UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TIAGO GIACOMELLI ALVES

SISTEMA DE CONTROLE DE POSE PARA UMA CADEIRA DE RODAS INTELIGENTE

Porto Alegre
2018

TIAGO GIACOMELLI ALVES

SISTEMA DE CONTROLE DE POSE PARA UMA CADEIRA DE RODAS INTELIGENTE

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.
Área de concentração: Controle e Automação

ORIENTADOR: Prof. Dr. Ivan Müller

CO-ORIENTADOR: Prof. Dr. Renato Ventura Bayan Henriques

Porto Alegre
2018

TIAGO GIACOMELLI ALVES

SISTEMA DE CONTROLE DE POSE PARA UMA CADEIRA DE RODAS INTELIGENTE

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Ivan Müller, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Banca Examinadora:

Prof. Dr. Edson Prestes e Silva Júnior, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Profª. Drª. Lucíola Campestrini, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Prof. Dr. Walter Fetter Lages, UFRGS
Doutor pelo Instituto Tecnológico de Aeronáutica – São José dos Campos, Brasil

Coordenador do PPGEE: _____
Prof. Dr. Valner João Brusamarello

Porto Alegre, junho de 2018.

DEDICATÓRIA

Dedico este trabalho a aqueles que são a base da minha vida: minha família, minha namorada e meus amigos.

Aos meus pais, André e Inês, que sempre foram os principais motivadores para que eu chegasse a escrever este texto hoje.

À minha namorada Vanessa, que alegra os meus dias.

Aos amigos e familiares, que compreendem minhas ausências mas não deixam de estar presentes.

AGRADECIMENTOS

Agradeço, primeiramente, à Deus por iluminar o meu caminho e prover a força necessária para que eu tenha chegado até aqui.

Aos meus pais por toda confiança, carinho, dedicação e por fornecerem os meios para que eu corra atrás dos nossos sonhos.

Aos meus orientadores Prof. Dr. Ivan Müller e Prof. Dr. Renato V. B. Henriques, por serem pacientes, dedicados, motivadores e pelos aconselhamentos ao longo do curso de mestrado.

Aos amigos Carlos Solon, Gabriel Schmitz e Guilherme Paim, pela convivência, irmandade e companheirismo.

Ao Programa de Pós-Graduação em Engenharia Elétrica (PPGEE) da UFRGS pela oportunidade e pela formação de qualidade.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) por financiar os meus estudos.

RESUMO

Com o objetivo de aumentar o conforto dos usuários de cadeiras de rodas motorizadas, propõe-se nesta dissertação a implementação de um sistema de controle de posição e orientação que permita a estes usuários percorrer as trajetórias desejadas com segurança, sem que haja necessidade de interação constante com o sistema de comando do dispositivo. Para atender a este objetivo, propõe-se uma metodologia para a implementação de um sistema de controle, capaz de levar o veículo de uma posição inicial até uma posição desejada, ou fazer com que este siga uma trajetória especificada, a partir de comandos recebidos pelo sistema.

O sistema de controle proposto baseia-se em uma lei de controle em cascata, composta por um controlador de pose não-linear e um controlador para o rastreamento de velocidade. São apresentadas duas estratégias de controle de velocidade. A primeira delas utiliza um controlador de velocidade com controladores Proporcional-Integral independentes por junta, enquanto a segunda, utiliza um controlador de velocidade baseado no modelo dinâmico do robô, linearizado por realimentação de estados. Os métodos são implementados utilizando-se o pacote *ros_control*, fornecido pelo *Robot Operating System* (ROS).

A avaliação dos métodos propostos é realizada com um robô móvel de acionamento diferencial, que possui a mesma configuração cinemática que a maioria das cadeiras de rodas motorizadas comerciais. São apresentados resultados de identificação dos parâmetros do modelo dinâmico do robô, bem como de convergência das variáveis controladas utilizando-se os métodos de controle propostos. Os resultados demonstram que os métodos atendem aos objetivos de controle propostos.

Palavras-chave: Tecnologias Assistivas, Robótica, Robôs Móveis, Sistemas de Controle, *Robot Operating System* (ROS).

ABSTRACT

In order to improve the comfort of power wheelchair users, it is proposed in this dissertation the implementation of a position and orientation control system which allow the users to travel the desired trajectories safely, without the need for constant interaction with the device command system. To reach this objective, it is proposed a methodology for the implementation of a control system, able to drive the vehicle from an initial position to the desired one or make it follow a specified trajectory, from commands received in the system.

The proposed control system is based on a cascade control law, composed by a non-linear pose controller and a velocity tracking controller. Two velocity control strategies are proposed. The first one uses a velocity controller composed of two joint-independent Proportional-Integral controllers while the second one uses a velocity controller based on the dynamic model, which is linearized by a state-feedback. The methods are implemented using the ros_control package, provided by the framework Robot Operating System (ROS).

The evaluation of the proposed methods is done with a differential-drive mobile robot, which has the same kinematic configuration as the majority of commercial power wheelchairs. The results of dynamic-model parameter identification, as well as the convergence of the controlled variables by using the proposed control methods, are presented. The results demonstrate that the methods achieve the proposed control objectives.

Keywords: Assistive Technology, Robotics, Mobile Robots, Control Systems, Robot Operating System (ROS).

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AIC	*Actuator Interface Card*
ARX	Auto-regressivo com entradas exógenas
AWF	*Automatic Wall Following Mode*
DP	*Door Passage Mode*
EIQ	*Encoder* Incremental de Quadratura
EERUF	*Error Eliminating Rapid Ultrasonic Firing*
GOA	*General Obstacle Avoidance Mode*
GDL	Graus de Liberdade
GFD	*Grey-fuzzy decision-making*
GPS	*Global Positioning System*
HAL	*Hardware Abstraction Layer*
HRIL	*Hardware Resource Interface Layers*
ICR	Centro de Rotação Instantâneo (*Instantaneous Center of Rotation*)
LURCH	*Let Unleashed Robots Crawl the House*
MIMO	*Multiple-input*, *Multiple-output*
MVFH	*Minimal Vector Field Histogram*
PI	Proporcional-Integral
PID	Proporcional-Integral-Derivativo
RLS	Mínimos Quadrados Recursivo (*Recursive least squares*)
RMR	Robô Móvel Terrestre Movimentado por Rodas
ROS	*Robot Operating System*
SISO	*Single-input*, *single-output*
TA	Tecnologia Assitiva
URDF	*Unified Robot Description Format*
VFF	*Vector Force Field*
VFH	*Vector Field Histogram*

LISTA DE SÍMBOLOS

a	Distância do centro do robô até o eixo das rodas;
α	Ângulo entre os sistemas do robô e das rodas;
α_p	Diferença entre a orientação do robô e o ângulo do erro de posição;
b	Distância entre as rodas e o eixo de simetria;
$\boldsymbol{B}(\boldsymbol{x})$	Dinâmica do modelo cinemático de pose;
β	Ângulo da roda orientável;
$\boldsymbol{\beta}$	Vetor de ângulos das rodas orientáveis;
$\boldsymbol{\beta}_c$	Vetor de ângulos das rodas orientáveis não centradas;
$\boldsymbol{\beta}_s$	Vetor de ângulos das rodas orientáveis centradas;
d	Distância entre o centro da roda e o seu ponto de giro;
d_{cm}	Distância das rodas ao centro de massa do robô;
$dim(\cdot)$	Dimensão de $\cdot$;
$\Delta D(k)$	Deslocamento linear do robô;
$\Delta D_d(k)$	Deslocamento linear da roda direita do robô;
$\Delta D_e(k)$	Deslocamento linear da roda esquerda do robô;
$\Delta\Theta(k)$	Deslocamento angular do robô;
e	Erro entre a referência e o valor medido;
$\boldsymbol{e_u}$	Vetor do erro de velocidade;
e_{u_1}	Erro de velocidade linear;
e_{u_2}	Erro de velocidade angular;
e_l	Erro de velocidade da roda esquerda;
e_r	Erro de velocidade da roda direita;
e_p	Módulo do erro de posição em coordenadas polares;
ξ	Coeficiente de amortecimento;
$\boldsymbol{\xi}_o$	Vetor de pose do robô descrito no sistema de coordenadas inercial;
$\boldsymbol{\xi}_c$	Vetor de pose do robô descrito no sistema de coordenadas do robô;

$f(x)$	Função não-linear de x;
$g(x)$	Função não-linear de x;
$h(x)$	Função não-linear de x;
$\boldsymbol{\eta}$	Variável de entrada do modelo cinemático;
δ_m	Grau de mobilidade;
δ_s	Grau de dirigibilidade;
δ_M	Grau de manobrabilidade;
$\boldsymbol{I}$	Matriz identidade;
$\boldsymbol{J}(\boldsymbol{q})$	Jacobiano das restrições;
k	Ganho proporcional;
K	Constante de ganho;
k_d	Coeficiente de ganho derivativo;
K_G	Constante de ganho genérica;
k_i	Coeficiente de ganho integral;
k_p	Coeficiente de ganho proporcional;
$L_f h_i(x)$	Derivada de Lie da saída $h_i(x)$ com relação à $f(x)$;
$L_{g_j} h_i(x)$	Derivada de Lie da saída $h_i(x)$ com relação à $g_j(x)$;
N	Número total de rodas;
N_c	Número total de rodas orientáveis não centradas;
N_f	Número total de rodas fixas;
N_s	Número total de rodas orientáveis centradas;
N_{sw}	Número total de rodas suecas;
$\mathcal{N}(\cdot)$	Espaço nulo de $\cdot$;
$NP_d(k)$	Número de pulsos lidos pelo *encoder* direito;
$NP_e(k)$	Número de pulsos lidos pelo *encoder* esquerdo;
$\boldsymbol{\mu}$	Vetor dos multplicadores de Lagrange μ_i;
μ_i	Multiplicador de Lagrange;
$\boldsymbol{\lambda}$	Vetor dos coeficientes de ponderação λ_i;
λ_i	Coeficiente de ponderação > 0;
$\boldsymbol{\gamma}$	Vetor dos coeficientes de ponderação γ_i;
γ_i	Coeficiente de ponderação > 0;
Γ	Ângulo da direção de velocidade nula da roda sueca;
φ	Deslocamento angular da roda;
$\boldsymbol{\varphi}$	Vetor de deslocamentos angulares da rodas;

Π	Número de pulsos por revolução dos *encoders*;
ψ_p	Ângulo do erro de posição em coordenadas polares;
Ψ	Ângulo entre a direção de movimento da roda e a direção X_o;
$\boldsymbol{q}$	Vetor de coordenadas de configuração;
${}^c\boldsymbol{R}_o$	Matriz de rotação ortogonal entre $\{X_o, Y_o, \theta_o\}$ e $\{X_c, Y_c, \theta_c\}$;
$\rho(\cdot)$	*Rank* de $\cdot$;
$\boldsymbol{S}(\boldsymbol{q})$	Dinâmica do modelo cinemático de configuração;
$\{X_c, Y_c, \Theta_c\}$	Definição do sistema de coordenadas do robô;
$\{X_o, Y_o, \Theta_o\}$	Definição do sistema de coordenadas inercial;
$\phi_i(k)$	Vetor de regressores i;
$\theta_i(k)$	Vetor de constantes i;
T_d	Tempo derivativo;
T_i	Tempo integral;
τ	Torque fornecido pelo atuador;
$\boldsymbol{\tau}$	Vetor dos torques fornecidos pelos atuadores;
$\boldsymbol{\tau}_o$	Vetor de entradas do modelo dinâmico de configuração;
$\boldsymbol{\tau}_s$	Vetor dos torques aplicados para orientação das rodas centradas;
$\boldsymbol{\tau}_c$	Vetor dos torques aplicados para orientação das rodas não centradas;
$\boldsymbol{\tau}_\varphi$	Vetor dos torques aplicados para rotação das rodas;
$\boldsymbol{\nu}$	Vetor de entradas equivalentes;
v	Velocidade Linear;
u	Variável de entrada genérica do modelo no espaço de estados;
$\boldsymbol{u}$	Vetor de entradas genérico do modelo no espaço de estados;
$\boldsymbol{u}_r$	Vetor de referências de velocidade;
u_{r_1}	Referência de velocidade linear;
u_{r_2}	Referência de velocidade angular;
V	Candidata à função de Lyapunov;
V_a	Tensão de armadura;
$\boldsymbol{V}_a$	Vetor de entradas do modelo dinâmico aumentado;
V_{a_r}	Tensão aplicada ao atuador direito;
V_{a_l}	Tensão aplicada ao atuador esquerdo;
x	Variável de estado genérica;
$\boldsymbol{x}$	Vetor de estados genérico;
$\boldsymbol{\zeta}$	Derivada de $\boldsymbol{\beta}_s$;
ω	Velocidade angular;

LISTAGENS

SUMÁRIO

1 INTRODUÇÃO

A deficiência faz parte da condição humana e quase todas as pessoas, em algum momento de suas vidas, experimentarão (diretamente ou através de algum familiar) uma deficiência temporária ou permanente. À medida em que se enfrenta o envelhecimento, a tendência é que sejam observadas dificuldades cada vez maiores com relação à funcionalidade do próprio corpo (WORLD HEALTH ORGANIZATION, 2011, 2015).

Mais de um bilhão de pessoas, quase 15% da população mundial, sofrem com algum tipo de deficiência (WORLD HEALTH ORGANIZATION, 2015). Estima-se que quase 200 milhões de pessoas apresentam, atualmente, dificuldades consideráveis na execução de tarefas simples (NASCIMENTO JÚNIOR, 2016). Estes números devem continuar a crescer nos próximos anos, principalmente, devido ao envelhecimento populacional e ao aumento dos casos de doenças crônicas como diabetes, problemas cardiovasculares e câncer (NASCIMENTO JÚNIOR, 2016; WORLD HEALTH ORGANIZATION, 2015).

O número de pessoas com deficiência no Brasil é expressivo. Segundo o Censo de 2010 (BRASIL, 2012), 45.606.048 brasileiros sofrem de algum tipo de deficiência, seja ela física, auditiva, motora ou intelectual. Isto equivale, em números da época, a 23,9% da população do Brasil. As deficiências visual e motora apresentam maior incidência, afetando, respectivamente, 18,6% e 7% da população total. A ocorrência de deficiências auditiva e mental ou intelectual seguiu as proporções de 5,10% e 1,40% da população total.

Novas tecnologias vêm sendo propostas no sentido de minimizar o impacto dos problemas impostos tanto pelas condições físicas dos portadores de necessidades especiais, quanto pelos problemas de acessibilidade enfrentados nos ambientes aos quais frequentam. As pesquisas nesta área levaram ao desenvolvimento de um novo campo do conhecimento, denominado Tecnologia Assistiva (ALVES DE OLIVEIRA NETO, 2013; YUSIF; SOAR; HAFEEZ-BAIG, 2016). Uma definição formal é apresentada por BRASIL (2016), onde o termo tecnologia assistiva é generalizado para produtos, equipamentos, dispositivos, recursos, metodologias, estratégias, práticas e serviços que apresentem como objetivo promover a funcionalidade, relacionada à atividade e a participação da pessoa com deficiência ou mobilidade reduzida, visando sua autonomia, independência, qualidade de vida e inclusão social.

Exemplos clássicos de tecnologias assistivas são as ferramentas de auxílio à mobilidade, projetadas com o objetivo de fornecer condições de movimentação a pessoas cujos deslocamentos são comprometidos pela deficiência física. Dispositivos característicos desta classe incluem cadeiras de rodas, andadores, *scooters*, bengalas, muletas, próteses e dispositivos ortopédicos (YUSIF; SOAR; HAFEEZ-BAIG, 2016).

As cadeiras de rodas dotadas de motores e acionamentos elétricos, ou simplesmente cadeiras de rodas motorizadas, proporcionam mobilidade funcional tanto para pessoas

com deficiências nos membros inferiores quanto para aquelas cuja deficiência encontra-se nos membros superiores. As versões comerciais destes dispositivos vêm apresentando avanços referentes à concepção, mas os sistemas de controle não apresentaram melhorias significativas nas ultimas décadas. Algumas tecnologias de controle acabam por não proporcionar mobilidade e conforto adequados para muitos usuários, carecendo de inovações que permitam com que mais pessoas consigam conduzir estes dispositivos com independência e segurança (DING; COOPER, 2005).

Ainda conforme DING; COOPER (2005), as primeiras cadeiras de rodas motorizadas comerciais são datadas dos anos cinquenta e o modelo que se tornou o padrão de produção apresentava dois motores e um sistema de comando do tipo *joystick*. Desde as primeiras cadeiras de rodas elétricas, a variável mais comumente controlada é a velocidade. Na maioria dos casos, o condutor aplica comandos através de um *joystick*, baseado em sua percepção da velocidade e da direção do veículo e, a partir da leitura da posição do *joystick*, é aplicado um nível de tensão ao enrolamento de cada motor, para que seja alcançada a velocidade desejada.

Com o intuito de permitir que pessoas com deficiência severa possam se locomover com segurança utilizando estes veículos, pesquisas têm sido realizadas com a aplicação de técnicas originadas nas áreas da robótica e da computação às cadeiras de rodas motorizadas, levando ao surgimento das cadeiras de rodas inteligentes ou cadeiras de rodas robóticas (ALVES DE OLIVEIRA NETO, 2013; DING; COOPER, 2005).

Estes veículos possuem características cinemáticas similares às apresentadas por robôs móveis e, como será discutido no Capítulo 3, podem ser representadas por modelos matemáticos que se aplicam a eles. Deste modo, podem ser utilizados robôs móveis como ferramentas para o desenvolvimento de cadeiras de rodas inteligentes. Os primeiros protótipos baseavam-se em robôs móveis aos quais foram adicionados assentos para os usuários. No entanto, a adaptação de cadeiras de rodas comerciais se provou mais adequada, vindo a se tornar a estratégia mais utilizada nos projetos de pesquisa. Esta estratégia traz como benefícios o emprego de uma estrutura mecânica testada e funcional, projetada para ser ergonômica e acomodar o usuário da maneira mais apropriada possível (CALABRESE, 2013; DING; COOPER, 2005; SIMPSON, 2005).

Esta dissertação vincula-se ao Projeto CAPES PROCAD 2013 Tecnologias Assitivas[1], ao qual fazem parte o grupo de Automação e Robótica do Programa de Pós-Graduação em Engenharia Elétrica (PPGEE) da UFRGS, o núcleo de Tecnologias Assistivas do PPGEE da UFES e o PPGEE da UFAM. O projeto tem como objetivo a pesquisa das tecnologias necessárias à concepção e o desenvolvimento de soluções científicas e tecnológicas, de *hardware* e *software*, para a integração de equipamentos de mobilidade, equipamentos de interação com o usuário e um ambiente inteligente.

Neste contexto, propõe-se uma metodologia para o controle de pose de cadeiras de rodas inteligentes. Com esta metodologia, é possível levar o veículo de uma posição inicial até uma posição desejada ou fazer com este siga uma trajetória especificada a partir de comandos recebidos pelo sistema de controle. Inicialmente, a avaliação de desempenho seria realizada utilizando-se a cadeira de rodas motorizada Freedom SX, adquirida pelo grupo de Automação e Robótica do PPGEE-UFRGS e previamente utilizada por MARQUES (2014). Contudo, para o desenvolvimento dos métodos propostos são necessários sensores para a medição dos deslocamentos das rodas, não disponíveis no veículo, e que demandariam modificações na estrutura mecânica para que fossem instalados. Deste

[1]Projeto intitulado Cooperação Acadêmica na área de Sistemas de Automação e Controle para Tecnologias Assistivas.

modo, optou-se por realizar a avaliação de desempenho com a utilização de um robô móvel com acionamento diferencial, que possui a configuração cinemática apresentada pela grande maioria das cadeiras de rodas comerciais e dispõe dos sensores necessários.

1.1 Motivação

A principal motivação deste trabalho reside em contribuir, no âmbito do projeto CAPES PROCAD 2013 Tecnologias Assistivas, com o desenvolvimento de ferramentas e métodos aplicáveis às tecnologias assistivas, em especial às cadeiras de rodas inteligentes, visando aumentar o conforto dos usuários deste tipo de veículo.

1.2 Objetivos

Este trabalho tem como objetivo a implementação de um sistema de controle de posição e orientação que permita aos usuários de cadeiras de rodas motorizadas percorrer trajetórias desejadas com segurança, sem que haja necessidade de interação constante com o sistema de comando do dispositivo. Assume-se que o usuário possua as interfaces ou meios para enviar os comandos de deslocamento para o sistema de controle proposto. Por fim, caracterizam-se como usuários do sistema proposto as pessoas com mobilidade reduzida, capazes de movimentar os membros superiores.

Visando atender o objetivo principal, são enumerados os seguintes objetivos específicos:

- Utilizar o *framework Robot Operating System* (ROS) (QUIGLEY et al., 2009) como plataforma de desenvolvimento, o que possibilita a integração de métodos e algoritmos consolidados na área de Robótica, bem como a extensibilidade deste projeto;
- Realizar um estudo sobre os modelos cinemático e dinâmico de um robô móvel com acionamento diferencial;
- Utilizar um robô móvel como estudo de caso para a avaliação dos sistemas de controle propostos;
- Apresentar uma proposta de controle de posição e orientação para um robô móvel de acionamento diferencial.

1.3 Estrutura da Dissertação

A fim de trazer maior clareza sobre o tema abordado e, também, sobre as soluções propostas, este trabalho está dividido nos seguintes capítulos:

2. Revisão da Literatura: neste capítulo é apresentada uma visão geral sobre as cadeiras de rodas motorizadas e as abordagens utilizadas para a robotização destas;
3. Fundamentação Teórica: abordam-se aqui os modelos matemáticos que podem ser aplicados às cinco configurações cinemáticas possíveis de robôs móveis. Além disto, são discutidos os métodos de estimação de posição e orientação. Por fim, uma discussão sobre os métodos clássicos de controle aplicáveis a robôs móveis é apresentada;

4. Modelo Matemático do Robô Móvel Twil: expõe-se neste capítulo um modelo matemático particularizado para um robô móvel com acionamento diferencial, que inclui os efeitos dinâmicos dos atuadores. Este modelo é posteriormente parametrizado em um formato apropriado para que os parâmetros físicos do modelo sejam identificados;

5. Métodos de Controle: neste capítulo são apresentados os métodos de controle de posição e velocidade que serão avaliados nesta dissertação;

6. Implementação: discute-se aqui as características de implementação das soluções propostas neste trabalho;

7. Resultados: neste capítulo são demonstrados os resultados obtidos com a integração das técnicas sob estudo neste trabalho;

8. Conclusões: aqui estão dispostas a avaliação quanto aos resultados alcançados, considerações em relação às técnicas abordadas e também em relação a trabalhos futuros.

No próximo capítulo são elencadas abordagens utilizadas em cadeiras de rodas inteligentes e as estruturas de controle empregadas nestes dispositivos.

2 REVISÃO DA LITERATURA

A cadeira de rodas é um dispositivo de tecnologia assistiva projetado para possibilitar a locomoção de pessoas que não possuem aptidão física para caminhar. São constituídas tipicamente de uma estrutura metálica tubular, um assento, quatro rodas e um guidão localizado na parte superior traseira. Exigem, contudo, que o usuário possua aptidão física dos membros superiores ou a presença de uma outra pessoa capaz de impor movimento à cadeira, através do guidão traseiro, para a realização dos deslocamentos (BASTOS-FILHO; KUMAR; ARJUNAN, 2014).

Embora o desenvolvimento de chassis mais leves e ergonômicos tenha permitido melhores condições de uso, o emprego da motorização elétrica trouxe maior possibilidade de autonomia para usuários com deficiência física nos membros inferiores e superiores. Desta forma, a cadeira de rodas motorizada tem revolucionado a vida de milhares de pessoas, ajudando no processo de inclusão social e permitindo ao usuário desfrutar de estilos de vida similares aos de pessoas não portadoras de necessidades especiais (DING; COOPER, 2005; BASTOS-FILHO; KUMAR; ARJUNAN, 2014).

Estes veículos são encontrados no mercado sob os mais variados tipos, mas que compartilham as características de motorização elétrica dupla (acionamento diferencial) e sistema de comando de deslocamento através de *joystick* (Figura 1), as quais têm origem nos protótipos desenvolvidos no início dos anos cinquenta (WOODS; WATSON, 2003; DING; COOPER, 2005). Os modelos diferenciam-se principalmente quanto à capacidade de carga, tipo de chassis, possibilidade de elevar o usuário e características dos terrenos e pavimentos por onde podem circular (NASCIMENTO JÚNIOR, 2016; DING; COOPER, 2005; WOODS; WATSON, 2003).

(a) Cadeira de rodas motorizada Freedom Compact 20.
Fonte: FREEDOM (2018)

(b) Cadeira de rodas motorizada WHILL Model-A, com rodas frontais omnidirecionais.
Fonte: WHILL (2018).

Figura 1: Cadeiras de rodas motorizadas comerciais.

As cadeiras motorizadas comerciais, em sua grande maioria, exigem do usuário apti-

dão física para controlar um *joystick* e capacidade visual para identificar e desviar de pessoas, obstáculos e situações de perigo (BASTOS-FILHO; KUMAR; ARJUNAN, 2014). Estes requisitos limitam a abrangência destas tecnologias, uma vez que usuários que possuem determinados tipos de deficiência podem encontrar dificuldades ou até mesmo impossibilidade de utilizar estes dispositivos. Estes usuários incluem, mas não limitam-se a apenas, indivíduos com capacidade visual reduzida, coordenação motora deficitária, deficiências cognitivas, tremores e espasticidade (SIMPSON, 2005).

