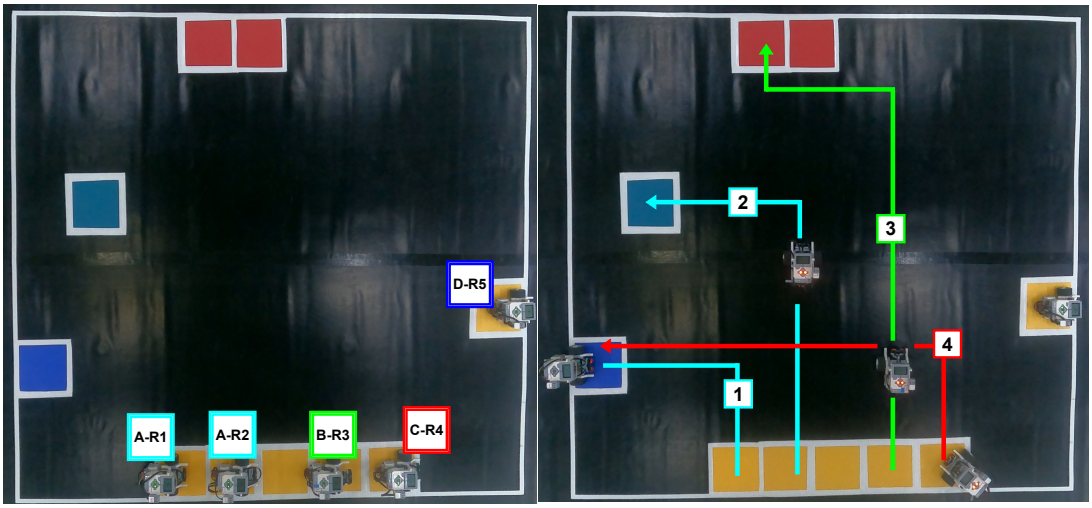


Fonte: O autor (2023).

Na Figura 61 é apresentado o momento inicial (na esquerda) com os robôs em repouso e outro momento que os robôs (na direita) já estão realizando o percurso para executar as quatro tarefas da primeira sequência. Os percursos estão destacados pelo mesmo padrão de cores estabelecido para os tipos de robôs. Nesse ínterim, mais uma vez as políticas de escolha foram respeitadas, contribuindo com as discussões já apresentadas com a sequência no ambiente simulado.

Já na Figura 62 é apresentada a sequência de tarefas T2, sendo observado o comportamento de escolha do tipo de robô com a prioridade. Para os testes com os robôs da LEGO foi possível executar apenas quatro tarefas com um robô até os erros nas posições ficarem significativos. Todos os testes realizados para esse primeiro cenário é apresentado em vídeo, disponível em: vídeo cenário 1.

Figura 62 – Comportamento no Cenário I com robôs da LEGO com a segunda sequência, indicando o momento inicial da frota e a movimentação dos robôs.



Fonte: O autor (2023).

5.3 SEGUNDO ESTUDO DE CASO

O segundo estudo de caso propõe apresentar o funcionamento do sistema com um número maior de robôs e com a adição de uma nova tarefa colaborativa. Com as mesmas tarefas: T1, T2 e T3, apresentadas no primeiro estudo de caso e uma nova T4. Possui o objetivo de apresentar o funcionamento da arquitetura e a flexibilidade da plataforma quanto ao número de robôs e maior diversidade de tarefas.

5.3.1 Cenário Base II

Para o cenário do segundo estudo de caso, foram tratados quatro tarefas distintas, com tarefas com prioridade, colaborativa e colaborativa com prioridade. Agora, a frota composta de 3 robôs do tipo A, 2 robôs do B e 1 robô para C e D, totalizando 7 AMRs no cenário. Com esse número de robôs foi optado pela realização dos testes apenas em ambiente simulado.

Tabela 6 – Execução e prioridades das tarefas no Cenário II.

Tarefa	Representação
T1	$AMR_A + AMR_B$
T2	$AMR_A + AMR_B + AMR_C$
T3	$AMR_D \cdot AMR_A$
T4	$AMR_C \cdot AMR_A + AMR_C \cdot AMR_D$

Fonte: O autor (2023).

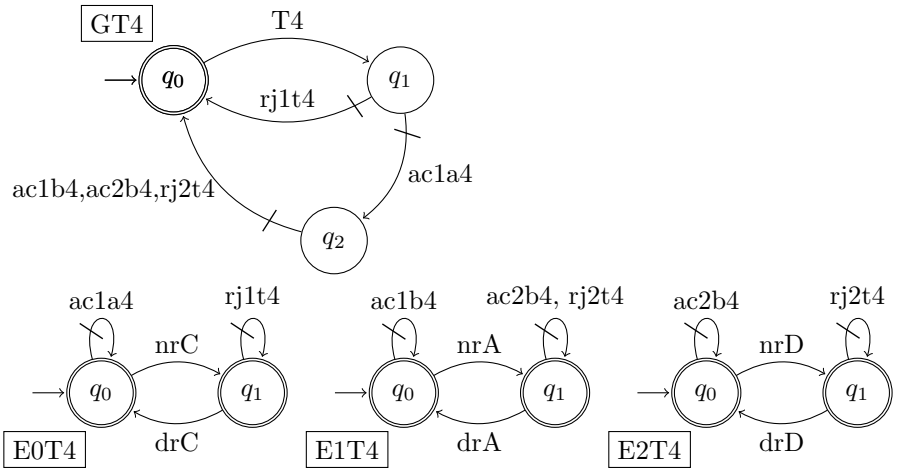
Para esse cenário é proposta a adição de uma nova tarefa colaborativa, com prioridade em um dos grupos. Na Tabela 6 são apresentadas as quatro tarefas submetidas ao sistema. T4 deve ser realizada de maneira colaborativa de robôs do tipo C com tipo A ou ainda um robô do tipo C

e outro do tipo D. Nesse contexto, deve ter disponibilidade de um robô do tipo C, enquanto há a prioridade de escolha do tipo A para a execução da tarefa.

5.3.2 Modelos e supervisores para o Cenário II

Os modelos apresentados no cenário I sobre as tarefas T1, T2 e T3, ainda são considerados para esse cenário II, nas Figuras 54, 56 e 58. Essa nova tarefa é definida como colaborativa, com o grupo α sendo composto pelos robôs do tipo C, enquanto os robôs dos tipos A e D compõem o grupo β, com prioridade para o tipo A. Assim, os modelos ($GT4$) e especificações de controle ($E0T4$, $E1T4$ e $E2T4$):

Figura 63 – Modelo (GT4) e especificações (E0T4, E1T4 e E2T4) para a tarefa T4.