A necessidade de atender a um número maior de usuários gerou o apelo por interfaces de comando e sistemas de controle mais sofisticados. Utilizando-se dos avanços tecnológicos experimentados pelas áreas da computação e da robótica, pesquisadores passaram a desenvolver protótipos de cadeiras de rodas inteligentes, dotadas de interfaces de comando customizadas para as necessidades dos usuários, e capacidade de navegar de forma autônoma e sem ocorrência de colisões (DING; COOPER, 2005; SIMPSON, 2005). Este trabalho apresenta como foco o estudo dos métodos de controle de deslocamento, aplicados em cadeiras de rodas inteligentes, cujas características são revisadas na Seção 2.2.

2.1 Cadeiras de Rodas Inteligentes

Desde o início dos anos oitenta, o conceito cadeira de rodas inteligente tem sido objeto de estudo em instituições de pesquisa de diversos países. Seu desenvolvimento tem como objetivo o aumento da autonomia de indivíduos que não possuem capacidade de comandar uma cadeira de rodas motorizada convencional, adaptando-a para ser utilizada através de interfaces de comando não convencionais e sistemas de suporte auxiliares (DING; COOPER, 2005; NASCIMENTO JÚNIOR, 2016).

Os primeiros protótipos baseavam-se em robôs móveis aos quais foram adicionados assentos para os usuários. A adaptação de cadeiras de rodas motorizadas comerciais, contudo, se provou mais adequada e tornou-se a estratégia mais utilizada nos projetos de pesquisa. Esta estratégia traz como benefícios o emprego de uma estrutura mecânica testada e funcional, projetada para ser ergonômica e acomodar o usuário da maneira mais apropriada possível (CALABRESE, 2013; DING; COOPER, 2005; SIMPSON, 2005).

Outra vantagem da adaptação de cadeiras de rodas é a possibilidade de reuso dos circuitos eletrônicos de controle e acionamento, especificamente projetados para o sistema de tração do veículo. No entanto, o aproveitamento destes circuitos geralmente demanda que sejam realizadas ou uma nova programação do *software* embarcado ou a engenharia reversa do protocolo, geralmente proprietário, utilizado para a comunicação entre a interface de comando e o sistema de controle e acionamento (CALABRESE, 2013; SIMPSON, 2005).

Em nível de funcionalidades, as cadeiras de rodas inteligentes podem ser classificadas conforme as seguintes categorias:

- Navegação Semi-Autônoma: Esta estratégia, muitas vezes denominada controle compartilhado, baseia-se em primitivas como evitar obstáculos, passar através de uma porta ou corredor estreito, ou seguir uma parede. Estes sistemas de navegação são projetados para auxiliar na realização de manobras difíceis e a trajetória a ser seguida é definida pelo usuário, através de um *joystick* ou interface customizada para suas necessidades. Dependendo do modo de operação, o controle de trajetória é mantido pelo usuário durante a maior parte do tempo e só ocorre intervenção quando determinada situação de risco é identificada, como por exemplo a possibilidade de colisão (CALABRESE, 2013; GONÇALVES, 2013).

- Navegação Autônoma: Nesta modalidade, o sistema comporta-se como um um robô móvel autônomo. O usuário precisa especificar somente um objetivo ou destino e o *software* de controle encarrega-se de realizar o planejamento e a execução da trajetória. Deste modo, a tarefa é realizada com conforto e segurança. As primitivas empregadas em navegação semi-autônoma também aplicam-se a esta modalidade (CALABRESE, 2013; GONÇALVES, 2013).

Os trabalhos desenvolvidos nesta área concentram-se, principalmente, no projeto de interfaces de comando customizadas (BASTOS-FILHO et al., 2014), estruturas mecânicas adaptáveis a diferentes tipos de terrenos e edificações (ALVES DE OLIVEIRA NETO, 2013), tratamento e fusão de dados de sensores, técnicas de navegação (DE LA CRUZ; CELESTE; BASTOS, 2011; GONÇALVES, 2013), modelagem matemática (JOHNSON; AYLOR, 1985; ONYANGO et al., 2009) e métodos de controle (CELESTE et al., 2008; MARTINS et al., 2008; DE LA CRUZ; CELESTE; BASTOS, 2012) e, também, nos sistemas de acionamento (FILGUEIRA, 2011; HAMANAKA, 2002; CANOZ et al., 2016).

Dada a abrangência do tema, uma visão ampla e aprofundada sobre os projetos extrapolaria o escopo deste trabalho. Serão, portanto, brevemente elencados alguns deles e suas respectivas particularidades, com o intuito de apresentar ao leitor os avanços que têm sido alcançados em relação a estas tecnologias. Uma análise mais detalhada pode ser encontrada nos trabalhos de YANCO (2000), SIMPSON (2005), NASCIMENTO JÚNIOR (2016) e LEAMAN; MANH LA (2017).

O trabalho publicado por LOZAC'H et al. (1976), um dos primeiros a tratar de interfaces de comando customizadas, relatou o desenvolvimento de uma unidade de controle de direção e velocidade, de uma cadeira de rodas elétrica, para usuários quadriplégicos. O dispositivo desenvolvido consistia em um transdutor potenciométrico do tipo *joystick*, posicionado atrás da cabeça do usuário e fixado à cadeira de rodas. As informações de deslocamento eram geradas a partir dos movimentos da cabeça sobre o mecanismo, e a reversão do sentido de movimento e o desligamento de emergência eram realizados por meio de botões, acionados com o movimento dos ombros. Os principais fatores levados em conta na concepção do projeto foram segurança contra possíveis traumatismos, conforto e máxima estabilidade de controle em superfícies irregulares. A unidade de controle não era fixada à cabeça do usuário e, portanto, interferia minimamente em outras atividades como falar ou comer.

A primeira cadeira de rodas semi-autônoma foi apresentada por JAFFE (1982). O protótipo denominado SMART ALEC (Universidade Stanford, 1980-1990) baseava-se em uma cadeira de rodas motorizada comercial, a qual foram adicionados *encoders* às rodas, um microcomputador e diversos sensores ultrassônicos. As medições realizadas por dois destes sensores eram utilizadas para realizar a triangulação da posição da cabeça do usuário. Os outros sensores ultrassônicos foram dispostos de modo a identificar obstáculos na frente da cadeira e as paredes nas laterais dela. Este protótipo apresentava como funcionalidades o desvio de obstáculos, possibilidade de seguir uma pessoa mantendo uma distância fixa e seguimento de parede.

A primeira cadeira de rodas totalmente autônoma foi apresentada por MADARASZ et al. (1986). O projeto conhecido por Madarasz *Wheelchair* (Universidade Estadual do Arizona, 1985-1986) foi concebido para operar dentro de um prédio comercial, transportando o usuário, de forma autônoma, de uma localização inicial para uma sala especificada. Foram utilizados, para esta finalidade, um sistema de planejamento de trajetória e um mapa do ambiente. O dispositivo contava com *encoders* acoplados às rodas, utilizados para medição de deslocamento, sistema de visão monocular, aplicado à localização

e verificação de objetos conhecidos (como a numeração das salas), estado do painel do elevador, mudanças no ambiente e, também, para manter a cadeira de rodas centrada nos corredores. Além destes, havia também um sensor ultrassônico do tipo *rangefinder* cujas medidas eram utilizadas para determinar a distância aos objetos ao redor da cadeira (MADARASZ et al., 1986).

O projeto TinMan II (KIPR, 1995), desenvolvido por MILLER; SLACK (1995), foi concebido para auxiliar o usuário na realização de manobras em áreas estreitas, passar por portas e, também, a dirigir-se para locais predeterminados sem ocorrência de colisões. Possibilitava ao usuário a seleção entre um modo totalmente manual e três modos semi-autônomos. Os modos semi-autônomos utilizavam como base as informações de *encoders* acoplados às rodas, para determinar deslocamentos, e medidas de sensores ultrassônicos, infravermelho e de contato, para detectar e evitar obstáculos. Um *joystick* era utilizado como interface de comando para os modos assistido e totalmente manual. No modo assistido, a cadeira seguia a intenção do usuário até que um obstáculo fosse detectado. Neste caso, a orientação seria ajustada para o ângulo seguro mais próximo da intenção do usuário. Os outros dois modos permitiam ao usuário, por meio do uso de botões, a seleção das ações de andar para frente ou girar em torno do próprio eixo (MILLER; SLACK, 1995; YANCO, 2000).

O projeto NavChair (Universidade de Michigan, 1990-2002), desenvolvido por LEVINE; BORENSTEIN; KOREN (1990); LEVINE et al. (1999), trouxe avanços no tratamento de dados de sensores e métodos de detecção de obstáculos. Baseava-se em uma cadeira de rodas motorizada comercial, dotada de um microcomputador e um conjunto de sensores ultrassônicos. Embora a leitura individual destes sensores apresente precisão limitada, foi empregado o método *Error Eliminating Rapid Ultrasonic Firing* (EERUF), desenvolvido por BORENSTEIN; KOREN (1992), para mitigar estes efeitos e criar um mapa das medições de sonar, relativo à região ao redor da cadeira. Este método rejeitava leituras inadequadas dos sensores, através da detecção de padrões temporais inconsistentes com o modo de operação sem erros. A precisão do mapa era aumentada através das informações de deslocamento, obtidas com o processamento das medidas dos sensores de rotação, originalmente acoplados às rodas. Isto possibilitava à NavChair a detecção de obstáculos sob uma resolução angular de cinco graus, em relação ao centro da cadeira (LEVINE et al., 1999). A navegação livre de colisões baseava-se, inicialmente, nos métodos *Vector Field Histogram* (VFH) (BORENSTEIN; KOREN, 1991a,b) e *Vector Force Field* (VFF) (BORENSTEIN; KOREN, 1989), originalmente desenvolvidos para a navegação de robôs móveis. Posteriormente, passaram a ser utilizadas versões modificadas destes métodos. O VFF foi desenvolvido para robôs móveis cilíndricos (BORENSTEIN; KOREN, 1989), onde a representação do robô era realizada por um ponto, e posteriormente adaptado para ser usado com veículos de formato retangular (BORENSTEIN; RASCHKE, 1991).

Conforme (LEVINE et al., 1999, p. 446), ao longo do desenvolvimento da NavChair constatou-se que seriam necessárias modificações no método VFH para torná-lo apropriado à utilização com as cadeiras de rodas. Inicialmente, este método era aplicado a robôs móveis cilíndricos e omnidirecionais, o que simplifica o cálculo das trajetórias para evitar colisões. Contudo, quando aplicado às cadeiras de rodas, com geometria retangular e comportamento não-holonômico, a probabilidade de ocorrência de falhas no desvio de obstáculos era mais elevada. Outra dificuldade encontrada relacionava-se às funções que deveriam ser alcançadas pelo sistema NavChair como, por exemplo, passar por portas. O método VFH não era apropriado para realizar este tipo de função, mantendo a mesma

proteção contra obstáculos em ambientes mais abertos. Além disto, aquele que era considerado um dos pontos fortes do método, quando aplicado à robôs móveis, a capacidade de mover-se em um ambiente com pessoas e obstáculos, e fazer mudanças abruptas de direção, com uma redução mínima de velocidade, não era apropriado para aplicações em cadeiras de rodas (LEVINE et al., 1999). Visando contornar estes problemas, foi desenvolvida uma adaptação do método VFH, denominada *Minimal Vector Field Histogram* (MVFH) (BELL et al., 1994), que juntamente com versão modificada do VFF, passou a ser utilizada pelo sistema NavChair (LEVINE et al., 1999). Contudo, a utilização destes métodos dependia do modo de operação do sistema de navegação. Estes modos eram divididos em três categorias: *General Obstacle Avoidance Mode* (GOA), *Door Passage Mode* (DP) e *Automatic Wall Following Mode* (AWF).

O modo de operação padrão do sistema NavChair era o GOA, cujo objetivo era permitir uma navegação rápida e suave em ambientes lotados, mantendo uma distância segura dos obstáculos. Neste caso, os dois métodos de navegação eram mantidos ativos. O segundo modo, DP, destinava-se a situações em que o NavChair precisaria passar entre dois obstáculos próximos, como por exemplo, uma porta. Neste caso, o modo DP atuaria para centralizar a cadeira em relação à porta, e posteriormente cruzá-la, utilizando somente o método MFVH. Por fim, o modo AWF fazia com que os o sistema NavChair modificasse os comandos recebidos do *joystick* do usuário, de modo que fosse seguida a direção da parede, em relação a um dos lados da cadeira. Neste caso, os métodos MVFH e VFF eram desabilitados, os sonares frontais e da lateral oposta, à parede sendo seguida, eram utilizados para detectar obstáculos, e os remanescentes, aplicados na navegação da cadeira.

O projeto LURCH (Politécnico de Milão, 2007-2015), um acrônimo para *Let Unleashed Robots Crawl the House* (CALABRESE, 2013), baseou-se em uma cadeira de rodas comercial, à qual foram adicionados dois *scanners laser*, dois *encoders*, para a medição dos deslocamentos das rodas, sonares posicionados na parte traseira, uma câmera e um computador. A versão atual dos softwares da LURCH foi desenvolvida utilizando o *framework Robot Operating System* (ROS) (QUIGLEY et al., 2009) e permite ao veículo dois modos de operação: a) manual (*driven by user*), onde a cadeira é comandada através de um *joystick* convencional (ou um *joypad* sem fio) e b) operação pelo computador (*driven by pc*), onde é utilizada uma tela *touch screen* para definir uma referência de posição, para a qual a cadeira deve se deslocar de forma autônoma.

No modo manual, pode ser habilitada a condução assistida (*assisted drive state*). Para esta finalidade, foram implementados nodos no ROS, responsáveis por processar as informações dos sensores e evitar colisões, enquanto o usuário comanda o veículo. Por outro lado, no modo autônomo (*autonomous drive*) é utilizada a *stack* de navegação do ROS, para o planetajamento das trajetórias e desvio dos obstáculos. Por fim, as referências de deslocamento são fornecidas para o sistema de navegação utilizando-se a ferramenta RVIZ do ROS. No RVIZ, disponibilizado ao usuário através da tela *touch screen*, é apresentado o mapa do ambiente, onde o usuário pode marcar os pontos para onde deseja se deslocar.

2.2 Modelos Matemáticos e Métodos de Controle

Desde o surgimento das primeiras cadeiras de rodas motorizadas a variável mais comumente controlada é a velocidade. No entanto, a grande maioria das versões comerciais não apresenta estruturas de realimentação. O condutor aplica comandos através de um joystick, baseado em sua percepção da velocidade e da direção do veículo e, a partir da

leitura da posição do joystick, o circuito de acionamento dos motores aplica o nível de tensão apropriado a cada um deles, para que seja alcançado o perfil de velocidade desejado (DING; COOPER, 2005; NASCIMENTO JÚNIOR, 2016).

Em DING; COOPER (2005) é apresentada uma análise sobre os diversos modelos de cadeiras de rodas motorizadas e os esforços empregados no desenvolvimento destas. Os autores pontuam que avanços vem sendo apresentados em relação ao projeto mecânico, mas os sistemas de controle não apresentaram melhorias significativas nas últimas décadas. Algumas tecnologias de controle acabam por não proporcionar mobilidade e conforto adequados para muitos usuários, carecendo de inovações que permitam com que mais pessoas consigam conduzir estes dispositivos com independência e segurança.

O tipo mais comum de controlador encontrado nestes veículos é o Proporcional-Integral (PI) (DING; COOPER, 2005), independente por junta, utilizado para o rastreamento do perfil de velocidade desejado pelo usuário. Geralmente, a informação de posição do *joystick* é utilizada pelo *software* de controle para gerar a referência de velocidade para cada uma das rodas, que devem ser seguidas pelos controladores PI dos respectivos motores (SILVA, 2007; GONÇALVES, 2013).

Historicamente, utilizavam-se os mesmos controladores para diversas cadeiras de rodas, independente das características do veículo e do usuário, resultado em desempenhos inapropriados ou, em algumas ocasiões, na instabilidade dos sistemas de controle (BROWN; INIGO; JOHNSON, 1989). No entanto, estes controladores podem ser sintonizados com base nas informações sobre o veículo e, também, sobre as características do usuário, permitindo que o sistema de controle seja adaptado às características do condutor (DING; COOPER, 2005).

Com o objetivo de contornar a necessidade de customizar o sistema de controle às características do usuário, em BROWN; INIGO; JOHNSON (1989) é apresentado um controlador de velocidade ótimo e adaptável, baseado em uma modificação do controlador PID.

Um sistema de controle baseado na técnica *grey-fuzzy decision-making* (GFD) foi aplicado a uma cadeiras de rodas motorizada por LUO; CHEN; HSIEN LIN (1999). O controlador utiliza informações de uma bússola digital e dos *encoders* acoplados às rodas para estimar o movimento do veículo. Os parâmetros do sistema são, então, determinados pelo GFD, que possui a vantagem de poder aproximar o comportamento dinâmico do sistema mesmo sob variações nas características do ambiente (DING; COOPER, 2005). Resultados experimentais são apresentados, onde a cadeira consegue seguir um robô móvel, mantendo-se a uma distância de $80\,\mathrm{cm}$ deste, mesmo sob constantes variações de velocidade apresentadas pelo robô.

Ainda segundo DING; COOPER (2005), os sistemas de controle de tração, comuns em automóveis, são pouco usados em cadeiras de rodas motorizadas. O emprego desse tipo de sistema poderia trazer benefícios significativos para os usuários de cadeiras de rodas motorizadas, principalmente em terrenos arenosos, molhados e na neve (NASCIMENTO JÚNIOR, 2016).

Em JOHNSON; AYLOR (1985) é desenvolvido o modelo dinâmico para uma cadeira de rodas, levando-se em conta um ambiente plano, as restrições ao movimento impostas pelas rodas *caster*, o atrito e a ausência de escorregamento das rodas. Neste trabalho, nenhum sistema de controle foi proposto. Os autores apresentam comparações entre os resultados por simulação e as respostas do sistema.

Em ONYANGO et al. (2009) é apresentado um modelo dinâmico mais sofisticado, que leva em conta a inclinação do veículo, o atrito e o escorregamento das rodas. O

modelo é derivado através do formalismo de Lagrange, considerando-se o veículo em um plano inclinado. A esta descrição é aplicada uma linearização entrada-saída baseada em uma transformação de coordenadas e uma lei de controle por realimentação de estados. São expostos resultados de simulação do controle de velocidade em uma superfície seca e em outra com escorregamento.

Conforme NASCIMENTO JÚNIOR (2016), a utilização de técnicas de controle não-linear como *Sliding Modes* não é muito frequente na literatura sobre cadeiras de rodas. No entanto, a utilização deste tipo de método, em conjunto com a aplicação de redes neurais, foi proposta por NGUYEN; NGUYEN; SU (2008).

Um outro tipo de controlador apresentado na literatura utiliza como entrada as forças aplicadas ao empurrar as rodas, e a partir destas forças o controlador altera a velocidade da cadeira (TSAI; HSUEH, 2013; OH; HORI, 2008). Apresenta-se como uma alternativa aos sistemas de controle de movimento totalmente manuais tradicionais, auxiliando o usuário a manter as funcionalidade do corpo, sem que haja a necessidade de aplicar muito esforço para movimentar a cadeira.

Nesta seção foram expostos alguns dos métodos de controle que são aplicados às cadeiras de rodas motorizadas. Estes veículos apresentam configurações cinemáticas similares às dos robôs móveis e, como será discutido no próximo capítulo, podem ser representados por modelos matemáticos que aplicam-se a eles. Desta forma, os métodos desenvolvidos para o controle de robôs móveis podem, também, ser aplicados às cadeiras de rodas motorizadas. Na Seção 3.4 são apresentadas as técnicas clássicas de controle de robôs móveis que, também, podem ser aplicadas a elas.

3 FUNDAMENTAÇÃO TEÓRICA

Esta dissertação apresenta como objeto de estudo as cadeiras de rodas inteligentes. Estes dispositivos possuem características cinemáticas similares às de um robô móvel. Portanto, os robôs móveis sob estudo serão apenas aqueles movimentados por rodas, cuja operação ocorre em ambiente terrestre. Assim, utilizar-se-á, a partir daqui, a terminologia Robô Móvel Terrestre Movimentado por Rodas (RMR) para definir estes dispositivos.

Neste capítulo, serão caracterizados os modelos cinemático e dinâmico de robôs móveis, necessários para o desenvolvimento de leis de controle baseadas nestas representações. Será, também, demonstrado que os RMRs podem ser enquadrados em cinco classes distintas, baseando-se nas características de mobilidade, que são relacionadas à configuração e ao posicionamento das rodas. Por fim, serão caracterizados os principais objetivos de controle e os métodos empregados no controle de robôs móveis.

3.1 Modelagem Matemática de Robôs Móveis

Diversos trabalhos na literatura têm apresentado modelos para configurações cinemáticas particulares de robôs móveis (MUIR; NEUMAN, 1987a; KILLOUGH; PIN, 1992; SIDEK; SARKAR, 2008; PETROV, 2010; DUŠEK; HONC; ROZSÍVAL, 2011; DE LA CRUZ; CARELLI, 2006; MARTINS et al., 2008; DHAOUADI; HATAB, 2013), ou procedimentos sistemáticos para obtenção destas descrições (MUIR; NEUMAN, 1987b; ALEXANDER; MADDOCKS, 1989).

Nesta seção, serão apresentados modelos matemáticos para um RMR genérico, baseado na formulação de CAMPION; BASTIN; DANDREA-NOVEL (1996); CAMPION; CHUNG (2008). Estes modelos podem ser divididos em quatro tipos: a) modelo cinemático de pose, b) modelo cinemático de configuração, c) modelo dinâmico de configuração e d) modelo dinâmico de pose.

Os modelos cinemáticos descrevem o robô em função da velocidade e orientação das rodas, ao passo que os modelos dinâmicos descrevem o robô em função das forças generalizadas aplicadas pelos atuadores. Os modelos de pose utilizam como estados somente a posição e a orientação do robô. Em contrapartida, os modelos de configuração apresentam como estados a pose e outras variáveis internas do robô, como, por exemplo, os deslocamentos das rodas (LAGES, 1998, p. 10).

Nesta dissertação, pressupõe-se que os robôs móveis sob estudo são constituídos de um corpo rígido, cujas rodas não sofrem deformações e que movimentam-se exclusivamente no plano horizontal (CAMPION; BASTIN; DANDREA-NOVEL, 1996). A pose do RMR no ambiente é definida em relação a um sistema de coordenadas inercial $\{X_o, Y_o, \Theta_o\}$ que caracteriza o *frame* sobre o qual ocorre o deslocamento (Figura 2). O sistema de coordenadas do robô é definido por $\{X_c, Y_c, \Theta_c\}$ e caracteriza-se como o *frame*

local fixado ao corpo do RMR. Desta forma, a pose do robô $\boldsymbol{\xi}_o$ descrita no sistema de coordenadas inercial e a matriz de rotação ortogonal ${}^c\boldsymbol{R}_o$ entre $\{X_o, Y_o, \Theta_o\}$ e $\{X_c, Y_c, \Theta_c\}$ são dadas por:

$$\boldsymbol{\xi}_o \triangleq \begin{bmatrix} x_c \\ y_c \\ \theta_c \end{bmatrix} \tag{1}$$

$${}^c\boldsymbol{R}_o = \begin{bmatrix} \cos(\theta_c) & \sin(\theta_c) & 0 \\ -\sin(\theta_c) & \cos(\theta_c) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

A velocidade do robô em relação ao sistema inercial é descrita por $\dot{\boldsymbol{\xi}}_o = \begin{bmatrix} \dot{x}_c & \dot{y}_c & \dot{\theta}_c \end{bmatrix}^T$. O mapeamento das componentes de $\dot{\boldsymbol{\xi}}_o$ no sistema $\{X_c, Y_c, \Theta_c\}$ é dado por:

$$\dot{\boldsymbol{\xi}}_c = {}^c\boldsymbol{R}_o \dot{\boldsymbol{\xi}}_o \tag{3}$$

Uma vez estabelecida a descrição do robô no ambiente, nas subseções a seguir serão caracterizados os tipos de rodas e as restrições impostas devido ao uso de cada um deles. Posteriormente serão considerados os aspectos relacionados à mobilidade dos robôs móveis, baseando-se nas restrições impostas pelas características cinemáticas. Por fim, serão derivados os quatro tipos de modelos, seguindo a formulação de CAMPION; BASTIN; DANDREA-NOVEL (1996).

Embora sejam necessários apenas os modelos de pose para o controle da posição e orientação espacial de um RMR, serão apresentados, também, os modelos de configuração, uma vez que o modelo dinâmico de pose é obtido a partir do modelo dinâmico de configuração (LAGES, 1998, p. 10).

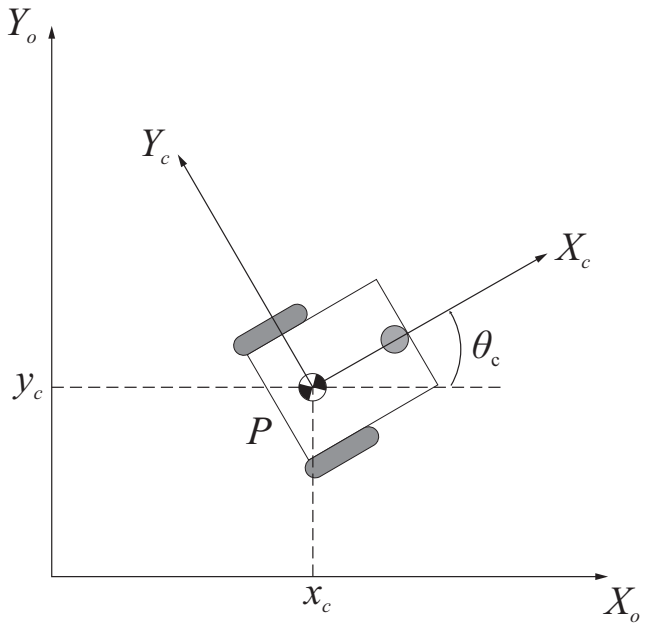


Figura 2: Definição dos sistemas de coordenadas de um RMR genérico.
Fonte: SIEGWART; NOURBAKHSH; SCARAMUZZA (2011, p. 59)

3.1.1 Tipos de Rodas

Serão consideradas duas classes de rodas idealizadas (Figura 3): a) rodas convencionais, que podem ser entendidas como estruturas compostas por aro e pneu, como as encontradas nas cadeiras de rodas tradicionais (Figuras 3a e 3b), ou estruturas compostas por rodas maciças, e b) rodas universais (ou rodas suecas), que possuem na estrutura mecânica roletes ou esferas (Figuras 3c e 3d).

(a) Roda convencional fixa utilizada na parte traseira de uma cadeira de rodas motorizada Freedom Compact 20.
Fonte: FREEDOM (2018)

(b) Roda convencional orientável não centrada utilizada na parte traseira de uma cadeira de rodas motorizada Freedom Compact 20.
Fonte: FREEDOM (2018)

(c) Roda universal.
Fonte: BARROS (2014, p. 33)

(d) Roda universal.
Fonte: BARROS (2014, p. 33)

Figura 3: Modelos de rodas.

Assume-se que o plano de cada roda permaneça vertical durante o movimento e que a rotação ocorre em torno de um eixo horizontal, cuja orientação em relação ao sistema de coordenadas $\{X_c, Y_c, \Theta_c\}$ pode ser fixa ou variável. Considerar-se-á, também, que o ponto de contato com o solo é reduzido a um único ponto no plano (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 267).

Supõe-se que as rodas convencionais satisfaçam à condição de rolar sem deslizar e, portanto, a velocidade do ponto de contato com o solo é zero. Ou seja, tanto a componente de velocidade no plano da roda quanto a componente de velocidade ortogonal a este plano são nulas. Para as rodas suecas, em contrapartida, supõe-se que apenas uma das componentes de velocidade do ponto de contato entre a roda e o plano seja zero durante o movimento. A direção desta componente é arbitrária, mas fixa em relação à orientação da roda (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 267).

A partir destas suposições, pode-se deduzir as restrições ao movimento do robô, impostas por cada uma das rodas acopladas ao chassis. A modelagem completa destas classes de rodas e a análise das restrições foram realizadas por LAGES (1998) e BARROS (2014). Portanto, nas subseções seguintes, serão apresentadas somente as restrições asso-

ciadas a estas classes de rodas.

3.1.2 Rodas Convencionais Fixas e Orientáveis Centradas

As rodas convencionais fixas (Figura 3a) são frequentemente encontradas na parte traseira das cadeiras de rodas motorizadas (Figura 1a), compondo o sistema de tração do veículo. Caracterizam-se pela ausência de liberdade de movimento em relação ao chassis, ou seja, o ângulo entre eles é fixo.

Na Figura 4 é apresentada a descrição deste tipo de roda em relação ao sistema de coordenadas do robô $\{X_c, Y_c, \Theta_c\}$. O ponto *A* representa o local de fixação da roda ao chassis. A posição do ponto *A* no sistema de coordenadas do robô é representada utilizando-se coordenadas polares, por meio da distância *l*, entre os pontos *P* e A, e do ângulo α. A orientação do plano da roda em relação a *l* é representada pelo ângulo β, que neste caso é fixo. O ângulo de rotação da roda sobre seu eixo horizontal é definido por φ e o raio da roda por r.

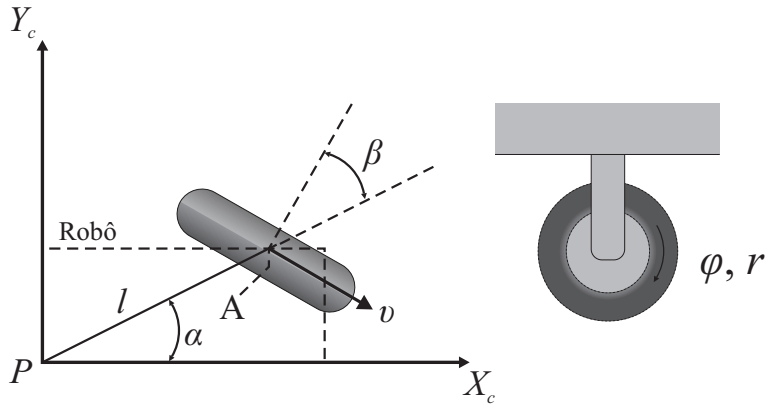


Figura 4: Definição do sistema de coordenadas da roda convencional fixa.
Fonte: CAMPION; CHUNG (2008, p. 393)

A roda orientável centrada, por outro lado, é frequentemente encontrada em triciclos. É um tipo de roda cujo plano pode ser rotacionado em torno de um eixo vertical, que passa pelo seu centro. Na Figura 5 é apresentada a descrição deste tipo de roda em relação ao sistema de coordenadas do robô $\{X_c, Y_c, \Theta_c\}$. A descrição deste tipo de roda é a mesma utilizada para a convencional fixa, com exceção do angulo β, que neste caso não é constante.

Tanto a roda fixa quanto a orientável centrada impõem duas restrições ao movimento do robô. A primeira delas, que representa a restrição ao movimento ao longo do plano da roda, é dada por (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998):

$$\begin{bmatrix} -\sin(\alpha+\beta) & \cos(\alpha+\beta) & l\cos(\beta) \end{bmatrix} {}^{c}\boldsymbol{R}_o \dot{\boldsymbol{\xi}}_o + r\dot{\varphi} = 0 \tag{4}$$

O primeiro termo da soma está relacionado ao movimento total ao longo do plano da roda. Os elementos do vetor representam os mapeamentos de cada uma das velocidades no sistema de coordenadas inercial ($\dot{x}_c$, $\dot{y}_c$ e $\dot{\theta}_c$) às suas contribuições para o movimento no plano da roda. Este movimento, conforme a restrição (4), deve ser igual ao alcançando girando-se a roda, representado pelo segundo termo da soma ($r\dot{\varphi}$).

Além disto, a condição de não deslizamento impõe que a componente do movimento ortogonal ao plano da roda seja zero. Esta condição é representada pela segunda restrição,

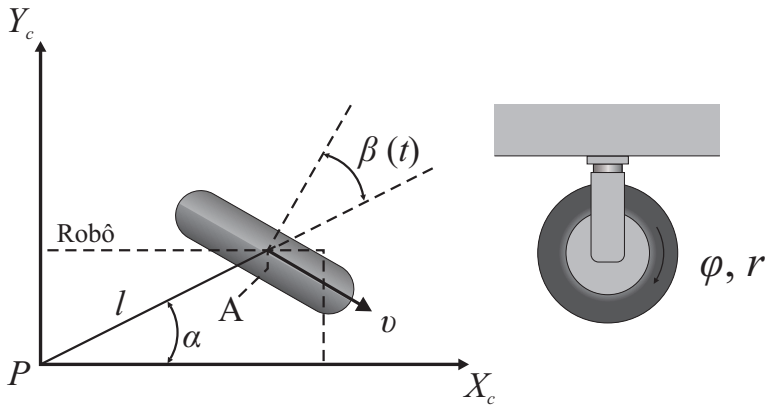


Figura 5: Definição do sistema de coordenadas da roda convencional orientável centrada. Fonte: CAMPION; CHUNG (2008, p. 393)

que é dada por (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998):

$$\begin{bmatrix}\cos(\alpha+\beta) & \sin(\alpha+\beta) & l\sin(\beta)\end{bmatrix} {}^{c}\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o = 0 \tag{5}$$

De forma análoga, os elementos do vetor representam os mapeamentos de cada uma das velocidades no sistema de coordenadas inercial ($\dot{x}_c$, $\dot{y}_c$ e $\dot{\theta}_c$) às suas contribuições para o movimento ortogonal ao plano da roda.

3.1.3 Rodas Convencionais Orientáveis não Centradas

As rodas orientáveis não centradas (Figura 3b), comumente denominadas *caster wheels*, são frequentemente encontradas na parte frontal das cadeiras de rodas motorizadas (Figura 1a). Este tipo de roda também pode ter sua orientação alterada. A rotação do plano da roda, no entanto, ocorre em torno de um eixo vertical que não passa pelo centro da roda. A descrição deste tipo de roda, em relação ao sistema de coordenadas do robô $\{X_c, Y_c, \Theta_c\}$, é apresentada na Figura 6.

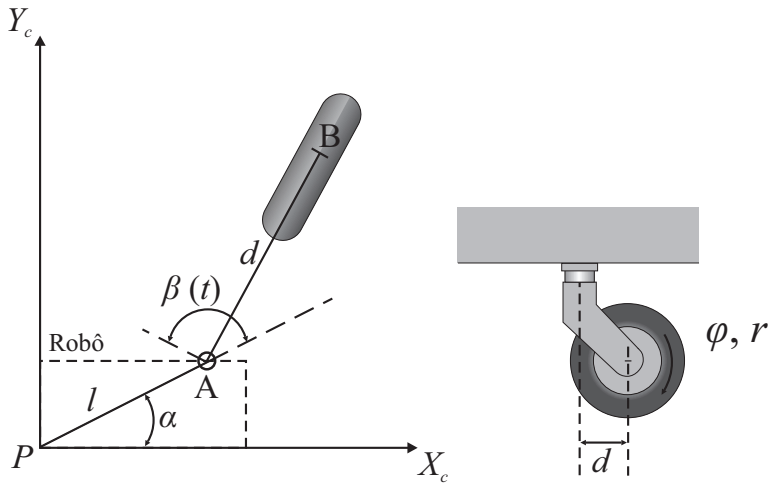


Figura 6: Definição do sistema de coordenadas da roda convencional orientável centrada. Fonte: CAMPION; CHUNG (2008, p. 393)

Pode-se observar que o centro da roda passou a ser denotado por *B*, cuja conexão ao chassis é realizada por meio de uma haste rígida, de comprimento constante *d*, entre os

pontos *A* e *B*, que pode ser rotacionada sobre um eixo fixo vertical que passa por *A*. A posição do ponto *A* em relação ao sistema de coordenadas do robô é equivalente a apresentada nos casos anteriores. Este tipo de roda também impõe duas restrições diferentes ao movimento do robô. A primeira delas, que representa a restrição ao movimento ao longo do plano da roda, é dada por (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998):

$$\begin{bmatrix} -\sin(\alpha+\beta) & \cos(\alpha+\beta) & l\cos(\beta) \end{bmatrix} {}^{c}\boldsymbol{R}_{o}\dot{\boldsymbol{\xi}}_{o} + r\dot{\varphi} = 0 \tag{6}$$

Esta restrição é idêntica à primeira restrição das rodas orientáveis centradas e fixas (4). Por outro lado, a geometria da roda *caster* produz um impacto significativo na restrição relacionada ao movimento no plano ortogonal ao da roda, que é dada por:

$$\begin{bmatrix} \cos(\alpha+\beta) & \sin(\alpha+\beta) & d+l\sin(\beta) \end{bmatrix} {}^{c}\boldsymbol{R}_{o}\dot{\boldsymbol{\xi}}_{o} + d\dot{\beta} = 0 \tag{7}$$

Isto deve-se ao fato de que a força lateral na roda ocorre no ponto *A*, dado que este é o ponto de contato com o chassis. Deste modo, qualquer movimento ortogonal ao plano da roda deve ser balanceado por uma quantidade equivalente e oposta de movimento para sua reorientação (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011, p. 67).

A diferença fundamental entre uma roda orientável centrada e uma não centrada reside no fato de que a roda orientável centrada precisa de algum elemento (como um atuador elétrico) que modifique seu ângulo β, ao passo que a roda não centrada pode posicionar-se naturalmente na direção do movimento (BARROS, 2014, p. 33).

3.1.4 Rodas Universais

As rodas suecas (Figuras 3c-3d) não são muito frequentes em cadeiras de rodas motorizadas, embora já existam veículos comerciais utilizando-as (Figura 1b). A posição do ponto *A* em relação ao sistema de coordenadas do robô é descrita de forma semelhante à roda fixa convencional, conforme apresentado na Figura 7. Porém, é necessário a introdução de mais um parâmetro (Γ) para representar o ângulo entre o eixo dos roletes e o plano da roda principal.

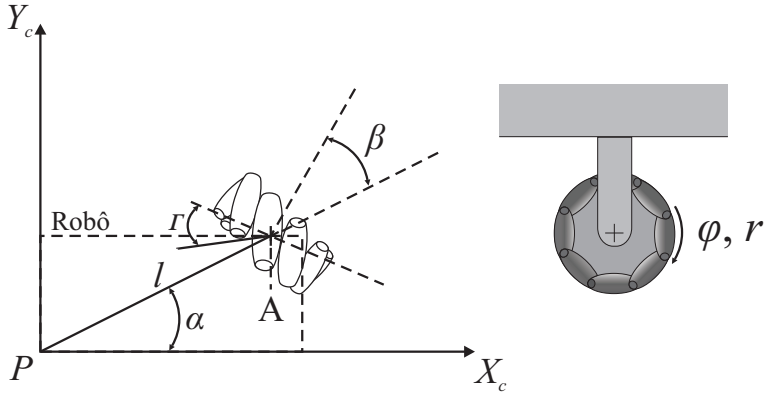


Figura 7: Definição do sistema de coordenadas da roda sueca.
Fonte: SIEGWART; NOURBAKHSH; SCARAMUZZA (2011, p. 69)

Tendo em vista que se trata de uma roda sueca, o componente de velocidade normal é desconhecido, uma vez que a velocidade do ponto de contato nesta direção não é nula.

Com isto, a restrição ao movimento imposta pela roda pode ser escrita conforme (LAGES, 1998, p. 17):

$$\begin{bmatrix} -\sin(\alpha+\beta+\Gamma) & \cos(\alpha+\beta+\Gamma) & l\cos(\beta+\Gamma) \end{bmatrix} {}^{c}\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o + r\dot{\varphi}\cos(\Gamma) = 0 \tag{8}$$

Assume-se que o valor de Γ seja diferente de $\frac{\pi}{2}$ dado que, para $\Gamma = \frac{\pi}{2}$, a roda sujeitar-se-ía a uma restrição idêntica à de uma roda fixa, perdendo a funcionalidade de uma roda sueca.

3.1.5 Características de Mobilidade de Robôs Móveis

Conforme apresentado anteriormente, para cada uma das classes de rodas consideradas, impõe-se uma ou mais restrições ao movimento do robô. As características de mobilidade podem, portanto, ser avaliadas considerando-se a combinação de todas as restrições cinemáticas impostas pelas diferentes rodas utilizadas por um RMR. Nas expressões a seguir serão utilizados os seguintes índices para identificar quantidades relacionadas às subclasses de rodas: f para as rodas convencionais fixas, s para as rodas orientáveis centradas, c para as não centradas e sw para as rodas suecas.

Considerar-se-á um RMR genérico com N rodas das classes descritas acima. O número total de rodas associado ao RMR é definido por $N = N_f + N_s + N_c + N_{sw}$ onde N_f, N_s, N_c e N_{sw} definem a quantidade de rodas de cada subclasse. O conjunto de coordenadas de configuração de um robô móvel é, portanto, descrito em relação ao tempo pelos vetores (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 270):

$$\boldsymbol{\xi}_o(t) = \begin{bmatrix} x_c(t) \\ y_c(t) \\ \theta_c(t) \end{bmatrix} \tag{9}$$

$$\boldsymbol{\beta}(t) = \begin{bmatrix} \boldsymbol{\beta}_s(t) \\ \boldsymbol{\beta}_c(t) \end{bmatrix} \tag{10}$$

$$\boldsymbol{\varphi}(t) = \begin{bmatrix} \varphi_f(t) \\ \varphi_s(t) \\ \varphi_c(t) \\ \varphi_{sw}(t) \end{bmatrix} \tag{11}$$

onde $\boldsymbol{\xi}_o(t)$, $\boldsymbol{\beta}(t)$ e $\boldsymbol{\varphi}(t)$ definem, respectivamente, as coordenadas para a pose no sistema inercial, os ângulos de orientação das rodas orientáveis e os ângulos de rotação das rodas em relação aos respectivos eixos de rotação. As restrições cinemáticas associadas ao RMR podem, então, ser obtidas reescrevendo as expressões (4) a (8) na forma matricial:

$$\boldsymbol{J}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c){}^{c}\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o + \boldsymbol{J}_2\dot{\boldsymbol{\varphi}} = 0 \tag{12}$$

$$\boldsymbol{C}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c){}^{c}\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o + \boldsymbol{C}_2\dot{\boldsymbol{\beta}}_c = 0 \tag{13}$$

onde $\boldsymbol{J}_1$, $\boldsymbol{J}_2$, $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, são matrizes cujas linhas derivam-se das restrições (4) a (8), dadas

por:

$$\boldsymbol{J}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \triangleq \begin{bmatrix} \boldsymbol{J}_{1f} \\ \boldsymbol{J}_{1s}(\boldsymbol{\beta}_s) \\ \boldsymbol{J}_{1c}(\boldsymbol{\beta}_c) \\ \boldsymbol{J}_{1sw} \end{bmatrix} \tag{14}$$

$$\boldsymbol{C}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \triangleq \begin{bmatrix} \boldsymbol{C}_{1f} \\ \boldsymbol{C}_{1s}(\boldsymbol{\beta}_s) \\ \boldsymbol{C}_{1c}(\boldsymbol{\beta}_c) \end{bmatrix} \tag{15}$$

$$\boldsymbol{C}_2 \triangleq \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{C}_{2c} \end{bmatrix} \tag{16}$$

onde $\boldsymbol{J}_{1f}$, $\boldsymbol{J}_{1s}$, $\boldsymbol{J}_{1c}$, $\boldsymbol{J}_{1sw}$ são, respectivamente, matrizes de dimensões $(N_f \times 3)$, $(N_s \times 3)$, $(N_c \times 3)$ e $(N_{sw} \times 3)$, e $\boldsymbol{J}_2$ é uma matriz de constantes $(N \times N)$ cujos elementos da diagonal são os raios das rodas ou, no caso de rodas suecas, o raio das rodas multiplicados por $\cos\Gamma$. Além disto, $\boldsymbol{C}_{1f}$, $\boldsymbol{C}_{1s}$, $\boldsymbol{C}_{1c}$ são matrizes de dimensões $(N_f \times 3)$, $(N_s \times 3)$ e $(N_c \times 3)$, e $\boldsymbol{C}_{2c}$ é uma matriz $(N_c \times N_c)$ cujos elementos da diagonal são iguais as distâncias d das N_c rodas não centradas (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998).

Se consideradas apenas as primeiras $(N_f + N_s)$ restrições de (13), referentes às rodas fixas e orientáveis centradas, tem-se:

$$\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o = 0 \tag{17}$$

onde:

$$\boldsymbol{C}_1^*(\boldsymbol{\beta}_s) \triangleq \begin{bmatrix} \boldsymbol{C}_{1f} \\ \boldsymbol{C}_{1s}(\boldsymbol{\beta}_s) \end{bmatrix} \tag{18}$$

Desta forma o vetor ${}^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o$ pertence ao espaço nulo de $\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)$ e as limitações da mobilidade de um RMR são relacionadas ao *rank* de $\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)$. Conforme CANUDAS DE WIT; SICILIANO; BASTIN (1996, p. 272), a ocorrência de $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = 3$ implicaria em ${}^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o = 0$, o que tornaria impossível a movimentação do RMR no plano. Portanto, é necessário que o *rank* de $\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)$ seja menor ou igual a dois.

Definição 1. Grau de Mobilidade

Define-se **grau de mobilidade** de um robô móvel como:

$$\delta_m = dim\,\mathcal{N}(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = 3 - \rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) \tag{19}$$

Examinando-se o caso em que $\rho(\boldsymbol{C}_{1f}) = 2$, haverá no mínimo duas rodas fixas cujos planos não são paralelos. Caso o RMR possua mais de duas rodas fixas, os eixos de todas elas deverão concorrer para o mesmo centro de rotação instantâneo (ICR, *Instantaneous Center of Rotation*). Uma vez que o ICR é fixo, em decorrência dos planos das rodas não serem paralelos, o único movimento possível ao RMR seria a rotação em torno deste ponto. Esta limitação não é aceitável e, portanto, impõe-se que $\rho(\boldsymbol{C}_{1f}) \leq 1$, ou seja, se o RMR possuir mais de uma roda convencional fixa, todas elas deverão estar em um mesmo eixo. Além disto, tem-se que $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) \leq \rho(\boldsymbol{C}_{1f}) + \rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s))$. Contudo, sob a condição $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = \rho(\boldsymbol{C}_{1f}) + \rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) < 2$ ter-se-ia os centros das rodas

orientáveis centradas sobre o eixo comum das rodas fixas, situação em que as rodas orientáveis centradas perderiam a capacidade de atuar sobre a alocação do ICR. Portanto, assume-se também que $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = \rho(\boldsymbol{C}_{1f}) + \rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) \leq 2$ (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998).

Definição 2. Grau de Dirigibilidade

Define-se **grau de dirigibilidade** de um robô móvel como o número de rodas convencionais orientáveis centradas que podem ser orientadas independentemente para dirigir o robô, dado por:

$$\delta_s = \rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) \tag{20}$$

Caso o RMR seja equipado com mais de δ_s rodas convencionais orientáveis centradas, o movimento das $N_c - \delta_s$ rodas-extra deverá ser coordenado com as demais, garantindo, assim, a existência do ICR a cada instante de tempo.

Deste modo, a partir das condições expostas acima e das definições (19) e (20), pode-se concluir que a configuração das rodas de um RMR deve ser tal que satisfaça as seguintes restrições (LAGES, 1998):

$$1 \leq \delta_m \leq 3 \tag{21}$$

$$0 \leq \delta_s \leq 2 \tag{22}$$

$$2 \leq \delta_m + \delta_s \leq 3 \tag{23}$$

O limite inferior de (21) indica que são considerados apenas os casos em que é possível a movimentação do RMR. A restrição (22), em contrapartida, caracteriza a existência de no máximo duas rodas orientáveis centradas independentes. As configurações em $\delta_m + \delta_s = 1$ implicam que a rotação ocorra somente em torno de um ICR fixo e, portanto, não são consideradas. Por fim, o limite superior de (23) decorre da condição de que $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) \leq 2$ e, na ocorrência de $\delta_s = 2$ o grau de mobilidade (δ_m) será unitário.

A partir do que foi exposto, pode-se concluir que apenas cinco classes de RMR, caracterizadas pelo par (δ_m, δ_s), satisfazem as restrições (21 - 23). Estas classes, consideradas de interesse prático para RMRs, são:

- **Classe** $(3, 0)$ - Os RMRs desta classe apresentam $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = 0$, portanto não possuem rodas fixas ($N_f = 0$) ou orientáveis centradas ($N_s = 0$). Estes robôs são denominados omnidirecionais, pois podem mover-se instantaneamente em qualquer direção sem que haja necessidade de reorientação.

- **Classe** $(2, 0)$ - Os RMRs desta classe apresentam $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = 1$ e $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = 0$, ou seja, possuem pelo menos uma roda fixa. Nos casos em que ($N_f > 1$), as rodas fixas devem estar posicionadas sobre um eixo comum ($\rho(\boldsymbol{C}_{1f}) = 1$). É comum que robôs desta classe operem em modo diferencial.

- **Classe** $(2, 1)$ - Os RMRs desta classe possuem pelo menos uma roda orientável centrada e não apresentam rodas fixas ($\rho(\boldsymbol{C}_{1f}) = 0$). Se houver mais de uma roda orientável centrada ($N_s > 1$), o movimento delas deverá ser coordenado de forma que $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = \delta_s = 1$ seja assegurado.

- **Classe** $(1, 1)$ - Nesta classe os RMRs apresentam $\rho(\boldsymbol{C}_{1f}) = 1$ e, portanto, possuem no mínimo uma roda fixa. Caso haja mais de uma roda fixa ($N_f > 1$), estas devem estar sobre um eixo comum. Além disto, $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = \delta_s = 1$, o que implica

na existência de no mínimo uma roda orientável centrada, cujo centro não deve estar sobre o eixo das rodas fixas. Se houver mais de uma ($N_s > 1$), além da condição anterior, o movimento destas dever ser coordenado de modo a satisfazer $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = 1$.

- **Classe** $(1, 2)$ - Nesta classe, os robôs não possuem rodas fixas, tendo em vista que $\rho(\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)) = \rho(\boldsymbol{C}_{1f}) + \rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = 2$ e $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = \delta_s = 2$. Portanto, devem haver duas ou mais rodas orientáveis centradas, sendo que, para ($N_s > 2$), o movimento destas deve ser coordenado de modo a assegurar $\rho(\boldsymbol{C}_{1s}(\boldsymbol{\beta}_s)) = 2$.

Em geral, os robôs móveis são projetados para que todas as rodas permaneçam em contato com o plano do solo. Neste sentido, apenas três rodas são suficientes para garantir o equilíbrio estático. Caso um RMR possua mais de três rodas, é necessário que haja um sistema de suspensão para que possa movimentar-se em pavimentos irregulares, mantendo todas as rodas em contato com o solo (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011). Deste modo, em classes como a $(2, 0)$, utiliza-se no mínimo uma roda orientável não centrada ou uma roda universal para garantir este equilíbrio.

Em relação ao enquadramento das cadeiras de rodas motorizadas nas classes de robôs citadas, embora existam diversos exemplos utilizando estruturas omnidirecionais, como em MASCARO; SPANO; ASADA (1997), KITAGAWA et al. (2002) e KUNDU et al. (2017), as versões tradicionais possuem acionamento diferencial (LEVINE et al., 1999; CELESTE et al., 2008; FREEDOM, 2018), sendo classificadas conforme a classe $(2, 0)$.