Fonte: O autor (2023).

Na Figura 63 são apresentados o modelo de planta referente à tarefa T4 e as especificações de controle, formados através dos modelos genéricos de tarefas colaborativas com prioridade, com os eventos renomeados para os tipos de robôs. Com $GT4$ representando os possíveis comportamentos do sistema ao receber uma tarefa T4. A especificação de controle $E0T4$ referente aos robôs do tipo C e as especificações $E1T4$ e $E2T4$ sobre a escolha entre a prioridade entre os robôs do tipo A e D. Realizando o recálculo para o supervisor monolítico é estabelecido:

$$G = G0 \| G1 \| G2 \| G3 \| GT1 \| GT2 \| GT3 \| GT4$$
$$E = E0T1 \| E1T1 \| E0T2 \| E1T2 \| E2T2 \| E0T3 \| E1T3 \| E0T4 \| E1T4 \| E2T4$$
$$R = G \| E$$
$$S = supC(G,R) \textbf{ (576 estados, 4608 transições)}$$

Esse cálculo para o supervisor monolítico foi utilizado os modelos e especificações de controle de todas as quatro tarefas. Já com para a abordagem modular local são obtidos os supervisores:

- S_1: 8 estados, 24 transições;
- S_2: 16 estados, 64 transições;
- S_3: 12 estados, 36 transições;
- S_4: 24 estados, 96 transições;

Agora o sistema pode embarcar os quatro supervisores, cada um relacionado a uma única tarefa. Realizando a composição síncrona de todos os supervisores modulares locais ($S_{1234} = S_1 \| S_2 \| S_3 \| S_4$), com a análise é identificado que o autômato obtido é não bloqueante. Assim, com o novo supervisor S_4, ainda é garantido o funcionamento do sistema sem conflito entre os supervisores.

Pela maneira que foram elaborados os modelos e especificações de controle, de maneira geral os supervisores modulares locais compartilham apenas eventos não controláveis e determinando que o sistema recebendo uma tarefa por vez, é estabelecido que independente do número de tarefas submetidas ao sistema, não terá conflito entre os supervisores modulares. Com essa característica, a arquitetura ganha ainda mais flexibilidade para determinar as tarefas que serão estabelecidas no sistema.

Com a adição da nova tarefa, ainda há vantagens para a implementação de supervisores, já que agora o monolítico possui no total 576 estados com 4608 transições, enquanto para o modular local o pior caso é um supervisor de 24 estados com 96 transições. O método para a implementação de supervisores continua o mesmo apresentado na dissertação, através da elaboração de dicionários.

5.3.3 Sequência de tarefas para o Cenário II

As sequências de tarefas submetidas nesse cenário deve ter a finalidade de demonstra o funcionamento dos módulos de Tratamento de Tarefas e a Escolha do Tipo de Robô. Com esse intuito, os testes foram realizados sem a simulação da bateria, para facilitar a análise das respostas do sistema.

Primeira sequência de tarefas: submetido T3 (M1), T3 (M3), T3 (M1) e T1 (M1). Para visualizar as respostas do coordenador e analisar o funcionamento do módulo de Tratamento de Tarefa. Com essa sequência é previsto que tenha pelo menos um aceite e um rejeito de tarefa.

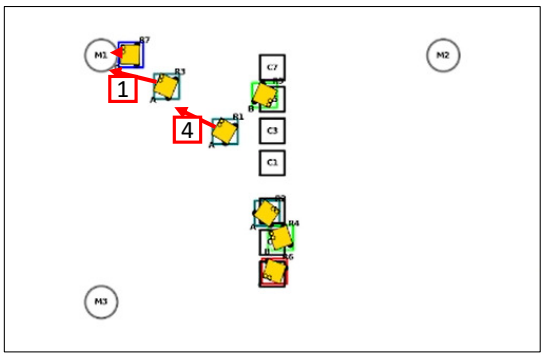
Segunda sequência de tarefas: foi submetida a sequência de T1 (M3), T4 (M2), T3 (M1), T1 (M2), T2 (M1), T1 (M3) e T2 (M2). Essa proposta tenta apresentar as ordens de prioridades estabelecidas para T1, T2 e T4.

Terceira sequência de tarefas: é proposto analisar as primeiras 81 respostas do sistema, considerando uma lista de tarefas gerada aleatoriamente entre as quatro tarefas já propostas, também aleatoriamente o destino dessas tarefas. Com o intuito de observar os tipos de robôs são os mais requisitados e quantas respostas de rejeito há com essa sequência.

5.3.4 Comportamento do coordenador no Cenário II no simulador

Considerando as sequências de tarefas apresentadas e o cenário, é aplicado ao sistema simulado através do Robotarium e a plataforma a arquitetura de coordenação. As respostas e o comportamento do sistema para a **primeira sequência** de tarefa submetida:

Figura 64 – Comportamento no Cenário II no simulador com a primeira sequência, indicando a movimentação dos robôs e a ordem das respostas para as tarefas.



N°	Tarefa	Robôs
1	T3	R3+R7
2	T3	Rej
3	T3	Não Analisada
4	T1	R1

Fonte: O autor (2023).

Tabela 7 – Respostas do coordenador para o cenário II com a primeira sequência.

N^o	Tarefa	Tipo AMR	Robô	Destino
1	$T3$	A e D	$R3+R7$	$M1$
2	$T3$	NA^*	$Rejeitada$	NA^*
3	$T2$	A	$R1$	$M1$
4	$T3$	NA^*	$Rejeitada$	NA^*

* Não se Aplica

Fonte: O autor (2023).

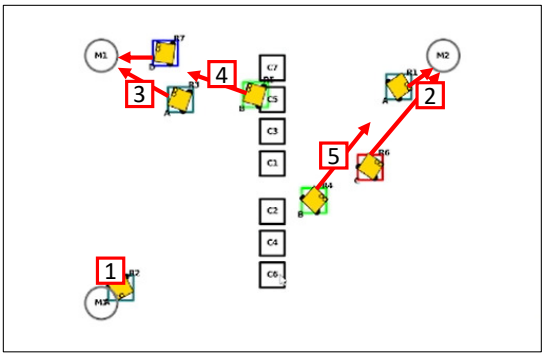
A Figura 64 e a Tabela 7 tratam, em conjunto, da apresentação do comportamento e respostas dadas pelo coordenador no cenário II com a primeira sequência. Com esses dados, é possível observar o funcionamento do sistema para uma rejeição de tarefa e, principalmente, o funcionamento do módulo de Tratamento de Tarefas.

Nas respostas dadas para a primeira sequência de tarefas, observa-se que uma das tarefas não foi analisada, pois a tarefa de número 3 é do mesmo tipo de tarefa que já foi rejeitada. Com o comportamento do coordenador, são apresentadas três respostas do sistema, dois com aceite e outra com rejeito. É observado que ao aceitar T2 de número 4, o coordenador volta a analisar a tarefa T3 de número 2, rejeitada pela segunda vez por falta de robôs disponíveis. E ao submeter a **segunda sequência de tarefas**:

Respostas do coordenador com a sequência de tarefas demonstrados pela interface:

Com essa segunda sequência, é possível observar o comportamento do módulo de Escolha do Tipo de Robô. Em vários momentos da sequência é analisado os tipos de robôs disponíveis.

Figura 65 – Comportamento no Cenário II no simulador com a segunda sequência, indicando a movimentação dos robôs e a ordem das respostas para as tarefas.



N°	Tarefa	Robôs
1	T1	R2
2	T4	R1+R6
3	T3	R3+R7
4	T2	R5
5	T2	R4

Fonte: O autor (2023).

Tabela 8 – Respostas do coordenador para o cenário II com a segunda sequência.

N^o	Tarefa	Tipo AMR	Robô	Destino
1	$T1$	A	$R2$	$M3$
2	$T4$	A e C	$R1 + R6$	$M2$
3	$T3$	A e D	$R3 + R7$	$M1$
4	$T1$	B	$R5$	$M1$
5	$T2$	B	$R4$	$M2$

* Não se Aplica

Fonte: O autor (2023).

- Número 1: tarefa sendo analisada é T1, há robôs do tipo A e B disponíveis. Escolhendo o tipo A pela sua prioridade.
- Número 2: tarefa sendo analisada é T4, há robôs do tipo A, C e D disponíveis. Escolhendo o tipo C e do tipo A.
- Número 3: tarefa analisada T3, sendo colaborativa há robôs do tipo A e D disponíveis.
- Número 4: tarefa analisada T2, há apenas robôs do tipo B disponíveis para essa tarefa.
- Número 5: tarefa analisada T2, há apenas robôs do tipo B disponíveis para essa tarefa.

Com o funcionamento do módulo de Escolha do Tipo de Robô, destacados os números 4 e 5 analisam o funcionamento com a prioridade entre os tipos para a tarefa T2. Já sobre a última sequência, onde são analisadas as primeiras 81 respostas do sistema, levou o total de $304,29$ segundos. Do total de respostas, 61 tarefas foram aceitas, enquanto obteve 20 rejeitos.

Na Tabela 9, todas as informações sobre as 81 respostas do sistema são apresentadas, permitindo analisar individualmente as respostas para cada tipo de tarefa. Como o esperado, o tipo de robô mais requisitado para a realização da tarefa foi o tipo A, em contraponto ao tipo C, que possui a menor prioridade para executar as tarefas, sendo escolhido apenas dez vezes nesse total de tarefas.

Tabela 9 – Respostas do coordenador no cenário II com o simulador para a terceira sequência.

Tipo/Rejeito	T1	T2	T3	T4	Total
Tipo A	16	13	10	6	45
Tipo B	6	6	*X*	*X*	12
Tipo C	*X*	0	*X*	10	10
Tipo D	*X*	*X*	10	4	14
Rejeita	0	0	*X*	*X*	0
Rejeita1	*X*	*X*	17	0	17
Rejeita2	*X*	*X*	3	0	3

X Não se Aplica

Fonte: O autor (2023).

Entre as respostas de rejeito, naturalmente estariam presentes uma resposta de rejeito para cada tarefa de prioridade também (T1 e T2), contudo na Tabela 9 a resposta "Rejeita" é dada para ambas as tarefas com prioridade e as respostas "Rejeita 1" e "Rejeita 2" são dadas para as tarefas colaborativas (T3 e T4). Naturalmente as tarefas que precisam de dois robôs para ser executada terá maior rejeito, enquanto para a tarefa que mais de um tipo robô pode ser escolhido para executar será mais difícil de ser rejeitada. Disponível em: vídeo cenário 2.

5.4 TERCEIRO ESTUDO DE CASO

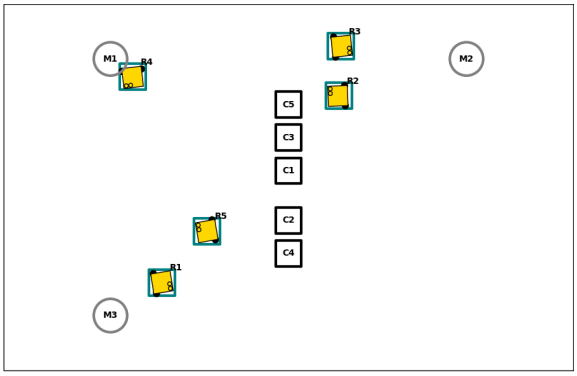
Visando testar e apresentar os dois algoritmos de escolha, é proposto mais um estudo de caso considerando apenas um tipo de robô executando apenas um tipo de tarefa. Tendo em vista a análise do funcionamento do módulo Escolha do Robô que considera as informações individuais dos robôs.

5.4.1 Cenário Base III

O cenário previsto para esse estudo de caso é baseado em apenas um tipo de robô *A*. Para analisar o funcionamento dos algoritmos de escolha, foi optado pela execução apenas no ambiente simulado com Robotarium. Com o uso do simulador, é possível analisar o resultado com um número maior que robôs, além de submeter uma sequência de tarefas maior ao sistema. Os testes foram realizados com cinco robôs simulados, todos do mesmo tipo. Para cada teste foi submetida a mesma sequência de tarefas e seus destinos.

Na Figura 66 é apresentado o cenário base 3, com os cinco robôs em funcionamento, todos do mesmo tipo. Além de testes com todas as informações dos robôs, é interessante a realização de um teste desconsiderando o nível de bateria. Nesse teste é considerado que os robôs possuem bateria infinita, logo não é um critério a ser considerado na escolha do robô. Podendo ser analisado os dados e informações dos dois algoritmos apenas considerando a distância dos robôs e analisando a eficiência e o comportamento do sistema partindo dessa característica.

Figura 66 – Cenário Base 3, com cinco robôs do mesmo tipo.



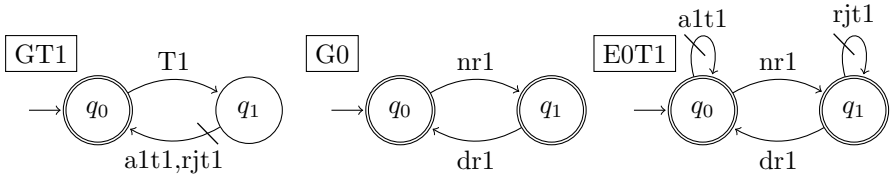
Fonte: O autor (2023).