3.1.6 Modelo Cinemático de Pose

A partir de (17) tem-se que ${}^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o$ pertence ao espaço nulo de $\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)$. Portanto pode-se escrever:

$$\dot{\boldsymbol{\xi}}_o = {}^o\boldsymbol{R}_c\Sigma(\boldsymbol{\beta}_s)\boldsymbol{\eta} \tag{24}$$

onde $\Sigma(\boldsymbol{\beta}_s)$ é uma matriz cujas colunas formam uma base do espaço nulo de $\boldsymbol{C}_1^*(\boldsymbol{\beta}_s)$ e a matriz ${}^o\boldsymbol{R}_c$ é dada por:

$${}^o\boldsymbol{R}_c = {}^c\boldsymbol{R}_o^{-1} = {}^c\boldsymbol{R}_o^T = \begin{bmatrix} \cos(\theta_c) & -\sin(\theta_c) & 0 \\ \sin(\theta_c) & \cos(\theta_c) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{25}$$

Ainda, pode-se verificar que as dimensões de $\Sigma(\boldsymbol{\beta}_s)$ e $\boldsymbol{\eta}$ serão sempre $3 \times \delta_m$ e $\delta_m \times 1$ e, definindo-se $\boldsymbol{\zeta} = \dot{\boldsymbol{\beta}}_s$, (24) pode ser aumentada para:

$$\begin{cases} \dot{\boldsymbol{\xi}}_o = {}^o\boldsymbol{R}_c\Sigma(\boldsymbol{\beta}_s)\boldsymbol{\eta} \\ \dot{\boldsymbol{\beta}}_s = \boldsymbol{\zeta} \end{cases} \tag{26}$$

Esta expressão pode ser considerada uma representação do sistema no espaço de estados, denominada modelo cinemático de pose, cujas entradas são $\boldsymbol{\eta}$ e $\boldsymbol{\zeta}$, e as variáveis de estado são as coordenadas de pose $\dot{\boldsymbol{\xi}}_o$ e as coordenadas angulares $\boldsymbol{\beta}_s$ (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 283). Caso o RMR não possua rodas orientáveis centradas ($\boldsymbol{\beta}_s = 0$), a matriz $\Sigma(\boldsymbol{\beta}_s)$ não dependerá de $\boldsymbol{\beta}_s$. Neste caso, o modelo (26) reduz-se a (24).

O modelo cinemático de pose pode ser escrito de forma genérica como:

$$\dot{\boldsymbol{x}} = \boldsymbol{B}(\boldsymbol{x})\boldsymbol{u} \tag{27}$$

onde $\boldsymbol{B}(\boldsymbol{x})$, $\boldsymbol{x}$ e $\boldsymbol{u}$ são definidos por:

$$\boldsymbol{B}(\boldsymbol{x}) \triangleq \begin{cases} {}^{o}\boldsymbol{R}_c\Sigma, & N_s = 0 \\ \begin{bmatrix} {}^{o}\boldsymbol{R}_c\Sigma(\boldsymbol{\beta}_s) & 0 \\ 0 & \boldsymbol{I} \end{bmatrix}, & N_s \geq 0 \end{cases} \tag{28}$$

$$\boldsymbol{x} \triangleq \begin{cases} \boldsymbol{\xi}_o, & N_s = 0 \\ \begin{bmatrix} \boldsymbol{\xi}_o \\ \boldsymbol{\beta}_s \end{bmatrix}, & N_s \geq 0 \end{cases} \tag{29}$$

$$\boldsymbol{u} \triangleq \begin{cases} \boldsymbol{\eta}, & N_s = 0 \\ \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\zeta} \end{bmatrix}, & N_s \geq 0 \end{cases} \tag{30}$$

Nesta seção foi introduzida uma classificação geral para todos os tipos práticos de RMRs, baseada nos graus de mobilidade (δ_m) e dirigibilidade (δ_s). Utilizando-se apenas o número e a configuração das rodas fixas (N_f) e orientáveis centradas (N_s), é possível determinar a qual classe particular um robô pertence. É importante ressaltar que para qualquer RMR particular é sempre possível selecionar a origem e a orientação do sistema de coordenadas do robô de modo que o modelo cinemático de postura do RMR assuma uma forma genérica, que é única para cada uma das classes (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 283). Em outras palavras, todos os robôs pertencentes a uma determinada classe podem ser descritos pelo mesmo modelo cinemático de postura.

Em CANUDAS DE WIT; SICILIANO; BASTIN (1996) são apresentadas maneiras de definir a origem e a orientação do sistemas de coordenadas do robô. São, também, apresentadas modelos de pose particularizados para cada uma das cinco classes, cujo resumo encontra-se na Tabela 1.

Definição 3. Grau de Manobrabilidade

Define-se **grau de manobrabilidade** de um robô móvel como o número de graus de liberdade que podem ser manipulados pelo RMR por meio das entradas $\boldsymbol{\eta}$ e $\boldsymbol{\zeta}$, dado por:

$$\delta_M = \delta_m + \delta_s \tag{31}$$

Embora o grau de mobilidade (δ_m) represente o número de graus de liberdade (GDL) que podem ser manipulados diretamente pelas entradas $\boldsymbol{\eta}$, sem a reorientação das rodas orientáveis centradas, esta quantidade não representa todos os GDL que um robô móvel pode manipular. Deve-se acrescentar a este número os δ_s GDL adicionais, acessíveis por meio da entrada $\boldsymbol{\zeta}$.

É importante salientar que a ação de $\boldsymbol{\zeta}$ sobre as coordenadas de pose $\boldsymbol{\xi}_o$ é indireta, uma vez que é obtida somente pelas coordenadas $\boldsymbol{\beta}_s$, que são relacionadas a $\boldsymbol{\zeta}$ por uma ação integral. Fisicamente, isto pode ser verificado considerando-se que $\boldsymbol{\zeta}$ modifica somente a orientação das rodas centradas e, portanto, influenciará as coordenadas de postura apenas se houver movimento (CANUDAS DE WIT; SICILIANO; BASTIN, 1996; LAGES, 1998).

É interessante observar que dois RMRs com os mesmos valores de δ_m, mas diferentes δ_M, não são equivalentes, dado que o robô com o maior δ_M é mais manobrável. No entanto, para um mesmo valor de δ_M, o robô com maior δ_m é mais manobrável. Portanto, a situação ideal é alcançada pela classe $(3, 0)$, que apresenta $\delta_M = \delta_m = 3$.

Tabela 1: Modelos Cinemáticos de Postura

Classe	Modelo
$(3,0)$ Omnidirecional	$\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}_c\end{bmatrix} = \begin{bmatrix}\cos(\theta_c) & -\sin(\theta_c) & 0\\ \sin(\theta_c) & \cos(\theta_c) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}$
$(2,0)$ Diferencial	$\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}_c\end{bmatrix} = \begin{bmatrix}\cos(\theta_c) & 0\\ \sin(\theta_c) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}$
$(2,1)$	$\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}_c\\ \dot{\beta}_s\end{bmatrix} = \begin{bmatrix}\cos(\theta_c+\boldsymbol{\beta}_s) & 0 & 0\\ \sin(\theta_c+\boldsymbol{\beta}_s) & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\eta_1\\ \eta_2\\ \zeta\end{bmatrix}$
$(1,1)$ Carro	$\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}_c\\ \dot{\beta}_s\end{bmatrix} = \begin{bmatrix}d\cos(\theta_c)\sin(\boldsymbol{\beta}_s) & 0\\ d\sin(\theta_c)\cos(\boldsymbol{\beta}_s) & 0\\ \cos(\boldsymbol{\beta}_s) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\eta_1\\ \zeta\end{bmatrix}$
$(1,2)$	$\begin{bmatrix}\dot{x}_c\\ \dot{y}_c\\ \dot{\theta}_c\\ \dot{\beta}_{s1}\\ \dot{\beta}_{s2}\end{bmatrix} = \begin{bmatrix}L[\sin(\beta_{s1})\cos(\theta_c+\beta_{s2})+\sin(\beta_{s2})\cos(\theta_c+\beta_{s1})] & 0 & 0\\ L[\sin(\beta_{s1})\sin(\theta_c+\beta_{s2})+\sin(\beta_{s2})\sin(\theta_c+\beta_{s1})] & 0 & 0\\ \sin(\beta_{s2}-\beta_{s1}) & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\eta_1\\ \zeta_1\\ \zeta_2\end{bmatrix}$

Isto fica evidente ao observar-se que o ICR de robôs da classe $(3,0)$ pode ser livremente alocado no plano, diretamente a partir das entradas $\boldsymbol{\eta}$, ao passo que para as classes $(1,2)$ e $(2,1)$ este comportamento é alcançando por intermédio da reorientação das rodas centradas. Contudo, este não é o caso das classes $(1,1)$ e $(2,0)$, que possuem $\delta_M = 2$. Nestas últimas duas, a posição do ICR está restrita a pertencer à linha que passa sobre o eixo das rodas fixas. Esta posição será determinada diretamente por $\boldsymbol{\eta}$ para robôs da classe $(2,0)$ ou a partir da orientação das rodas centradas para a classe $(1,1)$ (LAGES, 1998).

3.1.7 Modelo Cinemático de Configuração

Para obter o modelo cinemático de pose, utilizou-se apenas um subconjunto de (12) e (13), que correspondem às restrições impostas pelas rodas fixas e orientáveis centradas. Utilizando-se as demais restrições, pode-se obter expressões para as velocidades angulares e rotacionais que não foram consideradas no modelo anterior.

A partir de (12) e (13) obtém-se:

$$\dot{\boldsymbol{\beta}}_c = -\boldsymbol{C}_{2c}^{-1}\boldsymbol{C}_{1c}(\boldsymbol{\beta}_c)^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o \quad (32)$$

$$\dot{\boldsymbol{\varphi}} = -\boldsymbol{J}_2^{-1}\boldsymbol{J}_1(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)^c\boldsymbol{R}_o\dot{\boldsymbol{\xi}}_o \quad (33)$$

Combinando-se estas equações com $\dot{\boldsymbol{\xi}}_o$ de (24), as equações para $\dot{\boldsymbol{\beta}}_c$ e $\dot{\boldsymbol{\varphi}}$ tornam-se:

$$\dot{\boldsymbol{\beta}}_c = \boldsymbol{D}(\boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s)\boldsymbol{\eta} \tag{34}$$

$$\dot{\boldsymbol{\varphi}} = \boldsymbol{E}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s)\boldsymbol{\eta} \tag{35}$$

onde $\boldsymbol{D}(\boldsymbol{\beta}_c)$ e $\boldsymbol{E}(\boldsymbol{\beta})$ são definidos por:

$$\boldsymbol{D}(\boldsymbol{\beta}_c) \triangleq -\boldsymbol{C}_{2c}^{-1}\boldsymbol{C}_{1c}(\boldsymbol{\beta}_c) \tag{36}$$

$$\boldsymbol{E}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \triangleq -\boldsymbol{J}_2^{-1}\boldsymbol{J}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \tag{37}$$

Definindo-se $\boldsymbol{q}$ como o vetor de coordenadas de configuração:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{\xi}_o \\ \boldsymbol{\beta}_s \\ \boldsymbol{\beta}_c \\ \boldsymbol{\varphi} \end{bmatrix} \tag{38}$$

a evolução das coordenadas de configuração podem ser descritas pela seguinte forma compacta, resultante de (26), (34) e (35), denominada modelo cinemático de configuração (CAMPION; BASTIN; DANDREA-NOVEL, 1996):

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{u} \tag{39}$$

com:

$$\boldsymbol{S}(\boldsymbol{q}) = \begin{bmatrix} {}^o\boldsymbol{R}_c\Sigma(\boldsymbol{\beta}_s) & 0 \\ 0 & \boldsymbol{I} \\ \boldsymbol{D}(\boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s) & 0 \\ \boldsymbol{E}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s) & 0 \end{bmatrix} \tag{40}$$

$$\boldsymbol{u} \triangleq \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\zeta} \end{bmatrix} \tag{41}$$

As restrições (12), (13) e (17) podem ser representadas de forma compacta por:

$$\boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = 0 \tag{42}$$

com $\boldsymbol{J}(\boldsymbol{q})$ representando o Jacobiano das restrições, definido por:

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{J}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c){}^c\boldsymbol{R}_o & 0 & 0 & \boldsymbol{J}_2 \\ \boldsymbol{C}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c){}^c\boldsymbol{R}_o & 0 & \boldsymbol{C}_2 & 0 \\ \boldsymbol{C}_1^*(\boldsymbol{\beta}_s){}^c\boldsymbol{R}_o & 0 & 0 & 0 \end{bmatrix} \tag{43}$$

3.1.8 Modelo Dinâmico de Configuração

Os modelos apresentados nas subseções anteriores descrevem o comportamento do robô em função das velocidades das rodas e baseiam-se somente nas características geométricas deste. No entanto, fisicamente, as variáveis de entrada de um robô móvel são as forças generalizadas aplicadas pelos atuadores, que não são diretamente contempladas por estas descrições.

Nesta seção será derivado o modelo dinâmico de configuração, que permite relacionar os torques fornecidos pelos atuadores com a evolução do vetor de coordenadas generalizadas ($\boldsymbol{q}$) e constitui, do ponto de vista mecânico, a descrição completa do sistema (CAMPION; CHUNG, 2008).

Para o desenvolvimento deste modelo, além das considerações realizadas para a cinemática, assume-se que o robô seja equipado com atuadores capazes de forçar a rotação das rodas (coordenadas de rotação $\boldsymbol{\varphi}$) e a orientação das rodas orientáveis centradas e orientáveis não centradas (coordenadas angulares $\boldsymbol{\beta}_s$ e $\boldsymbol{\beta}_c$). Os torques aplicados para rotação das rodas e para orientação das rodas orientáveis centradas e não centradas são denotados por $\boldsymbol{\tau}_\varphi$, $\boldsymbol{\tau}_s$ e $\boldsymbol{\tau}_c$, respectivamente.

Utilizando-se o formalismo de Lagrange, tem-se que a dinâmica de um RMR é descrita pelas seguintes $(3 + N_c + N + N_s)$ equações (CANUDAS DE WIT; SICILIANO; BASTIN, 1996):

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial\,\dot{\boldsymbol{\xi}}}\right) - \frac{\partial L}{\partial \boldsymbol{\xi}}\right)^T = {}^o\boldsymbol{R}_c\,\boldsymbol{J}_1^T(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)\mu_1 + {}^o\boldsymbol{R}_c\,\boldsymbol{C}_1^T(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)\mu_2 \tag{44}$$

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{\beta}}_s}\right) - \frac{\partial L}{\partial \boldsymbol{\beta}_s}\right)^T = \boldsymbol{\tau}_s \tag{45}$$

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{\beta}}_c}\right) - \frac{\partial L}{\partial \boldsymbol{\beta}_c}\right)^T = \boldsymbol{C}_2^T\mu_2 + \boldsymbol{\tau}_c \tag{46}$$

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{\varphi}}}\right) - \frac{\partial L}{\partial \boldsymbol{\varphi}}\right)^T = \boldsymbol{J}_2^T\mu_1 + \boldsymbol{\tau}_\varphi \tag{47}$$

onde μ_1 é o coeficiente de Lagrange associado à restrição (12), μ_2 o coeficiente de Lagrange associado à restrição (13) e L é a função de Lagrange definida por:

$$L = K - P \tag{48}$$

para a qual K e P representam, respectivamente, as energias cinética e potencial. Tendo em vista que o RMR move-se exclusivamente sobre um plano horizontal, não há variação da energia potencial (P) e, portanto, a equação (48) resume-se a $L = K$ nas expressões (44) a (47), que podem ser representadas de forma compacta por (CAMPION; CHUNG, 2008):

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial L}{\partial \boldsymbol{q}}\right)^T = \boldsymbol{\tau} + \boldsymbol{J}^T(q)\boldsymbol{\mu} \tag{49}$$

onde $\boldsymbol{\tau} = \begin{bmatrix}0 & \boldsymbol{\tau}_s^T & \boldsymbol{\tau}_c^T & \boldsymbol{\tau}_\varphi^T\end{bmatrix}^T$ é o vetor das forças generalizadas associadas aos torques aplicados pelos atuadores, o termo $\boldsymbol{J}^T(q)\boldsymbol{\mu}$ caracteriza o vetor de forças generalizadas associadas às restrições cinemáticas e $\boldsymbol{\mu} = \begin{bmatrix}\mu_1 & \mu_2 & 0\end{bmatrix}$ é o vetor dos multiplicadores de Lagrange associados às restrições cinemáticas.

Procede-se em seguida com a eliminação dos multiplicadores de Lagrange. Inicialmente, pré-multiplicam-se as expressões (44), (46) e (47), respectivamente, por ${}^c\boldsymbol{R}_o$, $\boldsymbol{D}^T(\boldsymbol{\beta}_c)$ e $\boldsymbol{E}^T(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)$ e, realizando-se a soma dos resultados, obtém-se:

$$\begin{aligned}{}^c\boldsymbol{R}_o\left(\frac{d}{dt}\left(\frac{\partial T}{\partial\,\dot{\boldsymbol{\xi}}}\right) - \frac{\partial T}{\partial \boldsymbol{\xi}}\right)^T + \boldsymbol{D}^T(\boldsymbol{\beta}_c)\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{\beta}}_c}\right) - \frac{\partial L}{\partial \boldsymbol{\beta}_c}\right)^T + \\ \boldsymbol{E}^T(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)\left(\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{\varphi}}}\right) - \frac{\partial L}{\partial \boldsymbol{\varphi}}\right)^T = \left(\boldsymbol{J}_1(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c) + \boldsymbol{J}_2\boldsymbol{E}(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)\right)^T\mu_1 + \\ \left(\boldsymbol{C}_1(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c) + \boldsymbol{C}_2\boldsymbol{D}(\boldsymbol{\beta_c})\right)^T\mu_2 + \boldsymbol{D}^T(\boldsymbol{\beta}_c)\boldsymbol{\tau}_c + \boldsymbol{E}^T(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)\boldsymbol{\tau}_\varphi\end{aligned} \tag{50}$$

Tendo em vista que $\boldsymbol{D}(\boldsymbol{\beta}_c)$ e $\boldsymbol{E}(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c)$ satisfazem as seguintes equações:

$$\boldsymbol{J}_1(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c) + \boldsymbol{J}_2\boldsymbol{E}(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c) = 0 \tag{51}$$

$$\boldsymbol{C}_1(\boldsymbol{\beta}_s,\boldsymbol{\beta}_c) + \boldsymbol{C}_2\boldsymbol{D}(\boldsymbol{\beta_c}) = 0 \tag{52}$$

A aplicação de (51) e (52) em (50) leva à eliminação dos multiplicadores de Lagrange μ_1 e μ_2, resultando em:

$$
\begin{aligned}
&{}^{c}\boldsymbol{R}_o \left(\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{\boldsymbol{\xi}}} \right) - \frac{\partial T}{\partial \boldsymbol{\xi}} \right)^T + \boldsymbol{D}^T(\boldsymbol{\beta}_c) \left(\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\boldsymbol{\beta}}_c} \right) - \frac{\partial L}{\partial \boldsymbol{\beta}_c} \right)^T + \\
&\boldsymbol{E}^T(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \left(\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\boldsymbol{\varphi}}} \right) - \frac{\partial L}{\partial \boldsymbol{\varphi}} \right)^T = \boldsymbol{D}^T(\boldsymbol{\beta}_c)\boldsymbol{\tau}_c + \boldsymbol{E}^T(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{\tau}_\varphi
\end{aligned} \tag{53}
$$

Além disto, a energia cinética (K) de um RMR é dada por (CANUDAS DE WIT; SICILIANO; BASTIN, 1996, p. 295):

$$
\begin{aligned}
K = \quad & \frac{1}{2} \dot{\boldsymbol{\xi}}_o^{\,T} {}^{c}\boldsymbol{R}_o^T \left[\boldsymbol{M}(\boldsymbol{\beta}_c) {}^{c}\boldsymbol{R}_o \dot{\boldsymbol{\xi}}_o + 2\boldsymbol{V}(\boldsymbol{\beta}_c)\dot{\boldsymbol{\beta}}_c + 2\boldsymbol{W}\dot{\boldsymbol{\beta}}_s \right] + \\
& \frac{1}{2} \dot{\boldsymbol{\beta}}_c^{\,T} \boldsymbol{I}_c \dot{\boldsymbol{\beta}}_c + \frac{1}{2} \dot{\boldsymbol{\varphi}}^T \boldsymbol{I}_\varphi \dot{\boldsymbol{\varphi}} + \frac{1}{2} \dot{\boldsymbol{\beta}}_s^{\,T} \boldsymbol{I}_s \dot{\boldsymbol{\beta}}_s
\end{aligned} \tag{54}
$$

onde $\boldsymbol{M}(\boldsymbol{\beta}_c)$, $\boldsymbol{V}(\boldsymbol{\beta}_c)$, $\boldsymbol{W}$, $\boldsymbol{I}_c$, $\boldsymbol{I}_{\boldsymbol{\varphi}}$ e $\boldsymbol{I}_s$ são funções dos parâmetros de massa e inércia dos vários corpos rígidos que constituem o robô. Assim, substituindo-se (54) em (53) e eliminando-se as velocidades $\dot{\boldsymbol{\xi}}_o$, $\dot{\boldsymbol{\beta}}_s$, $\dot{\boldsymbol{\beta}}_c$ e $\dot{\boldsymbol{\varphi}}$ e as acelerações $\ddot{\boldsymbol{\xi}}_o$, $\ddot{\boldsymbol{\beta}}_s$, $\ddot{\boldsymbol{\beta}}_c$ e $\ddot{\boldsymbol{\varphi}}$ com a utilização de (26), (34) e (35) e suas derivadas, obtém-se:

$$
\begin{aligned}
&\boldsymbol{H}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\dot{\boldsymbol{\eta}} + \Sigma^T(\boldsymbol{\beta}_s)\boldsymbol{V}(\boldsymbol{\beta}_c)\dot{\boldsymbol{\zeta}} + \boldsymbol{f}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c, \boldsymbol{\eta}, \boldsymbol{\zeta}) \\
&\qquad = \Sigma^T(\boldsymbol{\beta}_s)[\boldsymbol{D}^T(\boldsymbol{\beta}_c)\boldsymbol{\tau}_c + \boldsymbol{E}^T(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{\tau}_\varphi]
\end{aligned} \tag{55}
$$

e

$$
\boldsymbol{V}^T(\boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s)\dot{\boldsymbol{\eta}} + \boldsymbol{I}_s\dot{\boldsymbol{\zeta}} + \boldsymbol{f}_2(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c, \boldsymbol{\eta}, \boldsymbol{\zeta}) = \boldsymbol{\tau}_s \tag{56}
$$

onde:

$$
\begin{aligned}
\boldsymbol{H}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) \triangleq \quad & \Sigma^T(\boldsymbol{\beta}_s)[\boldsymbol{M}(\boldsymbol{\beta}_c) + \boldsymbol{D}^T(\boldsymbol{\beta}_c)\boldsymbol{V}^T(\boldsymbol{\beta}_c) + \boldsymbol{V}(\boldsymbol{\beta}_c)\boldsymbol{D}(\boldsymbol{\beta}_c) + \\
& \boldsymbol{D}^T(\boldsymbol{\beta}_c)\boldsymbol{I}_{\boldsymbol{\beta}}\boldsymbol{D}(\boldsymbol{\beta}_c) + \boldsymbol{E}^T(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{I}_{\boldsymbol{\varphi}}\boldsymbol{E}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)]\Sigma(\boldsymbol{\beta}_s)
\end{aligned} \tag{57}
$$

Deste modo, o modelo dinâmico de configuração de um RMR é formado pelas equações (26), (34), (35), (55) e (56). Neste modelo genérico, $\boldsymbol{\tau}_s$, $\boldsymbol{\tau}_c$ e $\boldsymbol{\tau}_\varphi$ representam os torques que podem potencialmente ser aplicados para orientação e rotação das rodas. Na prática, no entanto, apenas alguns destes torques são aplicados, pois geralmente será utilizado o número mínimo necessário de atuadores.

É evidente que cada roda orientável centrada deve necessariamente possuir um atuador para sua orientação, pois, caso contrário, comportar-se-ia como uma roda fixa. Em compensação, os vetores $\boldsymbol{\tau}_c$ e $\boldsymbol{\tau}_\varphi$ podem apresentar componentes identicamente nulos, desde que a rotação e a orientação das rodas as quais estes estejam associados possam ser obtidas através do acionamento das demais rodas.