5.4.2 Tarefa proposta e supervisor

Com o intuito de analisar o funcionamento do módulo de escolha de robô, é proposto uma tarefa simples. Por ter disponível apenas um tipo de robô, a tarefa deve ser executada apenas quando ter a disponibilidade desse tipo.

Utilizando o método estabelecido com os modelos genéricos, na Figura 67 apresentam-se os modelos com os eventos sobre prioridade.

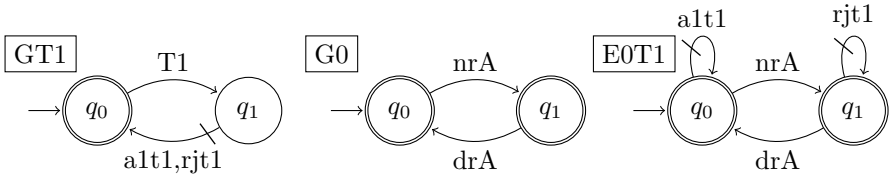
Figura 67 – Modelos para a tarefa simples do Cenário III obtido pelos modelos genéricos.



Fonte: O autor (2023).

Na Figura 68, apresentam-se os modelos já renomeados com os eventos de disponibilidade do tipo de robô (Tipo A).

Figura 68 – Modelos para a tarefa simples do Cenário III.



Fonte: O autor (2023).

O supervisor resultante é obtido, respeitando o método apresentado anteriormente, com os modelos de planta e a especificação de controle da Figura 68. Obtendo G através de $G = GT1 \| G0$ e a especificação de controle já é dada como $E = E0T1$. Assim é possível obter o autômato

$R = G||E$, para que finalmente, obter o supervisor $S = supC(G,R)$. A implementação é dada de apenas um supervisor com 4 estados e 8 transições.

5.4.3 Sequência de tarefas para Cenário III

Um ponto importante para esse teste é a sequência de tarefas que será submetida ao sistema. A sequência de tarefas e seus destinos devem ser o mesmo para todos os testes. Nos testes serão analisados o levantamento do número de tarefas aceitas em um período. O período deve contemplar: o momento onde todos os robôs são escolhidos e o momento que a maioria dos robôs esteja com nível baixo de bateria.

Como o sistema é considerado com apenas uma tarefa, a sequência se repetirá com a mesma tarefa, com o número total de 70 respostas dadas pelo sistema. Já sobre o destino, foi escolhido de modo aleatório visando uma boa distribuição nos destinos e considerando os destinos de Máquina 1 (M1), Máquina 2 (M2) e Máquina 3 (M3). Considerando todas as tarefas, na sequência há 26 tarefas para a M1, 23 para M2 e 21 para M3.

5.4.4 Comportamento do coordenador no Cenário III no simulador

Aplicando a arquitetura de coordenação no cenário, é possível analisar cada um dos algoritmos e determinando as suas vantagens e desvantagens para o cenário proposto. Destacando que além dos dois algoritmos de escolha, os testes foram realizados variando os parâmetros de modo que dê preferência a uma característica (distância ou nível de bateria). É importante destacar que o coordenador analisará uma tarefa apenas se existir algum tipo de robô disponível naquele momento, logo não haverá respostas de rejeito nesse cenário.

Tabela 10 – Tempos de execução de tarefas no cenário III considerando a bateria.

Algoritmo	Tempo 20 Tarefas	Tempo 40 Tarefas	Tempo 60 Tarefas
Guloso (Equilibrado)	$69,35s$	$187,75s$	$489,85s$
Guloso (Distância)	$64,79s$	$190,74s$	$543,56s$
Guloso (Bateria)	$69,31s$	$196,76s$	$483,74s$
Probabilidade (Equilibrado)	$63,28s$	$144,20s$	$534,75s$
Probabilidade (Distância)	$63,29s$	$148,71s$	$593,53s$
Probabilidade (Bateria)	$63,32s$	$162,24s$	$512,40s$

Fonte: O autor (2023).

Na Tabela 10 são apresentados os tempos em que o sistema atinge determinado número de tarefas aceitas. É possível analisar em três pontos, quando o sistema atinge 20, 40 e 60 tarefas aceitas e encaminhadas aos robôs. Com o consumo de bateria e a constância de requerimentos de tarefas, esses pontos são interessantes, por ser possível analisar o início do sistema em 20 tarefas, quando os robôs já estão com níveis baixos de bateria (40 tarefas). Por fim, quando o sistema já está em funcionamento em um tempo considerável (60 tarefas).

Tabela 11 – Tempos de execução de tarefas no cenário III desconsiderando a bateria.

Algoritmo	Tempo 20 Tarefas	Tempo 40 Tarefas	Tempo 60 Tarefas
Guloso	$90,40s$	$160,31s$	$241,17s$
Com probabilidade	$93,33s$	$177,45s$	$255,53s$

Fonte: O autor (2023).

Já na Tabela 11 são apresentadas as respostas para a sequência de tarefas desconsiderando a bateria no sistema. Nesse teste, mais uma vez analisando o sistema quando atinge 20, 40 e 60 tarefas aceitas. Ao analisar o comportamento do sistema, é observado que o sistema alcançou o número de 60 tarefas aceitas muito antes que os testes anteriores, já que os robôs não tinham a necessidade de ficar na base até atingir um valor considerável de bateria.

Para esse cenário, o algoritmo guloso pode aparentar como o mais eficiente, observando o sistema a longo prazo por uma pequena margem com relação ao com probabilidade, sem alterações decisivas no comportamento para estabelecer qual o algoritmo de escolha mais adequado para essa situação. Contudo, estabelece a flexibilidade da plataforma para implementações de tipos diferentes de algoritmos de escolha.

5.5 CONCLUSÃO DO CAPÍTULO

Nesse capítulo foram apresentados os dados da proposta de coordenador em três estudos de casos, cada um com objetivo de apresentar algumas características específicas do funcionamento do método proposto. O primeiro estudo de caso abordou o funcionamento geral do sistema e com foco nas tarefas com prioridade. Também é apresentado a flexibilidade da plataforma e a possibilidade de implementação em cenários em laboratório.

O segundo estudo de caso apresentou outro cenário, com novas tarefas e sequências que apresentam situações diferentes para as políticas de gerenciamento previamente definidas, como o comportamento do coordenador ao rejeitar uma tarefa. Por fim, o terceiro estudo de caso estabelece o funcionamento entre as soluções através dos algoritmos de escolha, não com o intuito de determinar o mais eficiente, mas para apresentar a versatilidade da arquitetura.

Com foco no comportamento do sistema aceitando e rejeitando uma tarefa, observando aspectos do primeiro e do segundo estudos de casos, são destacados as seguintes conclusões sobre o funcionamento com a proposta de solução do módulo de **Tratamento de Tarefas**:

- Respeitou a ordem de requerimento submetida ao sistema; e
- Evitou deixar robôs ociosos;

Observando o comportamento do sistema com a solução para o módulo de Tratamento de Tarefas são propostas análises sobre seu desempenho e funcionamento. Com as respostas sobre as tarefas rejeitadas, é possível realizar um relatório de rejeito. Se determinado quais as

principais respostas de rejeitos de tarefas, são identificados os tipos de robôs que o sistema tem mais necessidade. Esse relatório pode contribuir para adquirir novos robôs conforme a necessidade do ambiente, para reduzir o número de rejeitos e aumentar o uso da frota no sistema futuramente.

Também é possível analisar no primeiro e segundo estudo de caso, os tipos de robôs escolhidos e para quais as tarefas que eles foram atribuídos. Com essas informações, é possível estabelecer as seguintes conclusões sobre as soluções para o módulo de **Escolha do Tipo de Robô**:

- Respeitou todas as prioridades estabelecidas;
- Estabeleceu o funcionamento de tarefas colaborativas; e
- Apresentou a flexibilidade em considerar novas tarefas.

Por último, analisando principalmente o terceiro estudo de caso, é observada a flexibilidade para algoritmos de escolha para o módulo de Escolha do Robô, com os dados da alocação de tarefas com ambos algoritmos e como as variações entre os pesos para as características de bateria e distância. Analisando em específico o módulo de **Escolha do Robô** são destacados:

- A flexibilidade em tipos diferentes de algoritmos;
- flexibilidade no uso de características dos robôs.

Com isso, as soluções propostas para cada módulo de coordenação garantiram as políticas de gerenciamento propostas. Também nesse contexto, foi possível analisar o funcionamento da plataforma desenvolvida. Lembrando que a plataforma é constituída pela interface com usuário, simulador e a comunicação com os robôs da LEGO. Sendo possível destacar as seguintes características:

- Possibilitou a flexibilidade do número de robôs nos quatro tipos estabelecidos;
- Durante os testes com as tarefas propostas, estabeleceu o funcionamento e a implementação dos supervisores;
- Apresentou flexibilidade entre diversas estratégias para o gerenciamento das frotas;
- Se apresentou como sólido nos diversos testes e experimentos, considerando tanto o ambiente simulado como real;
- Apresentou e armazenou as informações relevantes para serem analisadas durante e após os testes; e
- Estabeleceu a comunicação do coordenador com a frota de robôs.

É importante apresentar uma análise dos ambientes simulados e os elaborados com os robôs de LEGO. O ambiente simulado, mais explorado pelos testes e sequências de tarefas, apresentou resultados sólidos e flexibilidade na elaboração dos cenários. Ainda sobre o simulado, apresentou facilidade em manipular e interpretar os dados obtidos, principalmente os testes mais longos com mais de 20 tarefas sendo submetidas.

Já no cenário real, utilizando robôs LEGO EV3, foi possível realizar as duas sequências de tarefas propostas para o cenário. Contudo, devido às limitações do kit educacional da LEGO, foi restringido o número de tarefas que cada robô poderia realizar. Isso ocorre, pois ao decorrer das realizações das tarefas, o robô apresenta um crescente erro em sua posição, após quatro tarefas realizadas por um mesmo robô, esse erro se torna muito significativo para estabelecer a posição correta do robô. Mesmo com essa dificuldade, a plataforma e a arquitetura garantiram a coordenação dos robôs com o comportamento esperado e dentro das políticas e especificações determinadas.

6 CONCLUSÕES E TRABALHOS FUTUROS

Com o desenvolvimento de novas tecnologias para robôs móveis autônomos, é fundamental explorar novas maneiras de coordenar frotas compostas de diversos tipos de robôs. Nesse contexto, essa dissertação teve como objetivo principal a proposta de um sistema de coordenação por meio de uma arquitetura dividida em três módulos de coordenação. Essa proposta difere-se dos trabalhos já estabelecidos, dispondo de um módulo específico para determinar o tipo de robô para executar a tarefa considerando as necessidades do ambiente. Com o intuito de tratar de forma genérica (sem um ambiente, políticas ou funções específicas) a essa alocação de tarefas para frotas heterogêneas, identifica-se a dificuldade de comparar quantitativamente essa proposta com os trabalhos já estabelecidos.

Com os estudos e as plataformas disponíveis por fabricantes que tratam da coordenação de robôs móveis com arquiteturas, foi identificada a dificuldade de coordenar frotas com vários tipos de robôs, principalmente aquelas compostas por robôs com funções diferentes. Com isso, a arquitetura proposta por essa dissertação estabelece uma maior flexibilidade para aplicações de políticas de gerenciamento, em comparação com estratégias de alocações de tarefas simples sem considerar políticas e regras do ambiente.

Para aumentar a flexibilidade desses sistemas com frotas de robôs heterogêneos, foram propostos dois tipos de tarefas: uma tarefa sendo realizada apenas por um robô e outra por meio da colaboração de dois robôs de tipos distintos. Sendo que, o usuário, empresa ou o programador de tarefas estabelece quais os robôs podem realizar a tarefa juntamente com a ordem de prioridades dos tipos de robôs para a execução dessas tarefas.

Proposta a arquitetura, foi apresentada uma plataforma desenvolvida para a visualização dos resultados obtidos através do uso da arquitetura. A plataforma apresentou uma interface que garantiu a determinação de características do sistema como a quantidade de robôs por tipo, estabelecendo uma flexibilidade no tamanho da frota. Também proporcionou a implementação de supervisores representados por autômatos, também não apresentando problemas para a implementação de múltiplos supervisores considerando a abordagem modular local.

A plataforma garantiu uma grande flexibilidade para o desenvolvimento de testes dentre as muitas características possíveis desses sistemas. Para a simulação da frota de robôs, possibilitou a simulação da bateria (consumo e o carregamento), além de disponibilizar o posicionamento de cada robô no ambiente. Apresentando assim as possibilidades de implementações dos algoritmos de otimização de escolha, como o algoritmo guloso.

Além do comportamento do coordenador no simulador, a plataforma garantiu a possibilidade da validação por meio de robôs montados com o kit LEGO EV3 e um ambiente construído em laboratório. Essa validação demonstra e estabelece as possibilidades do uso da arquitetura para ambientes com frotas reais.