Portanto, para garantir a total mobilidade de um RMR, além dos torques para a orientação das rodas orientáveis centradas, quando houver este tipo de rodas no robô, devem ser utilizados N_m atuadores adicionais ($N_m \geq \delta_m$) para ou forçar a rotação de algumas das rodas ou forçar a orientação de algumas das rodas orientáveis não centradas. Tem-se então que o vetor de torques desenvolvidos por estes atuadores $\boldsymbol{\tau}_m$ pode ser obtido de:

$$
\begin{bmatrix} \boldsymbol{\tau}_c \\ \boldsymbol{\tau}_\varphi \end{bmatrix} = \boldsymbol{P}\boldsymbol{\tau}_m \tag{58}
$$

onde $\boldsymbol{P}$ é uma matriz $(N_c + N) \times N_m$ que seleciona os componentes de $\begin{bmatrix}\boldsymbol{\tau}_c & \boldsymbol{\tau}_\varphi\end{bmatrix}^T$ que são efetivamente utilizados como entradas de controle. Deste modo, definindo-se $\boldsymbol{B}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c) = \Sigma^T(\boldsymbol{\beta}_s)\begin{bmatrix}\boldsymbol{D}^T(\boldsymbol{\beta}_c) & \boldsymbol{E}^T(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\end{bmatrix}$, a expressão (55) pode ser reescrita como:

$$\boldsymbol{H}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\dot{\boldsymbol{\eta}} + \Sigma^T(\boldsymbol{\beta}_s)\boldsymbol{V}(\boldsymbol{\beta}_c)\dot{\boldsymbol{\zeta}} + \boldsymbol{f}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c, \boldsymbol{\eta}, \boldsymbol{\zeta}) = \boldsymbol{B}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{P}\boldsymbol{\tau}_m \tag{59}$$

É imprescindível que a matriz $\boldsymbol{B}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{P}$ possua *rank* completo para quaisquer valores de $\boldsymbol{\beta}_s$ e $\boldsymbol{\beta}_c$, pois de outra forma existirão valores de $\boldsymbol{\beta}$ para os quais o robô tornar-se-á sub-atuado. Isto se justifica pelo fato de que se o *rank* de $\boldsymbol{B}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{P}$ não for completo, não haverá graus de liberdade suficientes nas entradas de controle para determinar a alocação do ICR do RMR (LAGES, 1998, p. 28). Desta condição resulta o número mínimo de atuadores que devem ser utilizados para uma determinada classe de RMR, conforme explicitado na tabela 2.

Tabela 2: Número mínimo de atuadores para cada classe de robô móvel

Classe	(3,0)	(2,0)	(2,1)	(1,1)	(1,2)
Atuadores	3[1] ou 4[2]	2	3	2	4

O modelo dinâmico de configuração de um RMR pode ser reescrito na forma compacta:

$$\begin{cases} \dot{\boldsymbol{q}} &= \boldsymbol{S}(\boldsymbol{q})\boldsymbol{u} \\ \boldsymbol{H}(\boldsymbol{\beta})\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u}) &= \boldsymbol{F}(\boldsymbol{\beta})\boldsymbol{\tau}_0 \end{cases} \tag{60}$$

utilizando-se as seguintes definições:

$$\boldsymbol{\beta} \triangleq \begin{bmatrix}\boldsymbol{\beta}_s \\ \boldsymbol{\beta}_c\end{bmatrix} \tag{61}$$

$$\boldsymbol{q} \triangleq \begin{bmatrix}\boldsymbol{\xi}_o \\ \boldsymbol{\beta} \\ \boldsymbol{\varphi}\end{bmatrix} \tag{62}$$

$$\boldsymbol{u} \triangleq \begin{bmatrix}\boldsymbol{\eta} \\ \boldsymbol{\zeta}\end{bmatrix} \tag{63}$$

$$\boldsymbol{H}(\boldsymbol{\beta}) \triangleq \begin{bmatrix}\boldsymbol{H}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\dot{\boldsymbol{\eta}} & \Sigma^T(\boldsymbol{\beta}_s)\boldsymbol{V}(\boldsymbol{\beta}_c) \\ \boldsymbol{V}^T(\boldsymbol{\beta}_c)\Sigma(\boldsymbol{\beta}_s) & \boldsymbol{I}_s\end{bmatrix} \tag{64}$$

$$\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u}) \triangleq \begin{bmatrix}\boldsymbol{f}_1(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c, \boldsymbol{\eta}, \boldsymbol{\zeta}) \\ \boldsymbol{f}_2(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c, \boldsymbol{\eta}, \boldsymbol{\zeta})\end{bmatrix} \tag{65}$$

$$\boldsymbol{F}(\boldsymbol{\beta}) \triangleq \begin{bmatrix}\boldsymbol{B}(\boldsymbol{\beta}_s, \boldsymbol{\beta}_c)\boldsymbol{P} & 0 \\ 0 & \boldsymbol{I}\end{bmatrix} \tag{66}$$

$$\boldsymbol{\tau}_o \triangleq \begin{bmatrix}\boldsymbol{\tau}_m \\ \boldsymbol{\tau}_s\end{bmatrix} \tag{67}$$

3.1.9 Modelo Dinâmico de Pose

O modelo dinâmico de configuração de um RMR pode ser simplificado para:

$$\begin{cases} \dot{\boldsymbol{q}} &= \boldsymbol{S}(\boldsymbol{q})\boldsymbol{u} \\ \dot{\boldsymbol{u}} &= \boldsymbol{\nu} \end{cases} \tag{68}$$

[1] utilizando rodas suecas

[2] utilizando rodas convencionais

utilizando-se a seguinte realimentação de estados:

$$\boldsymbol{\tau}_0 = \boldsymbol{F}^{\dagger}(\boldsymbol{\beta})[\boldsymbol{H}(\boldsymbol{\beta})\boldsymbol{\nu} + \boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u})] \tag{69}$$

onde $\boldsymbol{\nu}$ representa um conjunto de δ_m entradas de controle auxiliares e $\boldsymbol{F}^{\dagger}(\boldsymbol{\beta})$ denota uma inversa à esquerda de $\boldsymbol{F}(\boldsymbol{\beta})$.

Além disto, para aplicações de controle em geral, o interesse encontra-se basicamente nas coordenadas de pose do RMR ($\boldsymbol{\xi}_o$). Neste sentido, os valores das variáveis internas $\boldsymbol{\beta}_c$ e $\boldsymbol{\varphi}$ não são de interesse prático e, portanto, podem ser ignorados. Desta forma, é obtido o modelo dinâmico de pose, definido por (LAGES, 1998):

$$\begin{cases} \dot{\boldsymbol{x}} &= \boldsymbol{B}(\boldsymbol{x})\boldsymbol{u} \\ \dot{\boldsymbol{u}} &= \boldsymbol{\nu} \end{cases} \tag{70}$$

para o qual $\boldsymbol{x}$ e $\boldsymbol{u}$ mantém-se definidos conforme (29) e (30).

É importante salientar que este modelo descreve totalmente a dinâmica entre as coordenadas de pose ($\boldsymbol{\xi}_o$) e as entradas de controle ($\boldsymbol{\nu}$). Embora as coordenadas $\boldsymbol{\beta}_c$ e $\boldsymbol{\varphi}$, tenham sido suprimidas do modelo, estas permanecem implicitamente na realimentação definida pela expressão (69). Por fim, o modelo dinâmico de pose herda as propriedades estruturais do modelo cinemático de pose e, portanto, também é genérico, uma vez que é válido para qualquer classe possível de RMR.

3.2 Estimação de Pose por *Dead-Reckoning*

Várias técnicas têm sido apresentadas na literatura para a obtenção da pose de robôs móveis utilizando-se *encoder* (BORENSTEIN; KOREN, 1991a), visão computacional (WENG; COHEN; REBIBO, 1992; BAUMGARTNER; SKAAR, 1994; GEIGER; ZIEGLER; STILLER, 2011; SCARAMUZZA; FRAUNDORFER, 2011; ZHANG; SINGH, 2015; AQEL et al., 2016), sensores inerciais (CHEN et al., 2017; ALATISE; HANCKE, 2017), sonar (KUC; SIEGEL, 1987; BOZMA; KUC, 1994) e *laser* (ZHANG; SINGH, 2015; BOK; CHOI; KWEON, 2013). Em BORENSTEIN; EVERETT; FENG (1996) é apresentada uma discussão ampla sobre os principais sensores e métodos empregados na estimação de pose.

O método de *dead-reckoning* é derivado do termo *deduced reckoning*, usado na aviação, que consiste em determinar a posição da aeronave, usando uma bússola, a velocidade e o tempo de deslocamento (ROMERO et al., 2014). A forma mais simples de implementação deste método é denominada odometria, e consiste em calcular-se os deslocamentos linear e angular do robô, a partir dos deslocamentos das rodas, obtidos por *encoders*[3] incrementais.

A aplicação do *dead-reckoning*, entretanto, não é apropriada para estimação de pose de robôs móveis que desenvolvem trajetórias longas. Isto deve-se ao fato de que, pela própria natureza do método, os erros vão sendo acumulados ao integrar-se os deslocamentos (LAGES, 1998, p. 110). Os erros mais significativos são os sistemáticos, caracterizados pelas incertezas nos parâmetros geométricos utilizados para calcular a pose do robô. Contudo, estes erros podem ser reduzidos em até duas ordens de grandeza por meio de procedimentos adequados de calibração (BORENSTEIN; FENG, 1994). Já os erros não-sistemáticos (ou aleatórios) ocorrem devido a escorregamento das rodas, imperfeições no solo e fenômenos similares.

[3]Uma visão mais ampla sobre o uso de *encoders* e odometria pode ser obtida em BORENSTEIN; EVERETT; FENG (1996), ROMERO et al. (2014) e TZAFESTAS (2013).

Os métodos que utilizam visão computacional baseiam-se, geralmente, na detecção de *landmarks* existentes no ambiente, que podem ser naturais (LUO; LIN; CHI LAI, 2008; GUIZILINI, 2008) ou artificiais (MURATA; HIROSE, 1993; ZHONG; ZHOU; LIU, 2017), ou na estimação de movimento a partir de sequências de imagens (NEGAHDARIPOUR; HAYASHI; ALOIMONOS, 1995; SOATTO; FREZZA; PERONA, 1996). Possuem como desvantagens a sensibilidade à variação de luminosidade, e o tempo requerido para o processamento da imagem, o que dificulta a sua implementação em tempo real (LAGES, 1998).

Outro método baseado em visão computacional é a odometria visual (HUANG et al., 2017; GEIGER; ZIEGLER; STILLER, 2011; SCARAMUZZA; FRAUNDORFER, 2011) que é realizada estimando-se, incrementalmente, a pose do veículo a partir da análise das mudanças que seu movimento causa nas imagens capturadas, por cada uma de suas câmeras. Este tipo de método pode trazer vantagens em relação a odometria baseada em *encoders*. Em terrenos irregulares, por exemplo, a odometria visual não é afetada pelo escorregamento. Contudo, à semelhança da odometria por *encoders*, o erro de estimação acumula-se a cada passo de integração.

Conforme LAGES (1998, p. 111) dados provenientes de sonares de ultra-som podem ser processados em um tempo menor, mas possuem menor precisão em relação à orientação. Neste tipo de abordagem podem surgir problemas em decorrência dos efeitos de reflexão, difração e ângulo de abertura a que estão sujeitas as ondas sonoras (LEONARD; DURRANT-WHYTE, 1992). Em geral, os métodos que utilizam estes sensores baseiam-se no casamento de um mapa de ultra-som, armazenado na memória do robô, com a resposta obtida pelos sensores.

Uma alternativa para evitar as dificuldades inerentes ao uso destes sensores, é a utilização de medidas de distância obtidas a partir de um *scanner* a *laser* (RÖFER, 2001; ZHANG; SINGH, 2015; CHEN et al., 2017). Segundo LAGES (1998, p. 112), os métodos utilizados com sonares podem ser aplicados aos sensores *laser*, tendo-se a vantagem de que os efeitos de reflexão, difração e ângulo de abertura do feixe são minimizados.

O uso de unidades de medição inerciais (*inertial measurement units*), de baixo custo, também tem sido proposto para a estimação de pose de RMRs (BARSHAN; DURRANT-WHYTE, 1995; ALATISE; HANCKE, 2017). As unidades de medição inerciais são, geralmente, compostas por um giroscópio e um acelerômetro, ambos com medições em três eixos. A partir do acelerômetro, são obtidas as acelerações lineares e, a partir do giroscópio, as velocidades angulares. Deste modo, integrando-se duas vezes as medidas do acelerômetro, são determinados os deslocamentos relativos (DUDEK; JENKIN, 2008). Por outro lado, integrando-se as medidas do giroscópio, são obtidas medidas relativas de orientação. A utilização deste tipo de sensor pode, portanto, ser considerada como uma aplicação de *dead-reckoning*.

Contudo, antes de integrar as medidas de aceleração, devem-se estimar e extrair, baseado na estimativa de orientação do veículo, a resultante do vetor gravidade em cada uma delas. Em virtude disto, o fenômeno de *drift* nas medições do giroscópio acaba afetando o cancelamento do vetor de gravidade nas medidas do acelerômetro e, tendo em vista que a aceleração é integrada duas vezes, qualquer resíduo deste vetor leva a um erro quadrático na estimativa de posição. Consequentemente, o uso deste tipo de sensor requer uma modelagem cuidadosa dos erros.

Os sistemas *Global Positioning System* (GPS) representam o mecanismo mais comum para estimar a localização de um veículo. Este tipo de sistema fornece a posição absoluta em três dimensões, bem como a data e a hora, e encontra-se disponível em todos os lugares

do globo, embora não possam ser operados em ambientes internos. As medidas do sistema de GPS convencional possuem precisão no plano horizontal de aproximadamente vinte metros, mas que pode ser razoavelmente minimizada operando-se em modo diferencial, tornando-se uma alternativa apropriada para uso em ambientes externos (LAGES, 1998; DUDEK; JENKIN, 2008).

Evidentemente existem várias outras abordagens, mas mencionar todas elas aqui extrapolaria o escopo deste trabalho. No entanto, não há uma solução definitiva e única para o problema de estimação de pose de um robô móvel. Como foi evidenciado, os sensores de posição e orientação podem ser classificados em dois tipos: sensores absolutos e sensores incrementais. Um procedimento usual é a fusão de dados de sensores, operando sob princípios diferentes, utilizando-se o filtro de Kalman estendido, para a estimação de posição e orientação de robôs móveis (BORENSTEIN; EVERETT; FENG, 1996).

A fusão de dados de sensores absolutos e incrementais permite que sejam obtidas estimativas de posição e orientação sob intervalos de tempos compatíveis com as necessidades do sistema de controle, sem que no entanto os erros cresçam de forma ilimitada. Portanto, o efeito resultante é de que os erros dos sensores incrementais são zerados cada vez que uma leitura dos sensores absolutos está disponível (LAGES, 1998).

Contudo, a aplicação alvo deste trabalho é a estabilização de postura de um robô móvel, em um ponto arbitrário. Assim, as trajetórias desenvolvidas pelo veículo serão predominantemente curtas. Deste modo, se os parâmetros geométricos do robô forem calibrados apropriadamente, os erros de *dead-reckoning*, acumulados durante as trajetórias de teste, serão desprezíveis. Além disto, para avaliar o desempenho dos controladores, o que importa é a pose medida pelos sensores, que deverá convergir para os valores desejados. Se os valores medidos correspondem ou não aos valores reais, é uma questão que está além do escopo do problema de controle.

A determinação da pose do robô será, portanto, realizada utilizando-se o método de *dead-reckoning*. Tendo em vista que o robô utilizado como estudo de caso neste trabalho é da classe $(2, 0)$, será considerada somente a formulação para os RMRs de acionamento diferencial. As equações de *dead-reckoning* para outras classes de robôs podem ser verificadas em BORENSTEIN; EVERETT; FENG (1996).

Conhecendo-se o raio das rodas (r) e o número de pulsos por volta dos *encoders* (Π), é possível calcular o deslocamento linear desenvolvido pelas rodas em um intervalo de tempo, conforme (LAGES, 1998, p. 115):

$$\Delta D_d(k) = \frac{2\pi r N P_d(k)}{\Pi} \tag{71}$$

$$\Delta D_e(k) = \frac{2\pi r N P_e(k)}{\Pi} \tag{72}$$

Nas expressões acima, $\Delta D_d(k)$ e $\Delta D_e(k)$ representam os deslocamentos lineares das rodas direita e esquerda, no intervalo de tempo entre kT e $(k+1)T$. De forma similar, $NP_d(k)$ e $NP_e(k)$ representam a contagem dos pulsos nos *encoders* direto e esquerdo, no intervalo de tempo entre kT e $(k+1)T$.

Os deslocamentos linear e angular do robô, considerando-se como referência o ponto no centro do eixo das rodas fixas, podem ser calculados por:

$$\Delta D(k) = \frac{\Delta D_d(k) + \Delta D_e(k)}{2} \tag{73}$$

$$\Delta \theta_c(k) = \frac{\Delta D_d(k) - \Delta D_e(k)}{2b} \tag{74}$$

$$\tag{75}$$

onde $2b$ representa a distância axial entre as rodas. Desta forma, assumindo-se que a trajetória percorrida pelo veículo entre os instantes k e $(k+1)$ é um arco de circunferência (Figura 8), têm-se:

$$x_c(k+1) = x_c(k) + \Delta s(k)\cos(\Upsilon) + d_{cm}\Delta\theta_c(k)\sin(\Upsilon) \quad (76)$$

$$y_c(k+1) = y_c(k) + \Delta s(k)\sin(\Upsilon) - d_{cm}\Delta\theta_c(k)\cos(\Upsilon) \quad (77)$$

$$\theta_c(k+1) = \theta_c(k) + \Delta\theta_c(k) \quad (78)$$

onde d_{cm} é a distância das rodas ao centro de massa e:

$$\Upsilon = \theta_c(k) + \frac{\Delta\theta_c(k)}{2} \quad (79)$$

$$\Delta s(k) = \Delta D(k)\frac{\sin\left(\frac{\Delta\theta_c(k)}{2}\right)}{\frac{\Delta\theta_c(k)}{2}} \quad (80)$$

Embora as expressões (76) a (78) tenham sido desenvolvidas considerando-se uma trajetória circular, são, também, exatas quando o deslocamento ocorre sobre uma reta. Isto deve-se ao fato de que $\sin(x)/x = 1$ quando $x \to 0$. Desta forma, a expressão (80) resume-se a $\Delta s(k) = \Delta D(k)$ quando $\Delta\theta_c = 0$, não havendo, portanto, indeterminação matemática.

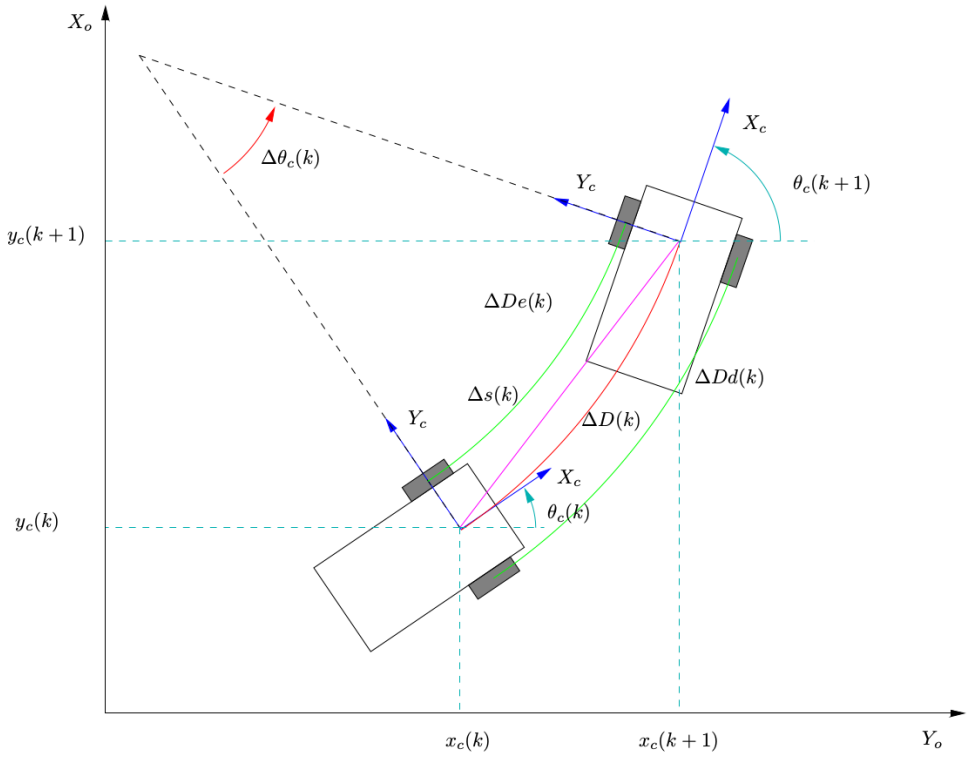


Figura 8: Trajetória circular utilizada para o desenvolvimento do algoritmo de odometria por encoder.
Fonte: LAGES (1998, p. 117)

3.3 Identificação de Sistemas

Há várias formas e técnicas de se obter modelos matemáticos para sistemas reais, sendo uma delas a modelagem caixa branca. Neste tipo de abordagem é imprescindível

o conhecimento aprofundado do sistema a ser modelado. Deste modo, além de estar bem familiarizado com as propriedades do sistema, é necessário conhecer as relações matemáticas que descrevem os fenômenos envolvidos (AGUIRRE, 2007, p. 31).

Por outro lado, a modelagem caixa-preta baseia-se somente nos dados de medição do sistema sob estudo. Uma das características desta técnica é que pouco ou nenhum conhecimento prévio do sistema é necessário e, portanto, a estrutura e os parâmetros do modelo são obtidos de forma empírica (NELLES, 2001; AGUIRRE, 2007).

Uma abordagem intermediária entre estas duas últimas denomina-se modelagem caixa-cinza. As técnicas deste tipo de abordagem caracterizam-se por usar informação auxiliar, que não se encontra no conjunto de dados utilizado durante a identificação. Muitas vezes, a ordem ou a estrutura do modelo são conhecidas, o que simplifica os processos posteriores de identificação. Contudo, não existe regra para a quantidade de informações ou o modo como o conhecimento a respeito do processo é utilizado, podendo haver abordagens caixa-cinza mais claras ou mais escuras, ou seja, tendendo para um dos métodos anteriores (NELLES, 2001; AGUIRRE, 2007). Uma visão detalhada sobre os diversos tipos de modelos, e suas propriedades matemáticas, pode ser encontrada em AGUIRRE (2007, pp. 56-66).

A identificação de sistemas é uma área da modelagem matemática que estuda métodos alternativos à abordagem caixa-branca (AGUIRRE, 2007, p. 36). De um modo geral, este procedimento pode ser representado pelo diagrama de blocos da Figura 9, onde $u(t)$ e $y(t)$ representam as entradas e saídas, respectivamente. Assim, a identificação de sistemas objetiva a obtenção de um modelo matemático que explique, pelo menos em parte e de forma aproximada, a relação de causa e efeito presente nos dados adquiridos do sistema dinâmico (AGUIRRE, 2007, p. 81).

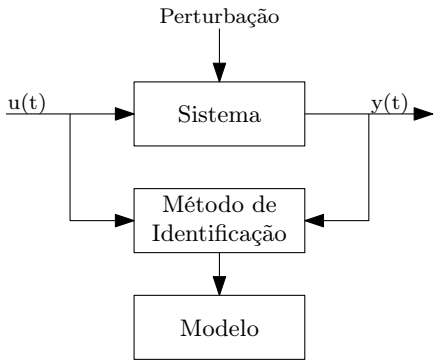


Figura 9: Diagrama de blocos da Identificação de Sistemas.
Fonte: LAGES (2017, p. 193).

Evidentemente um problema de identificação de sistemas é resolvido em diversas etapas, conforme o fluxograma na Figura 10. As principais etapas deste tipo de procedimento são definidas da seguinte forma (SÖDERSTRÖM; STOICA, 1989):

- Planejamento de experimentos e coleta de dados: Adquirir os dados de entrada e saída do processo é parte fundamental da identificação de sistemas. Conforme AGUIRRE (2007, p. 81), em muitos sistemas físicos reais, os únicos dados disponíveis serão aqueles adquiridos durante a operação normal do processo. Em outras situações, no entanto, será possível efetuar ensaios planejados especificamente para o processo sob identificação. Neste caso devem ser utilizados sinais de excitação

apropriados, de modo a extrair respostas com a maior quantidade possível de informações a respeito do processo[4].

- Determinação da estrutura do modelo: Nesta etapa é eleita uma estrutura de representação, baseando-se em um conjunto de modelos candidatos, e, também, no conhecimento *a priori* sobre o sistema a ser identificado (COELHO; COELHO, 2004). Isto é realizado, de um modo geral, elegendo-se a ordem do modelo ou dos polinômios envolvidos. Supondo, por exemplo, um problema de identificação determinística, em tempo contínuo, para o qual possa ser empregada uma representação do tipo função de transferência. A determinação da estrutura limitar-se-ía à definição do número de polos e de zeros e, também, se haveria inclusão (ou não) de um termo para o atraso de transporte.

- Estimação de parâmetros: Esta etapa inicia-se pela escolha do método que será utilizado para a estimação. Isto deve-se ao fato de que esta escolha depende do tipo de estrutura de modelo que será identificada. Se a representação escolhida na etapa anterior for linear nos parâmetros, pode ser utilizado o método clássico de mínimos quadrados[5] (*least squares*). São então determinados os parâmetros que fazem a resposta do modelo se aproximar das saídas medidas no processo físico, utilizando-se as mesmas entradas.

- Validação do Modelo: Nesta etapa são utilizados conjuntos de dados diferentes daquele empregado na estimação dos parâmetros para avaliar o desempenho do modelo identificado. Se a representação não ajustar de forma apropriada as medidas dos novos conjuntos de dados, o procedimento (Figura 10) deve ser executado novamente, de modo a determinar uma estrutura mais apropriada para representar o sistema.

Um fato importante a respeito dos modelos discutidos no próximo capítulo é que foram obtidos por modelagem caixa-branca. Contudo, como será evidenciado na seção 4.2, a obtenção dos parâmetros destas estruturas dependeria não somente das leis fundamentais relacionadas aos fenômenos observados mas também da realização de ensaios para determinação de características físicas individuais das peças do robô. Portanto, estes parâmetros serão determinados por um processo de identificação caixa-cinza, conforme apresentado na seção 7.1.