Considerando a plataforma e a arquitetura, foram estabelecidos cenários distintos com frotas e tipos de tarefas previamente determinadas para a realização de testes. Assim, foram

submetidas sequências de tarefas específicas para a visualização do comportamento do sistema.

Com as respostas e comportamento do sistema analisado nos testes, a arquitetura e suas soluções garantiram o funcionamento total do sistema, desde o requerimento da tarefa até ser executada por um ou mais robôs. O primeiro exito observado é na capacidade de tratamento das tarefas mediante às respostas do coordenador. Com a solução, ao identificar uma tarefa que não pode ser realizada com os robôs disponíveis no momento, assim sendo rejeitada. Contudo, o sistema submete para a análise outros tipos de tarefa, sempre tentando otimizar o uso da frota, mas tentando respeitar a ordem de chegada na lista de tarefas.

Além disso, com as sequências de tarefas propostas é possível considerar que o sistema garantiu as políticas de prioridades entre os tipos de robôs. Garantindo o comportamento do coordenador sobre a escolha entre os tipos de robôs por meio dos supervisores obtidos pela TCS. Com os modelos genéricos, apresentados para a solução do módulo de Escolha do Tipo de Robôs, fornecem a flexibilidade na definição das tarefas que podem ser executadas por uma frota. Além disso, esses modelos estabelecem uma facilidade para acrescentar novas tarefas ao sistema, sendo ela a tarefa simples ou colaborativa entre dois tipos de robôs.

Um aspecto importante observado através das respostas de aceite e rejeito dados pelo módulo de Escolha do Tipo de Robô, são as possíveis análises obtidas através da interpretação dessas respostas. Além da informação de que uma tarefa foi rejeitada, as respostas do sistema trazem a informações dos tipos de robôs que mais causaram rejeitos. Desse modo é possível elaborar um relatório de rejeito, que pode auxiliar na compra ou aquisição de novos robôs para a frota de robôs.

São apresentadas a seguir as contribuições da arquitetura de coordenação proposta pela presente dissertação:

- Sistema de coordenação funcional que garante o funcionamento geral de um sistema com frotas de robôs;
- Maior facilidade e flexibilidade de gerenciar frotas com vários tipos de robôs com capacidades e funções distintas;
- Aumento na flexibilidade no gerenciamento de frotas por meio de políticas de prioridade entre tipos de robôs diferentes, definida pelos projetistas;
- Aumento na flexibilidade para o número de robôs disponíveis de frotas de robôs, ainda assim garantindo a coordenação considerando essa escalabilidade;
- Garantia do comportamento de parte do sistema por métodos formais;
- Uma solução para o Tratamento de Tarefas que evita robôs ociosos, mas ainda considerando a ordem de requerimento das tarefas;

- Apresenta uma flexibilidade para escolher o robô que deve realizar a tarefa, com o usuário podendo escolher o algoritmo de escolha e as características mais importantes para essa definição;
- A arquitetura garante uma flexibilidade para soluções distintas entre os módulos de coordenação, o que possibilita estudos sobre quais estratégias mais adequadas para cada módulo;
- O sistema de coordenação apresentou-se funcional para diferentes cenários, com variações entre tipos de robôs, tarefas e ambientes (simulado e com robôs reais).

Além das contribuições sobre o uso da arquitetura, a plataforma para simulação que traz contribuições importantes para o trabalho. Dentre as contribuições mediantes da plataforma desenvolvida:

- A plataforma possibilitou a variação na quantidade de robôs, com cenários que podem apresentar de 0 até 4 tipos de robôs funcionais;
- A implementação de supervisores por meio de dicionários garantiu o uso de métodos formais com representação em autômatos, sem apresentar problemas e garantindo um bom funcionamento da plataforma;
- A plataforma forneceu informações e dados suficientes para observar o funcionamento da arquitetura e gerar discussões;
- Apresentou a flexibilidade necessária para submeter as soluções de cada módulo de coordenação;
- Disponibilizou ferramentas de simulação e recursos para comunicação com robôs reais (possibilitando coordenação com quaisquer robôs com capacidade de comunicação por meio do protocolo MQTT).

Contudo, é importante destacar as limitações dadas pela plataforma observadas com as soluções apresentadas pela dissertação. A primeira restrição é em uma necessidade de implementação de tarefas diferentes das descritas pelos modelos genéricos, a exemplo de uma tarefa que seja necessária mais de dois robôs para a execução. Além disso, outra limitação é para uma tarefa que seja necessária a colaboração de dois robôs de um mesmo tipo, para esse caso é necessária uma adaptação na proposta genérica de tarefa colaborativa.

Outra dificuldade identificada pela dissertação foram as limitações com o uso do kit educacional da LEGO. Com o robô montado e disponível para os testes, há a limitação dos sensores presentes no robô que gera erros e incertezas sobre a posição exata do robô conforme a execução das tarefas. Essa dificuldade, limita na execução de determinadas sequências de testes, principalmente nas sequências que um robô executa inúmeras tarefas. Além disso, pelo tamanho

dos robôs utilizados, há uma limitação no tamanho do ambiente para testes, sendo necessária a elaboração de uma célula com dimensões consideráveis para um robô estar posicionado.

6.1 TRABALHOS FUTUROS

Analisando as conclusões e as limitações sobre essa dissertação, é possível determinar as alternativas para trabalhos futuros, entre eles:

- Determinar um ambiente real com tipos de robôs diferentes e tarefas específicas;
- Propor uma aplicação real considerando frotas de robôs heterogêneos;
- Propor alternativas de políticas de gerenciamentos de tarefas, com diversas características que o usuário escolha a política como solução;
- Analisar alternativas de módulos de coordenação para serem acrescentados e aumentar a flexibilidade da plataforma quanto aos ambientes de aplicação;
- Propor uma solução dinâmica para a escolha do tipo de robôs, propondo novas possibilidades no gerenciamento da frota;
- Propor alternativas para elaboração de novas tarefas com características diferentes apresentadas pela dissertação.

REFERÊNCIAS

ABB. **Driving the next generation of mobile robotics**. 2023. Https://amr.robotics.abb.com/. [Online; accessed 31-January-2023]. Citado 3 vezes nas páginas 7, 23 e 24.

AN, X. et al. Multi-robot systems and cooperative object transport: Communications, platforms, and challenges. **IEEE Open Journal of the Computer Society**, v. 4, p. 23–36, 2023. Citado na página 31.

BRASSARD, G.; BRATLEY, P. **Fundamentals of Algorithmics**. USA: Prentice-Hall, Inc., 1996. ISBN 0133350681. Citado 2 vezes nas páginas 29 e 30.

CAI, K. Warehouse automation by logistic robotic networks: a cyber-physical control approach. **Frontiers of Information Technology Electronic Engineering**, v. 21, p. 693–704, 05 2020. Citado na página 26.

CASSANDRAS, C.G.; LAFORTUNE, S. **Introduction to Discrete Event Systems**. [S.l.: s.n.], 2021. ISBN 978-3-030-72272-2. Citado 6 vezes nas páginas 16, 33, 34, 35, 37 e 38.

CECHINEL, A. K. et al. Multi-robot task allocation using island model genetic algorithm ∗∗this study was financed by the coordenação de aperfeiçoamento de pessoal de nível superior – brasil (capes) – finance code 001. **IFAC-PapersOnLine**, v. 54, n. 1, p. 558–563, 2021. ISSN 2405-8963. 17th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2021. Citado 2 vezes nas páginas 28 e 29.

CHEN, J. Y. C.; BARNES, M. J.; HARPER-SCIARINI, M. Supervisory control of multiple robots: Human-performance issues and user-interface design. **IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)**, v. 41, n. 4, p. 435–454, 2011. Citado na página 31.

CHEN, Peter C. Y.; WONHAM, Walter. Real-time supervisory control of a processor for non-preemptive execution of periodic tasks. **Real-Time Systems**, v. 23, p. 183–208, 11 2002. Citado na página 19.

COMPUTING, Riverbank. **What is PyQt?** 2023. Https://www.riverbankcomputing.com/software/pyqt/. [Online; accessed 17-January-2023]. Citado 2 vezes nas páginas 31 e 58.

DAI, P. et al. A framework for multi-robot coverage analysis of large and complex structures. **Journal of Intelligent Manufacturing**, v. 33, 06 2022. Citado na página 26.

DULCE-GALINDO, J.A. et al. Distributed supervisory control for multiple robot autonomous navigation performing single-robot tasks. **Mechatronics**, v. 86, p. 102848, 2022. ISSN 0957-4158. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957415822000769>. Citado na página 18.

DULCE-GALINDO, J. A. et al. Autonomous navigation of multiple robots using supervisory control theory. In: **2019 18th European Control Conference (ECC)**. [S.l.: s.n.], 2019. p. 3198–3203. Citado 2 vezes nas páginas 18 e 49.

FABIAN, M.; HELLGREN, A. Plc-based implementation of supervisory control for discrete event systems. In: **Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)**. [S.l.: s.n.], 1998. v. 3, p. 3305–3310 vol.3. Citado 2 vezes nas páginas 40 e 68.

FLORENCIO, J. L. et al. A framework for automatic coordination of request-oriented deployment of multi-uav with cost optimization. In: **2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)**. [S.l.: s.n.], 2018. p. 1–8. Citado 2 vezes nas páginas 18 e 49.

FRAGAPANE, G. et al. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. **European Journal of Operational Research**, v. 294, 01 2021. Citado 3 vezes nas páginas 22, 23 e 25.

GROUP, The LEGO. **Getting started with LEGO® MINDSTORMS Education EV3 MicroPython**. 2023. Https://pybricks.com/ev3-micropython/. [Online; accessed 31-January-2023]. Citado na página 32.

HUANG, Y.; ZHANG, Y.; XIAO, H. Multi-robot system task allocation mechanism for smart factory. **2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)**, p. 587–591, 2019. Citado na página 16.

HUGH, J. **Integration and Automation of Manufacturing Systems**. [S.l.: s.n.], 2001. Citado na página 22.

IFR, International Federation of Robotics. **A Mobile Revolution**: How mobility is reshaping robotics. 2021. Https://ifr.org/papers/a-mobile-revolution-how-mobility-is-reshaping-robotics. [Online; accessed 01-July-2022]. Citado 3 vezes nas páginas 22, 25 e 27.

ISO, Organização Internacional de Normalização. **ISO 8373:2021**: Robotics — vocabulary. 3. ed. [S.l.], 2021. [Online; accessed 01-July-2022]. Citado 2 vezes nas páginas 22 e 41.

JANARTHANAN, V.; GOHARI, P.; SAFFAR, A. Formalizing real-time scheduling using priority-based supervisory control of discrete-event systems. **IEEE Transactions on Automatic Control**, v. 51, n. 6, p. 1053–1058, 2006. Citado na página 19.

JU, C.; SON, H. Il. Modeling and control of heterogeneous agricultural field robots based on ramadge–wonham theory. **IEEE Robotics and Automation Letters**, v. 5, n. 1, p. 48–55, 2020. Citado 5 vezes nas páginas 7, 16, 19, 20 e 44.

JU, C.; SON, H. Il. Modeling and control of heterogeneous field robots under partial observation. **Information Sciences**, v. 580, p. 419–435, 2021. ISSN 0020-0255. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025521008793>. Citado 2 vezes nas páginas 28 e 29.

JU, C.; SON, H. Il. A hybrid systems-based hierarchical control architecture for heterogeneous field robot teams. **IEEE Transactions on Cybernetics**, v. 53, n. 3, p. 1802–1815, 2023. Citado na página 18.

KHAMIS, A.; HUSSEIN, A.; ELMOGY, A. Multi-robot task allocation: A review of the state-of-the-art. In: ____. **Cooperative Robots and Sensor Networks 2015**. Cham: Springer International Publishing, 2015. p. 31–51. ISBN 978-3-319-18299-5. Disponível em: <https://doi.org/10.1007/978-3-319-18299-5_2>. Citado na página 29.

KONG, X. et al. Multi-robot task allocation strategy based on particle swarm optimization and greedy algorithm. In: **2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)**. [S.l.: s.n.], 2019. p. 1643–1646. Citado na página 30.

LI, J. et al. Research on hierarchical control architecture for autonomous underwater vehicle. p. 483–487, 2008. Citado 2 vezes nas páginas 19 e 44.

LI, Z. et al. A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand. **Journal of Intelligent Manufacturing**, v. 31, 02 2020. Citado na página 29.

LIU, Y.; FICOCELLI, M.; NEJAT, G. A supervisory control method for multi-robot task allocation in urban search and rescue. In: **2015 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)**. [S.l.: s.n.], 2015. p. 1–6. Citado 2 vezes nas páginas 19 e 44.

LOPES, Y. et al. Local modular supervisory implementation in microcontroller. In: **9th International Conference on Modeling, Optimization Simulation**. [S.l.: s.n.], 2012. Citado 2 vezes nas páginas 40 e 67.