3.4 Controle de Robôs Móveis

Ao longo das últimas três décadas o controle de sistemas mecânicos não holonômicos tem sido estudado por grupos de pesquisa do mundo inteiro. Entre estes sistemas, pode-se evidenciar os robôs móveis, e quaisquer veículos que possuam restrições não integráveis. Particularmente, há um desafio considerável na síntese de leis de controle para os sistemas não lineares e que não podem ser transformados em sistemas lineares (AGUIAR; ATASSI; PASCOAL, 2000). A complexidade do problema é elevada, quando considerados os resultados demonstrados por BROCKETT (1983), que caracteriza a impossibili-

[4]Um maior detalhamento sobre o planejamento de experimentos pode ser obtido em LJUNG (1999, pp. 418-452).

[5]Uma visão ampla sobre o método de mínimos quadrados e sua implementação recursiva podem ser encontradas em AGUIRRE (2007, pp. 219-241) e AGUIRRE (2007, pp. 323-326).

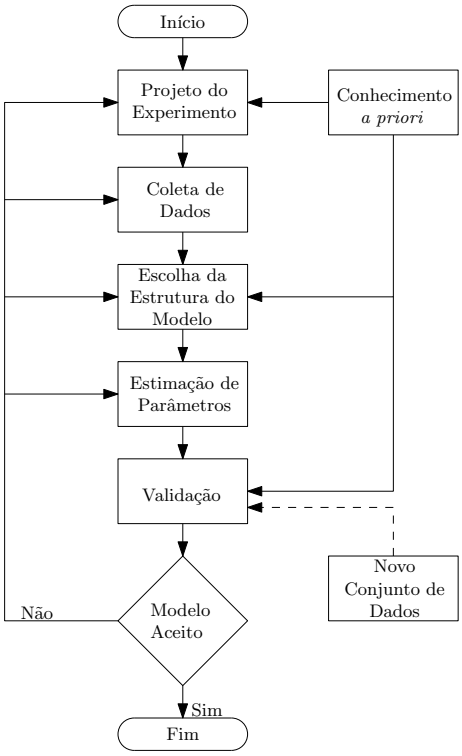


Figura 10: Fluxograma do procedimento de identificação de sistemas.
Fonte: SÖDERSTRÖM; STOICA (1989, p. 6)

dade de estabilização de sistemas não holonômicos sem deriva por leis de controle suaves e invariantes no tempo.

Os principais objetivos de controle para robôs móveis podem ser agrupados em duas classes (SICILIANO et al., 2009):

- **Estabilização de pose (*Parking*)**: Nesta abordagem, o robô deve convergir assintoticamente para uma pose de referência $\boldsymbol{\xi}_{ref}$, partindo de uma pose inicial $\boldsymbol{\xi}_i$. Entre os trabalhos nesta linha, podem ser citados CANUDAS DE WIT; SØRDALEN (1992); LAGES (1998); KUHNE; LAGES; GOMES DA SILVA JR. (2005).

- **Rastreamento de trajetória (*Tracking*)**: Nesta abordagem, o robô deve convergir assintoticamente e rastrear uma trajetória cartesiana desejada $(x_d(t),\ y_d(t))$, variante no tempo, a partir de uma pose inicial $\boldsymbol{\xi}_i$ que pode ou não fazer parte desta trajetória. Entre os trabalhos nesta linha, podem ser citados KANAYAMA et al. (1990); FIERRO; LEWIS (1995); ORIOLO; DE LUCA; VENDITTELLI (2002); CORRADINI; ORLANDO (2002); MARTINS et al. (2008); HUANG et al. (2014).

Um outro objetivo, denominado seguimento de caminho, pode ser considerado um caso particular do rastreamento de trajetória. Esta abordagem é menos restritiva, visto que não há especificação temporal para que a convergência seja alcançada (KÜHNE, 2005). Entre os trabalhos nesta linha, podem ser citados SARKAR; XIAOPING; KUMAR (1993); SAMSON (1995); ENCARNAÇÃO; PASCOAL (2002).

Leis de controle variantes no tempo, utilizadas em sistemas não holonômicos, foram demonstradas por SAMSON (1991), SAMSON; AIT-ABDERRAHIM (1991), POMET et al. (1992) e TEEL; MURRAY; WALSH (1995). Possuem como características baixas taxas de convergência e trajetórias altamente oscilatórias, podendo, em alguns casos, não

ser factíveis à uma implementação real (KÜHNE, 2005, p. 44). As leis de controle não suaves, por outro lado, podem superar as desvantagens associadas às leis de controle variantes no tempo. São comumente divididas em leis contínuas por partes ou de modo deslizante (KÜHNE, 2005, p. 49). Entre os trabalhos nesta linha, podem ser citados BLOCH; REYHANOGLU; MCCLAMROCH (1992), CANUDAS DE WIT; SØRDALEN (1992), LAGES (1998), ASTOLFI (1994) e CHWA (2004).

Um ponto importante em relação ao controle de robôs móveis é que a grande maioria das abordagens apresentadas na literatura baseiam-se somente no modelo cinemático do robô (KANAYAMA et al., 1990; WU et al., 1999; CARELLI; OLIVEIRA FREIRE, 2003; KUHNE; LAGES; GOMES DA SILVA JR., 2005). Isto significa assumir, por exemplo, que para um robô da classe $(2, 0)$ as entradas são as velocidades linear (v) e angular (ω), e as saídas as coordenadas de pose do robô. A principal razão para isto reside no fato de que considerar o modelo dinâmico eleva a complexidade do problema, e sua determinação precisa depende do conhecimento dos parâmetros relacionados ao veículo (como as massas e os momentos de inércia) e seus atuadores (MARTINS; SARCINELLI-FILHO; CARELLI, 2017).

Por outro lado, alguns robôs comerciais não permitem que sejam manipuladas as tensões aplicadas aos atuadores (ou os torques desenvolvidos por eles). Nestes casos existem malhas de controle internas, geralmente independentes por junta, responsáveis por controlar as velocidades dos atuadores, sendo esta a única variável passível de ser manipulada. Um caso similar ocorre quando são utilizados controladores convencionais (por exemplo, PIDs) para construir servos de velocidade ou posição para as juntas, mantendo-se estas como variáveis mecânicas manipuladas (LAGES, 1998; SICILIANO et al., 2009).

Nestes casos, assume-se que estas malhas internas são suficientemente rápidas em relação ao controle da cinemática, e que os efeitos de dinâmica sejam negligenciáveis. De fato, para robôs pequenos, para os quais os atuadores são frequentemente superdimensionados, essas considerações são válidas e, caso estas malhas de controle sejam apropriadamente ajustadas, o comportamento descrito pela cinemática deve ser alcançado. Contudo, com estas abordagens desprezam-se as perturbações mútuas produzidas pelos atuadores (LAGES, 1998).

No entanto, para aplicações de alto desempenho, quando são necessários movimentos de alta velocidade ou transporte de cargas pesadas, é essencial considerar os efeitos de dinâmica do robô, além de sua cinemática (MARTINS, 2009). Tendo em vista que o objetivo principal reside em solucionar os problemas de rastreamento de trajetória (ou de caminho) ou estabilização em uma pose (ou ponto), diversas abordagens têm sido apresentadas para lidar com os efeitos de dinâmica dos robôs.

A linearização por realimentação de estados apresenta-se como solução para reduzir a complexidade do modelo dinâmico (CAMPION; D'ANDREA-NOVEL; BASTIN, 1991; D'ANDREA-NOVEL; BASTIN; CAMPION, 1992; D'ANDRÉA-NOVEL; CAMPION; BASTIN, 1995; DE LA CRUZ; CARELLI, 2006; BARROS; LAGES, 2014; LAGES, 2017). Esta abordagem possibilita, além de tudo, utilizar ferramentas da teoria de sistemas lineares para o controle da parcela do modelo relacionada aos efeitos de dinâmica.

Quando os parâmetros do modelo são desconhecidos, faz-se necessário o uso de leis de controle adaptativas para ajustá-los *online*. Além disto, abordagens adaptativas possibilitam compensar variações paramétricas ocasionadas por mudanças nas condições operacionais, evitando que o desempenho seja degradado. Podem ser citados nesta linha os trabalhos de LAGES; HEMERLY (1998); DO; JIANG; PAN (2004); BARZAMINI; YAZDIZADEH; RAHMANI (2006); MARTINS et al. (2008); DE LA CRUZ; BASTOS;

CARELLI (2011); MARTINS; SARCINELLI-FILHO; CARELLI (2017).

Do ponto de vista de implementação, alguns autores têm proposto esquemas de controle do tipo cascata LAGES; HEMERLY (1998); MARTINS et al. (2008); MARTINS; SARCINELLI-FILHO; CARELLI (2017), tratando separadamente a cinemática e os efeitos de dinâmica. Neste sentido, o primeiro controlador calcula as referências de velocidade para o segundo. Este será, então, responsável por gerar as entradas dos atuadores, que levam à dinâmica desejada pelo primeiro deles.

No entanto, controladores monolíticos que tratam tanto da cinemática, quanto da dinâmica, também têm sido verificados na literatura (DE LA CRUZ; BASTOS; CARELLI, 2011). Em BARROS (2014), por exemplo, uma lei de controle não suave é utilizada para estabilização de pose, e o modelo dinâmico é linearizado por realimentação de estados. Então, é utilizada uma abordagem do tipo *backstepping*, baseada na teoria de Lyapunov.

As leis de controle utilizadas no desenvolvimento do presente trabalho são detalhadas a seguir.

3.4.1 Controlador PID

O controlador proporcional-integral-derivativo (PID) é o resultado da combinação das ações proporcional, integral e derivativa, para gerar o sinal de controle. Este método possui a capacidade de anular erros de regime permanente, por meio da ação integral, bem como antecipar o comportamento do processo, por meio da ação derivativa (BAZANELLA; GOMES DA SILVA JR., 2005, p. 49).

O sinal de controle gerado pelo controlador PID pode ser genericamente expresso por (ÅSTRÖM; HÄGGLUND, 1995, p. 64):

$$u_{pid}(t) = k\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \tag{81}$$

onde $e(t)$ é a diferença (ou o erro) entre a referência e a variável do processo, e os termos k, T_i e T_d são os parâmetros de sintonia denominados, respectivamente, ganho proporcional, tempo integral e tempo derivativo. A função de transferência deste controlador é dada por:

$$C(s) = \frac{U(s)}{E(S)} = k\left(1 + \frac{1}{sT_i} + sT_d\right) \tag{82}$$

No restante deste texto, contudo, será utilizada a seguinte representação:

$$C(s) = \frac{U(s)}{E(S)} = k_p + \frac{k_i}{s} + sk_d \tag{83}$$

onde:

$$k_p = k \tag{84}$$

$$k_i = \frac{k}{T_i} \tag{85}$$

$$k_d = kT_d \tag{86}$$

3.4.2 Linearização Entrada-Saída por Realimentação de Estados

O interesse por estratégias de linearização deve-se ao fato de que se o sistema puder ser linearizado, uma lei de controle linear poderá ser projetada, valendo-se da grande

quantidade de resultados teóricos disponíveis para sistemas lineares (LAGES, 1998, p. 32).

O método mais conhecido para esta finalidade consiste em aplicar uma aproximação linear para o sistema, expandindo-o em série de Taylor em torno de um ponto, e posteriormente desprezando os termos de ordem superior. Na linearização por realimentação, em contrapartida, obtém-se o cancelamento das não linearidades sem que os termos de ordem superior sejam desprezados, mas sim cancelados através da manipulação apropriada das entradas do sistema. Desta forma obtém-se uma descrição linear exata para o sistema não linear (SLOTINE; LI, 1991; LAGES, 1998).

Por outro lado, se os parâmetros do sistema não forem bem conhecidos, o cancelamento dos termos de ordem superior não será exato e, portanto, a descrição linear obtida não corresponderá ao sistema original LAGES (1998, p. 32). Deste modo, os efeitos das diferenças paramétricas apresentar-se-ão como distúrbios ao controlador, projetado para o sistema linearizado, que poderá perder eficiência ou até mesmo tornar-se instável. Contudo, conforme LAGES (1998, p. 33), na maioria das vezes, os parâmetros podem ser determinados apropriadamente por calibração.

Considerando-se a classe de sistemas não lineares multivariáveis com o mesmo número de entradas e saídas, da forma (SLOTINE; LI, 1991, p. 266):

$$\dot{x} = f(x) + \sum_{j=1}^{m} g_j(x)u_j \tag{87}$$

$$\begin{aligned} y_1 &= h_1(x) \\ &\vdots \\ y_m &= h_m(x) \end{aligned} \tag{88}$$

onde x é o vetor de estados ($n \times 1$), u é o vetor de entradas de controle ($m \times 1$), y é o vetor de saídas do sistema ($m \times 1$), e $f(\cdot)$, $g_i(\cdot)$ e $h(\cdot)$ são campos vetoriais suaves. O procedimento de linearização entrada-saída por realimentação de estados consiste em derivar as saídas y_i, até que as entradas apareçam explicitamente, e então, projetar uma lei de controle u para cancelar as não linearidades.

Diferenciando-se a saída y_i em relação ao tempo, obtém-se (LAGES, 1998, p. 34):

$$\dot{y}_i = L_f h_i(x) + \sum_{j=1}^{m} (L_{g_j} h_i(x))u_j \tag{89}$$

onde $L_f h_i(x)$ e $L_{g_j} h_i$ representam, respectivamente, as derivada de Lie (SLOTINE; LI, 1991, p. 229) da saída h_i com relação à f e a g_j, definidas por:

$$L_f h_i(x) = \sum_{j=1}^{n} \frac{\partial h_i(x)}{\partial x_j} \dot{x}_{j_f} \tag{90}$$

$$L_{g_j} h_i(x) = \sum_{j=1}^{n} \frac{\partial h_i(x)}{\partial x_j} \dot{x}_{j_g} \tag{91}$$

Nestas expressões $\dot{x}_{j_f}$ e $\dot{x}_{j_g}$ representam, respectivamente, as partes relacionadas à $f(x)$ e $g_j(x)$, da j-ésima equação de estado do sistema representado por (87) e (88). De forma genérica, assumindo-se que r_i é o menor valor inteiro para o qual no mínimo uma das

entradas apareça em y^{r_i}, tem-se:

$$y_i^{(r_i)} = L_f^{r_i} h_i(x) + \sum_{j=1}^{m} \left(L_{g_j} L_f^{r_i - 1} h_i(x) u_j \right) \tag{92}$$

sendo (LAGES, 1998, p. 34):

$$L_f^{r_i} h_i(x) = \sum_{j=1}^{n} \frac{\partial L_f^{r_i - 1} h_i(x)}{\partial x_j} \dot{x}_{j_f} \tag{93}$$

$$L_{g_j} L_f^{r_i - 1} h_i(x) = \sum_{j=1}^{n} \frac{\partial L_{g_j} L_f^{r_i - 2} h_i(x)}{\partial x_j} \dot{x}_{j_g} \tag{94}$$

onde no mínimo um dos termos $L_{g_j} L_f^{r_i - 1} h_i(x) \neq 0$, para todo x pertencente à região onde a linearização é válida. Realizando-se o procedimento acima para cada saída y_i, obtém-se (SLOTINE; LI, 1991, p. 267):

$$\begin{bmatrix} y_1^{(r_1)} \\ \cdots \\ \cdots \\ y_m^{(r_m)} \end{bmatrix} = \begin{bmatrix} L_f^{r_1} h_1(x) \\ \cdots \\ \cdots \\ L_f^{r_m} h_m(x) \end{bmatrix} + \boldsymbol{E}(x) u \tag{95}$$

onde $\boldsymbol{E}(x)$ é a matriz $(m \times m)$ definida por:

$$\boldsymbol{E}(x) = \begin{bmatrix} L_{g_1} L_f^{r_1 - 1} h_1(x) & \cdots & L_{g_m} L_f^{r_1 - 1} h_1(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{r_m - 1} h_1(x) & \cdots & L_{g_m} L_f^{r_m - 1} h_1(x) \end{bmatrix} \tag{96}$$

Deste modo, se a matriz $\boldsymbol{E}(x)$ for invertível para todo x pertencente à região de interesse, obtém-se a seguinte lei de controle por realimentação de estados:

$$u(x) = -\boldsymbol{E}^{-1}(x) \begin{bmatrix} L_f^{r_1} h_1(x) \\ \cdots \\ \cdots \\ L_f^{r_m} h_m(x) \end{bmatrix} + \boldsymbol{E}^{-1}(x) \nu \tag{97}$$

Com esta transformação da entrada, o sistema resultante em malha-fechada (Figura 11) é linear sob o ponto de vista entrada-saída, e possui a forma:

$$\begin{bmatrix} y_1^{(r_1)} \\ \cdots \\ \cdots \\ y_m^{(r_m)} \end{bmatrix} = \begin{bmatrix} \nu_1 \\ \cdots \\ \cdots \\ \nu_m \end{bmatrix} \tag{98}$$

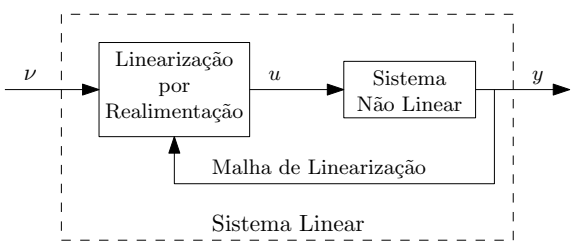


Figura 11: Diagrama de blocos da linearização por realimentação de estados.
Fonte: Adaptado de LAGES (1998).

Como pode ser observado, cada entrada ν_i afeta somente a saída $y_i^{(r_i)}$, resultando em m sistemas lineares e desacoplados. Deste modo, pode-se utilizar uma lei de controle linear para cada um destes sistemas, como por exemplo o PID, conforme representado pela Figura 12. Uma análise mais abrangente sobre a linearização por realimentação de estados pode ser encontrada em ISIDORI (1995) e SLOTINE; LI (1991).

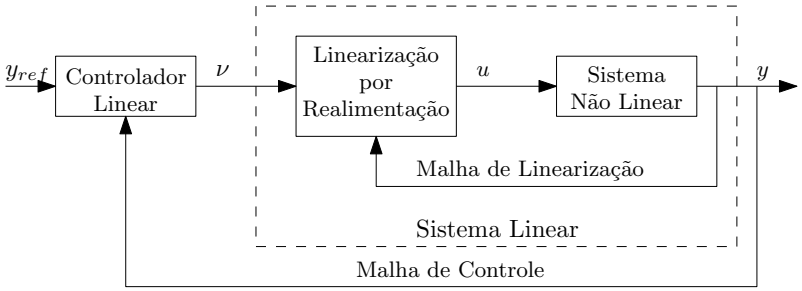


Figura 12: Diagrama de blocos de um controlador linearizante.
Fonte: LAGES (1998, p. 36).

3.4.3 Controle por Transformação Descontínua

O método que será apresentado aqui possui como finalidade a estabilização de um robô móvel em uma pose arbitrária. Embora seja possível aplicar esta estratégia para todos os tipos de RMRs, são necessárias pequenas adaptações para cada uma das classes. Assim, tendo em vista que o robô utilizado como estudo de caso neste trabalho é da classe $(2, 0)$, será considerada somente a formulação para os RMRs desta classe.

Foi demonstrado por BROCKETT (1982) que um sistema não holonômico sem deriva não pode ser assintoticamente estabilizado, em um ponto arbitrário, através de uma lei de controle suave e invariante no tempo. A estratégia de controle de pose que será apresentada (Figura 13) segue a estrutura demonstrada em LAGES (2017), é não linear e baseia-se na teoria de Lyapunov e na transformação do modelo para coordenadas polares.

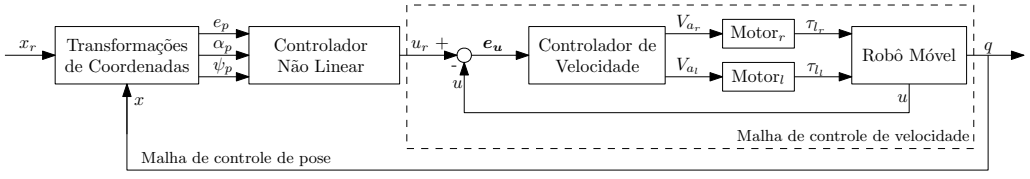


Figura 13: Controle de pose por transformação descontínua.

Conforme LAGES (1998), para aplicações de controle em geral, o que interessa são basicamente as coordenadas de pose do robô e, portanto, será utilizado somente o modelo cinemático de pose (26), específico para a classe $(2, 0)$. Conforme a Tabela 1, este modelo é definido por:

$$\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \cos(\theta_c) & 0 \\ \sin(\theta_c) & 0 \\ 0 & 1 \end{bmatrix}}_{\boldsymbol{B}(\boldsymbol{x})} \boldsymbol{u} \tag{99}$$

onde $\boldsymbol{u} = \begin{bmatrix} v & \omega \end{bmatrix}^T$ caracteriza as velocidades linear e angular do robô. A consideração somente da cinemática deve-se ao fato de que, para o desenvolvimento desta lei de controle, assumir-se-á a existência de um controlador de velocidade, responsável por assegurar a convergência de $\boldsymbol{u}$ para $\boldsymbol{u}_r$, de forma que a dinâmica possa ser negligenciada pelo controlador de pose. Desta forma, a região tracejada na Figura 13 é considerada como um único bloco, e a expressão (99) reescrita na forma:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \cos(\theta_c) & 0 \\ \sin(\theta_c) & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{u}_r \tag{100}$$

Assim como na maioria dos controladores baseados na teoria de Lyapunov, assume-se que o sistema deve convergir para sua origem (LAGES, 2017). No entanto, é de interesse estabilizar o robô em uma pose arbitrária $\boldsymbol{x}_r = \begin{bmatrix} x_{cr} & y_{cr} & \theta_{cr} \end{bmatrix}^T$, o que pode ser alcançado com uma transformação de coordenadas, que desloca a origem do novo sistema para a pose de referência, obtida por (BARROS, 2014, p. 80):

$$\begin{bmatrix} \overline{x}_c \\ \overline{y}_c \\ \overline{\theta}_c \end{bmatrix} \triangleq \begin{bmatrix} \cos\theta_{cr} & \sin\theta_{cr} & 0 \\ -\sin\theta_{cr} & \cos\theta_{cr} & 0 \\ 0 & 0 & 1 \end{bmatrix} (\boldsymbol{x} - \boldsymbol{x}_r) \tag{101}$$

A estabilização será, então, obtida utilizando-se uma transformação de coordenadas não suave (BARROS, 2014), definida por:

$$\begin{aligned} e_p &= \sqrt{\overline{x}_c^2 + \overline{y}_c^2} \\ \psi_p &= atan2(\overline{y}_c, \overline{x}_c) \\ \alpha_p &= \overline{\theta}_c - \psi_p \end{aligned} \tag{102}$$

Deste modo, o modelo (100) é reescrito na forma:

$$\begin{cases} \dot{e}_p = \cos\alpha_p u_{r_1} \\ \dot{\psi}_p = \frac{\sin\alpha_p}{e_p} u_{r_1} \\ \dot{\alpha}_p = -\frac{\sin\alpha_p}{e_p} u_{r_1} + u_{r_2} \end{cases} \tag{103}$$

Cabe salientar que (103) é válida somente para robôs de acionamento diferencial. Um procedimento similar para as demais classes de robôs móveis pode ser verificado em LAGES (1998, pp. 59-70).

Escolhendo-se a seguinte expressão como candidata à função de Lyapunov:

$$V = \frac{1}{2}(\lambda_1 e_p^2 + \lambda_2 \alpha_p^2 + \lambda_3 \psi_p^2) \tag{104}$$

com $\lambda_i > 0$, pode-se verificar que a derivada temporal de (104) é dada por:

$$\dot{V} = (\lambda_1 e_p \dot{e}_p + \lambda_2 \alpha_p \dot{\alpha}_p + \lambda_3 \psi_p \dot{\psi}_p) \tag{105}$$

Além disto, substituindo $\dot{e}_p$, $\dot{\alpha}_p$, $\dot{\psi}_p$ de (103) em (105), é obtida a seguinte expressão:

$$\dot{V} = \lambda_1 e_p \cos\alpha_p u_{r_1} - \lambda_2 \alpha_p \frac{\sin\alpha_p}{e_p} u_{r_1} + \lambda_2 \alpha_p u_{r_2} + \lambda_3 \psi_p \frac{\sin\alpha_p}{e_p} u_{r_1} \tag{106}$$

Ainda, para que $\dot{V}$ seja semi-definida negativa, podem ser escolhidas as seguintes entradas de controle (BARROS, 2014, p. 82):

$$u_{r_1} \triangleq -\gamma_1 e_p \cos\alpha_p \tag{107}$$

$$u_{r_2} \triangleq -\gamma_2\alpha_p - \gamma_1 \cos\alpha_p \sin\alpha_p + \gamma_1 \frac{\lambda_3}{\lambda_2} \cos\alpha_p \frac{\sin\alpha_p}{\alpha_p}\psi_p \tag{108}$$

Estas entradas de controle permitem que (106) seja reescrita na seguinte forma:

$$\dot{V} = -\gamma_1\lambda_1 e_p^2 \cos^2\alpha_p - \gamma_2\lambda_2\alpha_p^2 \leq 0 \tag{109}$$

A expressão (109), juntamente com a continuidade de V, assegura a estabilidade do sistema (LAGES, 2017). No entanto, a convergência para a origem ainda necessita ser comprovada.

Tendo em vista que V é limitada inferiormente, que V é não crescente, uma vez que $\dot{V} \leq 0$ e, também, que $\dot{V}$ é uniformemente contínua, pois $\ddot{V}$ é limitada, tem-se pelo lema de Barbalat (POPOV, 1973) que $\dot{V} \to 0$, e de (109) pode-se escrever:

$$-\gamma_1\lambda_1 e_p^2 \cos^2\alpha_p - \gamma_2\lambda_2\alpha_p^2 \to 0 \tag{110}$$

Ainda, γ_2 e λ_2 são termos positivos e, portanto, para que $\gamma_2\lambda_2\alpha_p^2 \to 0$, é necessário que $\alpha_p \to 0$. Por outro lado, fazendo $\alpha_p = 0$ em (110) verifica-se que $-\gamma_1\lambda_1 e_p^2 \to 0$. Logo, torna-se evidente que $e_p \to 0$ (BARROS; LAGES, 2014). A análise da convergência de ψ_p é realizada a partir do sistema em malha-fechada, que é obtido aplicando-se as realimentações (107) e (108) a (103), dado por:

$$\begin{cases} \dot{e}_p &= -\gamma_1 e_p \cos^2\alpha_p \\ \dot{\psi}_p &= -\gamma_1 \sin\alpha_p \cos\alpha_p \\ \dot{\alpha}_p &= -\gamma_2\alpha_p + \gamma_1 \frac{\lambda_3}{\lambda_2}\psi_p \frac{\sin\alpha_p}{\alpha_p} \cos\alpha_p \end{cases} \tag{111}$$

Tendo em vista que ψ_p é limitado e que a partir de (111) pode-se concluir que $\dot{\psi}_p$ também é limitado, segue que ψ_p é uniformemente contínuo. Além disto, $\dot{\alpha}_p$ é uniformemente contínua, pois $\ddot{\alpha}_p < \infty$ e, uma vez que $\alpha_p \to 0$, tem-se pelo lema de Barbalat que $\dot{\alpha}_p \to 0$. Portanto, com $\alpha_p \to 0$ e $\dot{\alpha}_p \to 0$, a terceira expressão de (111) pode ser escrita como (BARROS; LAGES, 2014):

$$\dot{\alpha}_p = \gamma_1 \frac{\lambda_3}{\lambda_2}\psi_p \frac{\sin\alpha_p}{\alpha_p} \cos\alpha_p = 0 \tag{112}$$

Da expressão (112) conclui-se que $\psi_p \to 0$ e, portanto, as entradas de controle (107) e (108) estabilizam o sistema (103) em sua origem (BARROS, 2014; LAGES, 2017).

4 MODELO MATEMÁTICO DO ROBÔ MÓVEL TWIL

O robô móvel Twil possui duas rodas fixas e uma roda de apoio do tipo *caster wheel*, que se caracteriza como uma roda orientável não centrada. Estas características construtivas o enquadram na classe (2,0). Modelos cinemáticos e dinâmicos para esta e outras classes de robôs móveis foram apresentados por LAGES (1998). Neste trabalho, porém, será utilizada a particularização destes resultados proposta por BARROS (2014) para o Twil.

A descrição do modelo será realizada tomando como referência o sistema de coordenadas apresentado na Figura 14. Neste sistema, X_c e Y_c são os eixos do sistema de coordenadas do robô, posicionado no centro do eixo das rodas de tração. O sistema de coordenadas inercial é formado pelos eixos X_o e Y_o e a pose do robô, em relação a este sistema, é definida por $\boldsymbol{x} = \begin{bmatrix} x_c & y_c & \theta_c \end{bmatrix}^T$.

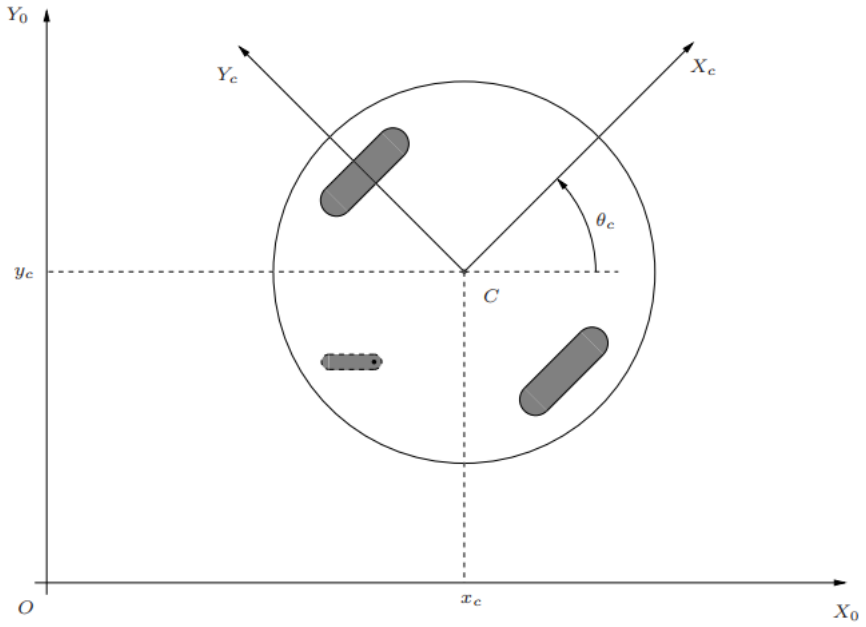


Figura 14: Sistema de coordenadas do Robô.

O modelo dinâmico de configuração apresentado por BARROS (2014) possui a seguinte estrutura:

$$\begin{cases} \dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{u} \\ \dot{\boldsymbol{u}} = \boldsymbol{H}(\boldsymbol{\beta})^{-1}(-\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u}) + \boldsymbol{F}(\boldsymbol{\beta})\boldsymbol{\tau}_0) \end{cases} \tag{113}$$

Nesta estrutura as entradas são os torques impostos pelos atuadores ($\boldsymbol{\tau}_0 = \begin{bmatrix} \tau_r, \tau_l \end{bmatrix}^T$) e

as saídas são as coordenadas de pose e as posições angulares das rodas ($\boldsymbol{q}$). Os elementos $\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u})$, $\boldsymbol{H}(\boldsymbol{\beta})$ e $\boldsymbol{F}(\boldsymbol{\beta})$ são matrizes que dependem das propriedades cinemáticas (geométricas) e dinâmicas (massas e inércias) do robô, dados por:

$$\boldsymbol{H}(\boldsymbol{\beta}) = I \tag{114}$$

$$\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u}) = -\begin{bmatrix} 0 & k_5 \\ k_6 & 0 \end{bmatrix} \begin{bmatrix} u_1 u_2 \\ {u_2}^2 \end{bmatrix} = -\boldsymbol{f}(\boldsymbol{u}) \tag{115}$$

$$\boldsymbol{F}(\boldsymbol{\beta}) = \begin{bmatrix} k_7 & k_7 \\ k_8 & -k_8 \end{bmatrix} = \boldsymbol{F} \tag{116}$$

onde k_5, k_6, k_7 e k_8 são constantes que dependem somente dos parâmetros geométricos e de inércia do robô e o vetor $\boldsymbol{u} = \begin{bmatrix} v & \omega \end{bmatrix}^T$ representa as velocidades linear e angular cuja relação com o movimento das rodas é caracterizada pelas seguintes expressões, obtidas de (35):

$$v = \frac{r(\dot{\varphi}_r + \dot{\varphi}_l)}{2} = u_1 \tag{117}$$

$$\omega = \frac{r(\dot{\varphi}_r - \dot{\varphi}_l)}{2b} = u_2 \tag{118}$$

onde os parâmetros r e b representam, respectivamente, o raio da base e a distância entre as rodas e o eixo de simetria.

Conforme LAGES (2017, p. 203), a partir da segunda equação de (113), pode-se ainda verificar que o termo $k_5 u_2^2$ representa a força centrífuga, ao passo que o termo $k_6 u_1 u_2$ representa a força devido à aceleração de Coriolis. Tendo em vista que a aceleração linear é proporcional à média dos torques aplicados às juntas atuadas, $1/k_7$ caracteriza a massa do robô. De modo similar, a aceleração angular é proporcional à diferença entre os torques aplicados às juntas e, portanto, $1/k_8$ representa o momento de inércia da base.

O modelo (113) pode ser interpretado como uma cascata entre a dinâmica do robô, cuja entrada são os torques ($\boldsymbol{\tau_o}$) e a saída são as velocidades da base ($\boldsymbol{u}$), e sua cinemática, que possui como entrada as velocidades da base ($\boldsymbol{u}$) e como saída a pose e as posições angulares das rodas ($\boldsymbol{q}$), conforme representado na Figura 15.

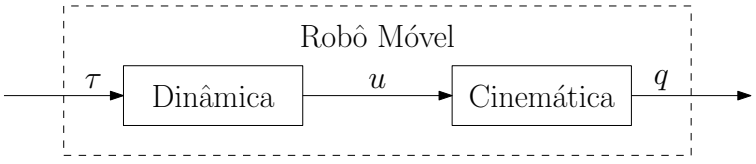


Figura 15: Representação em blocos do modelo do robô móvel.
Fonte: LAGES (2017, p. 201)

4.1 Modelo dos atuadores

O modelo apresentado no início desta seção representa de maneira apropriada os efeitos físicos aos quais a base do robô é submetida. Nesta estrutura, pressupõe-se a existência de um vetor de torques generalizado (τ) sendo aplicado às juntas do robô. Na grande

maioria dos robôs móveis, estes torques são fornecidos por motores de corrente contínua, acoplados a estas juntas por meio de engrenagens. Portanto, os efeitos dos atuadores devem ser considerados no modelo dinâmico.

Será utilizado como referência o modelo de um servomecanismo, constituído de um motor de corrente contínua com ímã permanente e um sistema de transmissão, equivalente ao tipo de atuador empregado nas juntas do Twil, cuja representação encontra-se na Figura 16. Os equacionamentos a seguir baseiam-se nas análises de FU; GONZALES; LEE (1987) e SPONG; HUTCHINSON; VIDYASAGAR (2005).

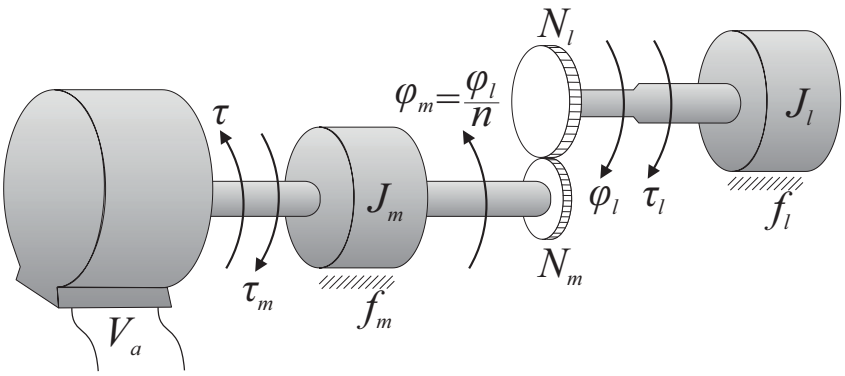


Figura 16: Representação de um servomecanismo baseado em um motor de corrente contínua com imã permanente.
Fonte: FU; GONZALES; LEE (1987, p. 207)

Conforme a Figura 16, o eixo do motor é acoplado à carga por meio de um sistema de transmissão. Supondo que não haja escorregamento neste sistema, pode-se assumir que os deslocamentos lineares nas engrenagens, do lado do motor (d_m) e do lado da carga (d_l), são os mesmos ($d_m = d_l$). Por outro lado, o deslocamento linear em cada engrenagem é dado pelo produto entre o deslocamento angular (φ) e o raio (r), conforme:

$$r_m \varphi_m = r_l \varphi_l \tag{119}$$

Uma vez que o número de dentes é proporcional ao raio de cada engrenagem, são obtidas as equações:

$$N_m \varphi_m = N_l \varphi_l \tag{120}$$

$$\frac{N_m}{N_l} = \frac{\varphi_l}{\varphi_m} = n \tag{121}$$

onde n representa a relação de engrenagens. Desta forma, as variáveis de junta, medidas no lado da carga, são obtidas a partir das seguintes expressões (FU; GONZALES; LEE, 1987):

$$\varphi_l = n\varphi_m \tag{122}$$

$$\dot{\varphi}_l = n\dot{\varphi}_m \tag{123}$$

$$\ddot{\varphi}_l = n\ddot{\varphi}_m \tag{124}$$

O torque desenvolvido pelo atuador (τ), na presença de uma carga acoplada ao mecanismo, é igual à soma dos torques dissipados por perdas no eixo do motor (τ_m) e por

reações da carga (τ_l), referidas ao eixo do motor (τ_l^*), conforme:

$$\tau = \tau_m + \tau_l^* \tag{125}$$

As perdas do lado do motor (τ_m) e o torque de reação da carga (τ_l) são caracterizados pelas seguintes expressões (FU; GONZALES; LEE, 1987):

$$\tau_m = J_m\ddot{\varphi}_m + f_m\dot{\varphi}_m \tag{126}$$
$$\tau_l = J_l\ddot{\varphi}_l + f_l\dot{\varphi}_l \tag{127}$$

onde f_m e J_m representam, respectivamente, o coeficiente de atrito viscoso com os mancais e o momento de inércia do rotor. De forma similar, f_l e J_l caracterizam, respectivamente, o coeficiente de atrito viscoso e o momento de inércia da carga.

Ainda segundo FU; GONZALES; LEE (1987), o princípio da conservação da energia requer que o trabalho realizado pela carga, referido ao eixo da mesma ($\tau_l\varphi_l$), seja igual ao trabalho realizado por esta, refletido ao eixo do motor ($\tau_l^*\varphi_m$), conforme:

$$\tau_l^*\varphi_m = \tau_l\varphi_l \tag{128}$$

Desta condição é obtida a expressão que define o torque de carga refletido ao eixo do motor (τ_l^*), dada por:

$$\tau_l^* = \frac{\tau_l\varphi_l}{\varphi_m} = n\tau_l \tag{129}$$

Substituindo as expressões (123), (124) e (127) em (129) é obtida sua forma explícita, representada por:

$$\tau_l^* = n^2(J_l\ddot{\varphi}_m + f_l\dot{\varphi}_m) \tag{130}$$

A forma explícita do torque gerado pelo atuador (τ) em relação ao eixo do motor é encontrada substituindo (127) e (130) em (125), conforme:

$$\tau = (J_m + n^2J_l)\ddot{\varphi}_m + (f_m + n^2f_l)\dot{\varphi}_m = J_e\ddot{\varphi}_m + f_e\dot{\varphi}_m \tag{131}$$

Nesta expressão $J_e = J_m + n^2J_l$ e $f_e = f_m + n^2f_l$ representam, respectivamente, os valores efetivos do momento de inércia e do coeficiente de atrito viscoso referenciados ao eixo do motor.

A análise do subsistema mecânico foi realizada nos parágrafos acima. Serão, a partir de agora, verificadas as relações que regem as dinâmicas do dispositivo, tomando como referência o circuito equivalente da Figura 17. Sabe-se que, em um motor de corrente contínua com ímãs permanentes, o torque desenvolvido no eixo do motor (τ) possui dependência apenas com a corrente de armadura (i_a), conforme a seguinte equação:

$$\tau = K_Ti_a \tag{132}$$

onde K_T é a constante de proporcionalidade de torque do motor. Já a força contra-eletromotriz desenvolvida pelo motor possui dependência apenas com a velocidade angular ($\dot{\varphi}_m$):

$$e_a = K_a\dot{\varphi}_m \tag{133}$$

Na expressão anterior, K_a representa a constante de proporcionalidade da força contra-eletromotriz. A partir da malha do subsistema eletromagnético, verifica-se a relação entre

a tensão de entrada (V_a), a velocidade angular ($\dot{\varphi}_m$) e a corrente de armadura (i_a), dada por:

$$V_a \quad = \quad R_a i_a + L_a \frac{di_a}{dt} + e_a \tag{134}$$

onde os parâmetros R_a e L_a representam a resistência elétrica e a indutância do circuito de armadura. Aplicando-se a transformada de Laplace à equação (134), sob condições iniciais nulas, e resolvendo em relação a $I_a(s)$, obtém-se:

$$I_a(s) = \frac{V_a(s) - sK_a\Phi_m(s)}{R_a + sL_a} \tag{135}$$

Um resultado similar é obtido para (132), cuja transformada de Laplace é dada por:

$$T(s) = K_T I_a(s) = K_T \frac{V_a(s) - sK_a\Phi_m(s)}{R_a + sL_a} \tag{136}$$

Ainda em relação ao torque gerado, porém em relação ao subsistema mecânico (131), esta transformada resulta em:

$$T(s) = s^2 J_e \Phi_m(s) + s f_e \Phi_m(s) \tag{137}$$

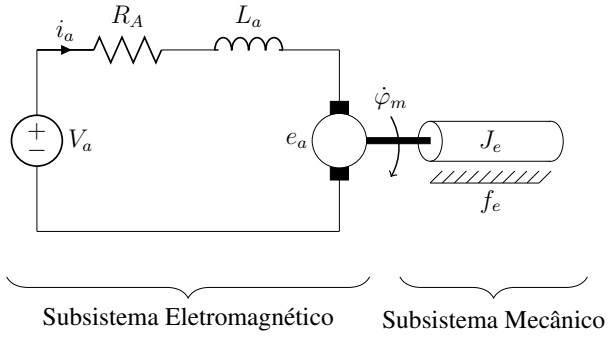


Figura 17: Representação do circuito elétrico equivalente de um motor de corrente contínua com imã permanente controlado pela tensão de armadura.

Assim, igualando-se (136) e (137) e rearranjando o resultado, de modo a explicitar $\Phi_m(s)/V_a(s)$, é encontrada a função de transferência entre o deslocamento angular (φ_m) e a tensão de armadura (V_a), dada por:

$$\frac{\Phi_m(s)}{V_a(s)} = \frac{K_a}{s[s^2 J_e L_a + (L_a f_e + R_a J_e)s + R_a f_e + K_T K_a]} \tag{138}$$

De forma similar, a função de transferência entre o deslocamento angular na carga (φ_l) e a tensão de armadura é obtida aplicando-se a relação (122) à (138), o que leva à expressão:

$$\frac{\Phi_l(s)}{V_a(s)} = \frac{nK_a}{s[s^2 J_e L_a + (L_a f_e + R_a J_e)s + R_a f_e + K_T K_a]} \tag{139}$$

Conforme (SPONG; HUTCHINSON; VIDYASAGAR, 2005), frequentemente pode-se assumir que a constante de tempo elétrica (L_a/R_a) é suficientemente menor que a constante de tempo mecânica (J_e/f_e), de modo que o efeito da indutância de armadura

(L_a) possa ser desprezado. Esta consideração é razoável para a maioria dos sistemas eletromecânicos e, principalmente, para o atuador em estudo nesta subseção, permitindo que as funções de transferência (138) e (139) sejam simplificadas para:

$$\frac{\Phi_m(s)}{V_a(s)} = \frac{K_a}{s(R_a J_e s + R_a f_e + K_T K_a)} = \frac{K}{s(T_m s + 1)} \tag{140}$$

$$\frac{\Phi_l(s)}{V_a(s)} = \frac{nK_a}{s(R_a J_e s + R_a f_e + K_T K_a)} = \frac{nK}{s(T_m s + 1)} \tag{141}$$

onde os parâmetros K e T_m representam, respectivamente, as constantes de ganho e de tempo do motor, definidas por:

$$K \triangleq \frac{K_T}{R_a f_e + K_a K_T} \tag{142}$$

$$T_m \triangleq \frac{R_a J_e}{R_a f_e + K_a K_T} \tag{143}$$

4.2 Modelo Dinâmico Aumentado

Nesta seção, um modelo aumentado de (113) será desenvolvido, para que seja possível considerar os efeitos de dinâmica dos atuadores. Neste sentido, cada uma das entradas será relacionada ao torque líquido entregue por um atuador à sua carga. Portanto, cada uma das entradas do robô recebe o torque desenvolvido no eixo do seu respectivo atuador, subtraídas as perdas por transmissão, refletido ao eixo da carga. Serão assumidos, por motivo de simplicidade, que os atuadores são perfeitamente iguais e, portanto, possuem os mesmos coeficientes de atrito viscoso (f_m), momentos de inércia em relação ao eixo (J_m), constantes elétrica (K_a) e de torque (K_T), indutâncias (L_a) e resistências de armadura (R_a).

Tomando como referência a modelagem do atuador apresentada na seção anterior, as expressões (125) e (132) caracterizam, respectivamente, o torque aplicado ao servomecanismo e aquele gerado pelo atuador. Substituindo-se (126) em (125) e igualando o resultado a (132), é obtida a seguinte expressão:

$$\tau = f_m \dot{\varphi}_m + J_m \ddot{\varphi}_m + \tau_l^* = K_T i_a \tag{144}$$

para a qual aplicando-se a relação (129) e rearranjando o resultado em função do torque de carga, obtém-se (145):

$$\tau_l = \frac{K_T i_a - f_m \dot{\varphi}_m - J_m \ddot{\varphi}_m}{n} \tag{145}$$

Tendo em vista que o sensor de deslocamento pode estar no eixo da carga, o que é o caso do Twil, pode-se utilizar (123) e (124) em (145) para obter:

$$\tau_l \;=\; \frac{K_T i_a}{n} - \frac{f_m \dot{\varphi}_l}{n^2} - \frac{J_m \ddot{\varphi}_l}{n^2} \tag{146}$$

Retornando ao modelo desenvolvido por (BARROS, 2014), o vetor de torques ($\boldsymbol{\tau_o}$) será substituído pelo vetor de torques de carga dos atuadores $\boldsymbol{\tau_{l_o}} = \begin{bmatrix} \tau_{l_r} & \tau_{l_l} \end{bmatrix}^T$. Deste modo, a expressão (146) será parametrizada em função de cada um dos atuadores. Realizando a soma e a subtração das equações (117) e (118), são determinadas as velocidades

angulares das rodas, representadas por:

$$\dot{\varphi}_{l_r} = \frac{u_1}{r} + \frac{u_2}{2c} \tag{147}$$

$$\dot{\varphi}_{l_l} = \frac{u_1}{r} - \frac{u_2}{2c} \tag{148}$$

onde $c = r/2b$. Resultados similares são obtidos para as acelerações, com a substituição de $\dot{\varphi}$ por $\ddot{\varphi}$ e de $\boldsymbol{u}$ por $\dot{\boldsymbol{u}}$, nas respectivas equações. Assim, os torques de carga dos atuadores direito e esquerdo são representados por:

$$\tau_{l_r} = \frac{K_T i_{ar}}{n} - \frac{f_m}{n^2}\left(\frac{u_1}{r} + \frac{u_2}{2c}\right) - \frac{J_m}{n^2}\left(\frac{\dot{u}_1}{r} + \frac{\dot{u}_2}{2c}\right) \tag{149}$$

$$\tau_{l_l} = \frac{K_T i_{al}}{n} - \frac{f_m}{n^2}\left(\frac{u_1}{r} - \frac{u_2}{2c}\right) - \frac{J_m}{n^2}\left(\frac{\dot{u}_1}{r} - \frac{\dot{u}_2}{2c}\right) \tag{150}$$

Por outro lado, a parcela de (113) que representa somente os efeitos da dinâmica é dada por:

$$\dot{\boldsymbol{u}} = \begin{bmatrix} 0 & k_5 \\ k_6 & 0 \end{bmatrix} \begin{bmatrix} u_1 u_2 \\ {u_2}^2 \end{bmatrix} + \begin{bmatrix} k_7 & k_7 \\ k_8 & -k_8 \end{bmatrix} \boldsymbol{\tau}_0 \tag{151}$$

A primeira linha desta expressão possui como entrada o produto entre k_7 e a soma dos torques de carga, ao passo que a segunda recebe o produto entre k_8 e a diferença entre eles. Substituindo-se (149) e (150) em (151), são verificadas as seguintes expressões:

$$\dot{u}_1 = k_5 u_2^2 + \frac{k_7 K_T (i_{a_r} + i_{a_l})}{n} - \frac{2k_7 f_m u_1}{rn^2} - \frac{2k_7 J_m \dot{u}_1}{rn^2} \tag{152}$$

$$\dot{u}_2 = k_6 u_1 u_2 + \frac{k_8 K_T (i_{a_r} - i_{a_l})}{n} - \frac{k_8 f_m u_2}{cn^2} - \frac{k_8 J_m \dot{u}_2}{cn^2} \tag{153}$$

Reorganizando estas equações de modo que as derivadas fiquem somente do lado esquerdo, e juntamente com as devidas simplificações, são obtidas:

$$\dot{u}_1 = \frac{k_5 u_2^2 rn^2}{rn^2 + 2J_m k_7} + \frac{k_7 K_T rn (i_{ar} + i_{al})}{rn^2 + 2J_m k_7} - \frac{2k_7 f_m u_1}{rn^2 + 2J_m k_7} \tag{154}$$

$$\dot{u}_2 = \frac{k_6 u_1 u_2 cn^2}{cn^2 + J_m k_8} + \frac{k_8 K_T cn (i_{ar} - i_{al})}{cn^2 + J_m k_8} - \frac{k_8 f_m u_2}{cn^2 + J_m k_8} \tag{155}$$

Como pode ser observado, as expressões que regem a dinâmica do robô passaram a depender dos efeitos relacionados aos atuadores. Uma vez que o modelo agora depende das correntes de armadura (i_a) dos atuadores, faz-se necessário equacionar o modo como elas variam. Substituindo (133) em (134) e rearranjando o resultado em função da derivada, obtém-se:

$$\frac{di_a}{dt} = \frac{V_a - i_a R_a - K_a \dot{\varphi}_m}{L_a} \tag{156}$$

Aplicando-se a relação (123), é encontrada uma expressão mais apropriada ao caso do Twil, dada por:

$$\frac{di_a}{dt} = \frac{V_a}{L_a} - \frac{i_a R_a}{L_a} - \frac{K_a \dot{\varphi}_l}{nL_a} \tag{157}$$

Parametrizando esta expressão em função de cada um dos atuadores, utilizando-se as relações (147) e (148), obtém-se:

$$\frac{di_{a_r}}{dt} = \frac{V_{a_r}}{L_a} - \frac{i_{a_r}R_a}{L_a} - \frac{K_a}{nL_a}\left(\frac{u_1}{r} + \frac{u_2}{2c}\right) \tag{158}$$

$$\frac{di_{a_l}}{dt} = \frac{V_{a_l}}{L_a} - \frac{i_{a_l}R_a}{L_a} - \frac{K_a}{nL_a}\left(\frac{u_1}{r} - \frac{u_2}{2c}\right) \tag{159}$$

A parcela do modelo que considera os efeitos de dinâmica, incluindo os atuadores, agora depende das expressões (158) e (159). Reorganizando estas expressões e colocando-as na forma matricial, é determinada a expressão que rege a dinâmica total do robô, dada por:

$$\begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \frac{di_{a_r}}{dt} \\ \frac{di_{a_l}}{dt} \end{bmatrix} = \begin{bmatrix} \frac{-2f_mk_7}{rn^2+2J_mk_7} & 0 & 0 & \frac{k_5rn^2}{rn^2+2J_mk_7} & \frac{k_7K_Trn}{rn^2+2J_mk_7} & \frac{k_7K_Trn}{rn^2+2J_mk_7} \\ 0 & \frac{-f_mk_8}{cn^2+J_mk_8} & \frac{k_6cn^2}{cn^2+J_mk_8} & 0 & \frac{k_8K_Tcn}{cn^2+J_mk_8} & \frac{-k_8K_Tcn}{cn^2+J_mk_8} \\ \frac{-K_a}{nrL_a} & \frac{-k_a}{2ncL_a} & 0 & 0 & \frac{-R_a}{L_a} & 0 \\ \frac{-K_a}{nrL_a} & \frac{k_a}{2ncL_a} & 0 & 0 & 0 & \frac{-R_a}{L_a} \end{bmatrix} \begin{bmatrix} u1 \\ u_2 \\ u_1u_2 \\ u_2^2 \\ i_{a_r} \\ i_{a_l} \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{L_a} & 0 \\ 0 & \frac{1}{L_a} \end{bmatrix} \begin{bmatrix} V_{a_r} \\ V_{a_l} \end{bmatrix} \tag{160}$$

Como pode ser observado, as entradas do modelo passaram a ser as tensões aplicadas aos enrolamentos dos atuadores. Tendo em vista que as correntes de armadura não dependem somente desta variável, uma vez que estão relacionadas às indutâncias de armadura, passa a ser necessário medi-las.