LOPES, Y. et al. Supervisory control theory applied to swarm robotics. **Swarm Intelligence**, v. 10, p. 65–97, 03 2016. Citado 2 vezes nas páginas 18 e 49.

MIR. **MIR FLEET**. 2023. Https://www.mobile-industrial-robots.com/solutions/mir-accessories/mir-fleet/. [Online; accessed 31-January-2023]. Citado na página 26.

MIR, Mobile Industrial Robots A/S. **Get Started with AMRs**. 2023. Https://www.mobile-industrial-robots.com/insights/get-started-with-amrs/. [Online; accessed 31-January-2023]. Citado 3 vezes nas páginas 7, 23 e 24.

MQTT. **MQTT: The Standard for IoT Messaging**. 2023. Https://mqtt.org/. [Online; accessed 31-January-2023]. Citado na página 33.

OMRON. **MobilePlanner Tablet Mobile Robot Management Software**. 2023a. Https://automation.omron.com/en/us/products/family/Mobile%20Planner. [Online; accessed 31-January-2023]. Citado 4 vezes nas páginas 7, 26, 27 e 28.

OMRON. **Robôs Móveis**. 2023b. Https://automation.omron.com/pt/br/products/families/mobile-robots. [Online; accessed 31-January-2023]. Citado 3 vezes nas páginas 7, 23 e 24.

OTTOMOTORS. **OTTO Fleet Manager**. 2023. Https://ottomotors.com/fleet-manager. [Online; accessed 31-January-2023]. Citado 3 vezes nas páginas 7, 26 e 27.

PEREIRA, D. et al. Multi-robot coordination for a heterogeneous fleet of robots. In: **ROBOT2022: Fifth Iberian Robotics Conference**. Cham: Springer International Publishing, 2023. p. 229–240. Citado na página 28.

PEREIRA, Jorge. **Sending and Receiving Messages with MQTT**. 2023. Https://www.ev3dev.org/docs/tutorials/sending-and-receiving-messages-with-mqtt/. [Online; accessed 31-January-2023]. Citado na página 33.

PICKEM, D. et al. The robotarium: A remotely accessible swarm robotics research testbed. p. 1699–1706, 2017. Citado 2 vezes nas páginas 31 e 60.

PINHEIRO, L. P. et al. Nadzoru: A software tool for supervisory control of discrete event systems. **IFAC-PapersOnLine**, v. 48, n. 7, p. 182–187, 2015. ISSN 2405-8963. 5th IFAC International Workshop on Dependable Control of Discrete Systems. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405896315007399>. Citado 2 vezes nas páginas 39 e 76.

QUEIROZ, M. H. de; CURY, J. E. R. Modular supervisory control of large scale discrete event systems. In: ____. **Discrete Event Systems: Analysis and Control**. Boston, MA: Springer US, 2000. p. 103–110. ISBN 978-1-4615-4493-7. Disponível em: <https://doi.org/10.1007/978-1-4615-4493-7_10>. Citado 2 vezes nas páginas 38 e 39.

QUEIROZ, M. H. de; CURY, J. E. R. Controle supervisório modular de sistemas de manufatura. **Sba Controle Automação Sociedade Brasileira de Automatica**, v. 13, p. 123–133, 05 2002. Citado 2 vezes nas páginas 39 e 45.

QUERALTA, J. P. et al. Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. **IEEE Access**, v. 8, p. 191617–191643, 01 2020. Citado na página 27.

RAMADGE, P.; WONHAM, W. Supervisory control of a class of discrete event systems. **SIAM Journal on Control and Optimization**, v. 25, p. 206–230, 01 1987. Citado na página 37.

RAMADGE, P.J.G.; WONHAM, W.M. The control of discrete event systems. **Proceedings of the IEEE**, v. 77, n. 1, p. 81–98, 1989. Citado 3 vezes nas páginas 16, 37 e 38.

SANTOS, J. et al. A* based routing and scheduling modules for multiple agvs in an industrial scenario. **Robotics**, v. 10, n. 2, 2021. ISSN 2218-6581. Disponível em: <https://www.mdpi.com/2218-6581/10/2/72>. Citado na página 67.

SARAVANAN, S. et al. Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems. **Industrial Robot**, 09 2020. Citado 4 vezes nas páginas 16, 28, 29 e 30.

SIMON, M. E. et al. Multi-robots coordination system for urban search and rescue assistance based on supervisory control theory. **Journal of Control, Automation and Electrical Systems**, Springer, v. 34, p. 484–495, 2023. Citado 5 vezes nas páginas 7, 19, 20, 21 e 44.

SOSTERO, M. Automation and robots in services: Review of data and taxonomy. **JRC Working Papers Series on Labour, education and Technology**, 12 2020. Citado na página 27.

TATSUMOTO, Y. et al. Application of online supervisory control of discrete-event systems to multi-robot warehouse automation. **Control Engineering Practice**, v. 81, p. 97–104, 2018. ISSN 0967-0661. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0967066118305070>. Citado 2 vezes nas páginas 18 e 32.

TECH, Georgia. **What is the Robotarium?** 2023. Https://www.robotarium.gatech.edu/. [Online; accessed 17-January-2023]. Citado 3 vezes nas páginas 17, 32 e 61.

TEIXEIRA, M.; CURY, J. E. R.; QUEIROZ, M. H. de. Local modular supervisory control of des with distinguishers. In: **ETFA2011**. [S.l.: s.n.], 2011. p. 1–8. Citado na página 39.

TORRICO, C. R.C.; LEAL, A. B.; WATANABE, A. T.Y. Modeling and supervisory control of mobile robots: A case of a sumo robot. **IFAC-PapersOnLine**, v. 49, n. 32, p. 240–245, 2016. ISSN 2405-8963. Cyber-Physical Human-Systems CPHS 2016. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405896316328932>. Citado na página 18.

VERMA, J.; RANGA, V. Multi-robot coordination analysis, taxonomy, challenges and future scope. **Journal of Intelligent Robotic Systems**, v. 102, 04 2021. Citado 4 vezes nas páginas 16, 26, 27 e 47.

S. Ware, Y. Sun, B. W. J. Soon, L. Lin e R. Su. **Automated guided vehicle management system and method**. Depositante: Delta Electronics International Singapore Pte Ltd: US 2021/0200240 A1, 31 dez. 2019, 14 set, 2020. Depósito: 1 jul. 2021. Concessão: 1 jul. 2021. Citado 2 vezes nas páginas 16 e 19.

WILSON, S. et al. The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems. **IEEE Control Systems Magazine**, v. 40, n. 1, p. 26–44, 2020. Citado na página 32.