No entanto, conforme mencionado na subseção anterior, a constante de tempo elétrica (L_a/R_a) geralmente é suficientemente menor que a constante de tempo mecânica (J_e/f_e), de modo que o efeito da indutância de armadura (L_a) possa ser desprezado (FU; GONZALES; LEE, 1987). Deste modo, pode-se simplificar a expressão (134), retirando-se o efeito desta indutância (L_a), o que permite reescrever as expressões (158) e (159) como:

$$i_{a_r} = \frac{V_{a_r} - K_a\dot{\varphi}_{m_r}}{R_a} = \frac{V_{a_r}}{R_a} - \frac{K_a\dot{\varphi}_{l_r}}{nR_a} = \frac{V_{a_r}}{R_a} - \frac{K_a}{nR_a}\left(\frac{u_1}{r} + \frac{u_2}{2c}\right) \tag{161}$$

$$i_{a_l} = \frac{V_{a_l} - K_a\dot{\varphi}_{m_l}}{R_a} = \frac{V_{a_l}}{R_a} - \frac{K_a\dot{\varphi}_{l_l}}{nR_a} = \frac{V_{a_l}}{R_a} - \frac{K_a}{nR_a}\left(\frac{u_1}{r} - \frac{u_2}{2c}\right) \tag{162}$$

Isto permite eliminar as duas últimas equações de (160), tendo em vista que a corrente agora é determinada em função da tensão de armadura e da velocidade angular. As equações (152) e (153), entretanto, precisam ser ajustadas. Então, substituindo-se a soma e a subtração de (161) e (162), respectivamente em (152) e (153), têm-se:

$$\dot{u}_1 = k_5u_2^2 + \frac{k_7K_T}{n}\left(\frac{V_{a_r} + V_{a_l}}{R_a} - \frac{2K_au_1}{nrR_a}\right) - \frac{2k_7f_mu_1}{rn^2} - \frac{2k_7J_m\dot{u}_1}{rn^2} \tag{163}$$

$$\dot{u}_2 = k_6u_1u_2 + \frac{k_8K_T}{n}\left(\frac{V_{a_r} - V_{a_l}}{R_a} - \frac{K_au_2}{ncR_a}\right) - \frac{k_8f_mu_2}{cn^2} - \frac{k_8J_m\dot{u}_2}{cn^2} \tag{164}$$

Reorganizando estas equações de modo que as derivadas fiquem somente do lado esquerdo e, também, que os termos do lado direito fiquem em função das velocidades e tensões de armadura, obtém-se:

$$\dot{u}_1 = \frac{-2k_7u_1\left(\frac{K_TK_a}{R_a}+f_m\right)}{rn^2+2J_mk_7} + \frac{k_5rn^2u_2^2}{rn^2+2J_mk_7} + \frac{rnk_7K_T\left(\frac{V_{a_r}+V_{a_l}}{R_a}\right)}{rn^2+2J_mk_7} \quad (165)$$

$$\dot{u}_2 = \frac{-k_8u_2\left(\frac{K_TK_a}{R_a}+f_m\right)}{cn^2+J_mk_8} + \frac{k_6cn^2u_1u_2}{cn^2+J_mk_8} + \frac{cnk_8K_T\left(\frac{V_{a_r}-V_{a_l}}{R_a}\right)}{cn^2+J_mk_8} \quad (166)$$

Como pode-se observar, alguns termos nestas expressões são constantes, permitindo que elas sejam rearranjadas para (ALVES; LAGES; HENRIQUES, 2018a):

$$\dot{u}_1 = K_Au_1 + K_Bu_2^2 + K_C\left(V_{a_r}+V_{a_l}\right) \quad (167)$$

$$\dot{u}_2 = K_Du_2 + K_Eu_1u_2 + K_F\left(V_{a_r}-V_{a_l}\right) \quad (168)$$

Nestas expressões mais simples, os termos K_A a K_F são as constantes que representam o relacionamento entre as características cinemáticas e dinâmicas do robô, e os efeitos dos atuadores, definidas por:

$$K_A = \frac{-2k_7\left(\frac{K_TK_a}{R_a}+f_m\right)}{rn^2+2J_mk_7} \quad (169)$$

$$K_B = \frac{k_5rn^2}{rn^2+2J_mk_7} \quad (170)$$

$$K_C = \frac{rnK_Tk_7}{R_a(rn^2+2J_mk_7)} \quad (171)$$

$$K_D = \frac{-k_8\left(\frac{K_TK_a}{R_a}+f_m\right)}{cn^2+J_mk_8} \quad (172)$$

$$K_E = \frac{k_6cn^2}{cn^2+J_mk_8} \quad (173)$$

$$K_F = \frac{cnK_Tk_8}{R_a(cn^2+J_mk_8)} \quad (174)$$

Desta forma, o modelo dinâmico aumentado pode então ser representado matricialmente por:

$$\begin{cases} \dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{u} \\ \dot{\boldsymbol{u}} = \boldsymbol{H}_2(\boldsymbol{\beta})^{-1}(-\boldsymbol{f}_2(\boldsymbol{\beta},\boldsymbol{u}) + \boldsymbol{F}_2(\boldsymbol{\beta})\boldsymbol{V}_a) \end{cases} \quad (175)$$

onde os elementos $\boldsymbol{f}_2(\boldsymbol{\beta},\boldsymbol{u})$, $\boldsymbol{H}_2(\boldsymbol{\beta})$ e $\boldsymbol{F}_2(\boldsymbol{\beta})$ são matrizes que dependem das propriedades cinemáticas e dinâmicas do robô e, também, dos efeitos dos atuadores, definidas por:

$$\boldsymbol{H}_2(\boldsymbol{\beta}) = I \quad (176)$$

$$\boldsymbol{f}_2(\boldsymbol{\beta},\boldsymbol{u}) = -\begin{bmatrix} K_A & 0 & 0 & K_B \\ 0 & K_D & K_E & 0 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_1u_2 \\ u_2^2 \end{bmatrix} = -\boldsymbol{f}_2(\boldsymbol{u}) \quad (177)$$

$$\boldsymbol{F}_2(\boldsymbol{\beta}) = \begin{bmatrix} K_C & K_C \\ K_F & -K_F \end{bmatrix} = \boldsymbol{F}_2 \tag{178}$$

Este modelo pode ser reescrito na forma (60), substituindo-se as matrizes $\boldsymbol{H}(\boldsymbol{\beta})$, $\boldsymbol{f}(\boldsymbol{\beta}, \boldsymbol{u})$ e $\boldsymbol{F}(\boldsymbol{\beta})$ por $\boldsymbol{H}_2(\boldsymbol{\beta})$, $\boldsymbol{f}_2(\boldsymbol{\beta}, \boldsymbol{u})$ e $\boldsymbol{F}_2(\boldsymbol{\beta})$.

Assim como em (160), o modelo aumentado (175) possui como entradas as tensões aplicadas aos atuadores $\boldsymbol{V}_a = \begin{bmatrix} V_{a_r} & V_{a_l} \end{bmatrix}^T$. No entanto, as expressões (167) e (168) são notadamente mais simples que aquelas em (160), o que facilita o processo posterior de identificação dos parâmetros. Além disto, a consideração sobre as constantes de tempo do atuador permite que o efeito da indutância de armadura seja desprezado e, deste modo, a corrente de armadura passa a depender somente da tensão de armadura, da resistência de armadura e da força contra-eletromotriz.

4.3 Parametrização do Modelo para Identificação

Os parâmetros do modelo (113) dependem das massas e momentos de inércia do robô de uma forma complexa. Conforme LAGES (2017), mesmo com o uso de sistemas modernos de projeto auxiliado por computador (CAD), é difícil obter com precisão apropriada estes valores. O projeto mecânico geralmente não prevê variações em composições de materiais, mudanças de posicionamento de baterias, sensores e computador.

O modelo (175), que considera o efeito dos atuadores, torna este processo ainda mais complexo. Além dos parâmetros físicos, faz-se necessário conhecer as características elétricas e mecânicas dos atuadores. Contudo, é possível contornar a dificuldade de obter uma representação algébrica com as características de (175), por intermédio do procedimento denominado identificação de sistemas (AGUIRRE, 2007).

A partir do modelo dinâmico aumentado (175), pode-se separar a parcela que trata apenas os efeitos de dinâmica, dada por:

$$\dot{\boldsymbol{u}} = \begin{bmatrix} K_A & 0 & 0 & K_B \\ 0 & K_D & K_E & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_1 u_2 \\ u_2^2 \end{bmatrix} + \begin{bmatrix} K_C & K_C \\ K_F & -K_F \end{bmatrix} \boldsymbol{V}_a \tag{179}$$

Esta expressão pode, por conveniência, ser desmembrada em duas equações mais simples, dadas por:

$$\dot{u}_1 = K_A u_1 + K_B u_2^2 + K_C(V_{a_r} + V_{a_l}) \tag{180}$$

$$\dot{u}_2 = K_D u_2 + K_E u_1 u_2 + K_F(V_{a_r} - V_{a_l}) \tag{181}$$

Isto permite que sejam identificadas três constantes em cada equação, sendo este processo menos complexo do que determinar simultaneamente seis parâmetros (K_A a K_F). Portanto, discretizando-se as equações (180) e (181) por Euler, são obtidos os seguintes modelos recursivos (ALVES; LAGES; HENRIQUES, 2018a):

$$\begin{aligned} y_1(k+1) &= \dot{u}_1(k) \approx \frac{u_1(k+1) - u_1(k)}{T} \\ &= K_A u_1(k) + K_B u_2^2(k) + K_C(V_{a_r}(K) + V_{a_l}(k)) \end{aligned} \tag{182}$$

$$\begin{aligned} y_2(k+1) &= \dot{u}_2(k) \approx \frac{u_2(k+1) - u_2(k)}{T} \\ &= K_D u_2(k) + K_E u_1(k) u_2(k) + K_F(V_{a_r}(k) - V_{a_l}(k)) \end{aligned} \tag{183}$$

Visando aplicar um algoritmo de identificação do tipo Mínimos Quadrados Recursivo (RLS), deve-se dispor estas equações em um formato apropriado. As expressões (182) e (183) serão, então, reescritas na forma $y(k+1) = \phi^T(k)\theta + \omega(k+1)$, onde $\omega(k+1)$ é um ruído branco Gaussiano, representando as incertezas no modelo, $\phi(k)$ é o vetor de regressores e θ o vetor de constantes que devem ser identificados. Este formato, apropriado para a utilização do método RLS, é obtido reescrevendo-se (182) e (183) conforme:

$$y_1(k+1) = \underbrace{\begin{bmatrix} u_1(k) \\ u_2^2(k) \\ V_{a_r}(k) + V_{a_l}(k) \end{bmatrix}^T}_{\phi_1^T} \underbrace{\begin{bmatrix} K_A \\ K_B \\ K_C \end{bmatrix}}_{\theta_1} \tag{184}$$

$$y_2(k+1) = \underbrace{\begin{bmatrix} u_2(k) \\ u_1 u_2(k) \\ V_{a_r}(k) - V_{a_l}(k) \end{bmatrix}^T}_{\phi_2^T} \underbrace{\begin{bmatrix} K_D \\ K_E \\ K_F \end{bmatrix}}_{\theta_2} \tag{185}$$

onde $\phi_1(k)$ e $\phi_2(k)$ são os vetores de regressores e $\theta_1(k)$ e $\theta_2(k)$, os vetores de constantes para os modelos (182) e (183). Estes vetores são definidos por:

$$\phi_1(k) = \begin{bmatrix} u_1(k) \\ u_2^2(k), \\ V_{a_r}(k) + V_{a_l}(k) \end{bmatrix} \tag{186}$$

$$\phi_2(k) = \begin{bmatrix} u_2(k) \\ u_1 u_2(k) \\ V_{a_r}(k) - V_{a_l}(k) \end{bmatrix} \tag{187}$$

$$\theta_1(k) = \begin{bmatrix} K_A \\ K_B \\ K_C \end{bmatrix} \tag{188}$$

$$\theta_2(k) = \begin{bmatrix} K_D \\ K_E \\ K_F \end{bmatrix} \tag{189}$$

Deste modo, podem ser estimadas as constantes θ_1 e θ_2 apresentadas acima, utilizando-se as seguintes equações do RLS (AGUIRRE, 2007):

$$\hat{y}_i(k+1) = \phi_i^T(k)\hat{\theta}_i(k) \tag{190}$$

$$K_i(k) = \frac{P_i(k-1)\phi_i(k)}{1 + \phi_i^T(k)P_i(k-1)\phi_i(k)} \tag{191}$$

$$\hat{\theta}_i(k+1) = \hat{\theta}_i(k) + K_i(k)\left(y_i(k+1) - \hat{y}_i(k+1)\right) \tag{192}$$

$$P_i(k) = \left(I - K_i(k)\phi_i^T(k)\right) P_i(k-1) \tag{193}$$

Nas expressões acima, $\hat{\theta}_i(k)$ representa uma estimativa para $\theta_i(k)$, $\hat{y}_i(k+1)$ é uma estimativa para $y_i(k+1)$, $k_i(k)$ representa os ganhos do estimador e $P_i(k)$ a matriz de covariância do erro de estimação.

5 MÉTODOS DE CONTROLE

Neste capítulo são apresentados os métodos de controle sob estudo no presente trabalho. Embora o objetivo final resida na síntese de controladores de pose, serão também avaliadas estratégias de controle de velocidade. Esta abordagem fundamenta-se no fato de que a variável mais comumente controlada em cadeiras de rodas motorizadas é a velocidade (DING; COOPER, 2005) e, também, que robôs móveis comerciais geralmente possuem malhas de controle internas, responsáveis por rastrear as velocidades comandadas (MARTINS; SARCINELLI-FILHO; CARELLI, 2017).

Serão apresentados dois sistemas de controle de velocidade, um deles independente por junta e o outro baseado no modelo dinâmico do robô, linearizado por realimentação de estados. Posteriormente, serão apresentados dois sistemas de controle de pose. Cada um deles utilizará uma estrutura em cascata, composta de um controlador de pose não linear e um dos controladores de velocidade.

5.1 Controle de Velocidade Independente por Junta

O controle de velocidade de robôs pequenos, cujos atuadores são suficientemente potentes em relação às forças dinâmicas as quais são submetidos, pode ser realizado utilizado-se controladores convencionais (por exemplo, PID), no sentido de constituir servos de velocidade para cada uma das rodas. Nesta abordagem, as juntas são consideradas de forma independente para efeitos de controle, ou seja, como sistemas *Single-input, single-output* (SISO). Contudo, os efeitos relacionados ao acoplamento não-linear entre as juntas, bem como as perturbações mútuas produzidas pelos atuadores, são negligenciadas no projeto dos controladores e, portanto, são tratadas como distúrbios.

Esta estratégia, denominada controle independente por junta (SPONG; HUTCHINSON; VIDYASAGAR, 2005), será utilizada para controlar as velocidades das rodas do robô, conforme o diagrama da Figura 18. Pode ser observado que a referência do sistema de controle é dada pelo vetor $\boldsymbol{u}_r$ cujos componentes são as velocidades linear e angular desejadas. Desta forma, o erro entre o vetor de referências $\boldsymbol{u}_r$ e o vetor de velocidades do robô $\boldsymbol{u}$ é caracterizado por $\boldsymbol{e_u} = \boldsymbol{u}_r - \boldsymbol{u}$. A conversão do erro de velocidade da base ($\boldsymbol{e_u}$) para os erros de velocidade das rodas e_r e e_l é realizada pelo bloco Conversão Base para Juntas da Figura 18, utilizando as expressões (147) e (148). O rastreamento das velocidades das rodas é alcançado com o uso de controladores PI nas juntas. Assume-se que os atuadores são idênticos e, portanto, utilizarão o mesmo projeto de controlador.

Considerando o modelo dinâmico simplificado do atuador (141), que relaciona o deslocamento angular na carga à tensão de armadura e aplicando a ele o operador de La-

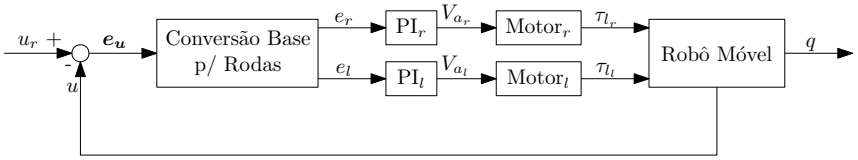


Figura 18: Controle de velocidade com PI independente por junta.

place (s), obtém-se:

$$G(s) = \frac{nK}{\tau_m s + 1} = \frac{nK/\tau_m}{s + 1/\tau_m} \tag{194}$$

onde a saída é a velocidade angular no eixo da roda. Tendo em vista que o sensor de deslocamento do Twil encontra-se no lado da carga e que o modelo dinâmico do motor será identificado, pode-se realizar a seguinte simplificação:

$$G(s) = \frac{K_G}{s + a} \tag{195}$$

onde K_G é uma constante de ganho da função de transferência de primeira ordem, que possui o polo em $s = -a$. Cada motor será comandado por um controlador PI, que possui como entrada o erro e_i e como saída a tensão de armadura V_{a_i}, conforme o diagrama da Figura 18. A função de transferência do controlador PI relacionado ao motor i é dada por:

$$C_i(s) = \frac{V_{a_i}(s)}{E_i(s)} = \frac{k_p s + k_i}{s} \tag{196}$$

Assim, a função de transferência em malha-fechada, de cada conjunto de motor e controlador, pode ser representada por:

$$H_i(s) = \frac{C_i(s)G_i(s)}{1 + C_i(s)G_i(s)} = \frac{K_G k_p s + K_G k_i}{s^2 + (K_G k_p + a)s + K_G k_i} \tag{197}$$

onde os polos de $H_i(s)$ determinam as características de desempenho da malha de velocidade i.

O projeto dos controladores será realizado aplicando-se o método da equação diofantina (BAZANELLA; GOMES DA SILVA JR., 2005, pp. 133-139). Tendo em vista que o sistema em malha-fechada é de segunda ordem, o polinômio que atende aos critérios de desempenho do projeto é caracterizado por $s^2 + 2\xi\omega_n + \omega_n^2$, onde ξ e ω_n representam, respectivamente, o coeficiente de amortecimento e a frequência natural desejados (LAGES, 2017).

O projeto dar-se-á, no entanto, em função do coeficiente de amortecimento e do tempo de assentamento (T_s), que para uma precisão de 2% é dado por (OGATA, 2010):

$$T_s = \frac{4}{\xi\omega_n} \tag{198}$$

Deste modo, os parâmetros de cada controlador PI são calculados por:

$$k_p = \frac{2\xi\omega_n - a}{K_G} \tag{199}$$

$$k_i = \frac{\omega_n^2}{K_G} \tag{200}$$

5.2 Controle de Velocidade da Dinâmica Linearizada

A estratégia de controle apresentada na proposta anterior é inapropriada para robôs móveis de alto desempenho, que são submetidos a altas velocidades ou massa elevada, uma vez que não pode garantir o rastreamento de referência sob estas condições (MARTINS; SARCINELLI-FILHO; CARELLI, 2017). Este é o caso, por exemplo, das cadeiras de rodas motorizadas. Torna-se, portanto, essencial considerar os efeitos de dinâmica aos quais o veículo será submetido. Propõe-se, nesta seção, uma estratégia de controle que leva em consideração estes efeitos, possuindo como vantagem a possibilidade de usar controladores lineares, conforme apresentado no diagrama de blocos da Figura 19.

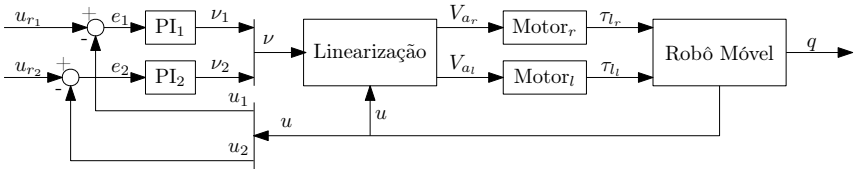


Figura 19: Controle de Velocidade da Dinâmica Linearizada.

Portanto, será aplicada uma linearização por realimentação de estados à parcela do modelo de configuração que representa os efeitos de dinâmica. Aplicando-se o procedimento da seção 3.4.2 à expressão (175), obtém-se a seguinte lei de controle por realimentação de estados:

$$\boldsymbol{V}_a \;=\; \boldsymbol{F}_2(\boldsymbol{\beta})^{-1}[\boldsymbol{H}_2(\boldsymbol{\beta})\boldsymbol{\nu} + \boldsymbol{f}_2(\boldsymbol{\beta}, \boldsymbol{u})] \tag{201}$$

onde $\boldsymbol{\nu} = \begin{bmatrix}\nu_1 & \nu_2\end{bmatrix}^T$ é o novo vetor de entradas de controle. Com a utilização desta realimentação em (179), obtém-se:

$$\dot{\boldsymbol{u}} \;=\; \boldsymbol{\nu} \tag{202}$$

A expressão (202) representa um sistema linear e desacoplado, o que significa que cada elemento de $\dot{\boldsymbol{u}}$ é controlado por um único elemento de $\boldsymbol{\nu}$ (LAGES, 2017). A função de transferência de cada um dos elementos de (202) é dada por:

$$G_i(s) = \frac{U_i(s)}{\mathcal{V}_i(s)} = \frac{1}{s} \tag{203}$$

Deste modo, cada canal de (202) pode ser estabilizado por um controlador PI. Conforme apresentado na Figura 19, cada PI possui como entrada o erro $e_i = u_{r_i} - u_i$ e como saída o esforço de controle ν_i. Então, a função de transferência do controlador PI relacionado ao subsistema i de (202) é dada por:

$$C_i(s) \;=\; \frac{\mathcal{V}_i(s)}{E_i(s)} = \frac{k_p s + k_i}{s} \tag{204}$$

Portanto, deve-se projetar um controlador para cada uma das variáveis em $\dot{\boldsymbol{u}}$, o que será realizado aplicando-se o método da equação diofantina, de forma análoga à proposta anterior.

A função de transferência em malha-fechada de cada subsistema de (202) é:

$$H_i(s) \;=\; \frac{C_i(s)G_i(s)}{1 + C_i(s)G_i(s)} = \frac{k_p s + k_i}{s^2 + k_p s + k_i} \tag{205}$$

onde o polinômio característico é de segunda ordem. Novamente, a expressão que atende aos critérios de projeto é caracterizada por $s^2 + 2\xi\omega_n + \omega_n^2$, onde ξ e ω_n representam, respectivamente, o coeficiente de amortecimento e a frequência natural desejados.

De forma análoga à proposta anterior, o projeto dar-se-á em função do coeficiente de amortecimento e do tempo de assentamento. Deste modo os parâmetros de cada controlador PI são calculados a partir de:

$$k_p = 2\xi\omega_n \quad (206)$$

$$k_i = \omega_n^2 \quad (207)$$

5.3 Controle de Pose com PI Independente por Junta

Conforme mencionado na Seção 3.4, foi demonstrado por Brocket (1982) que um sistema não holonômico sem deriva não pode ser assintoticamente estabilizado em um ponto arbitrário através de uma lei de controle suave e invariante no tempo. Este é o caso dos robôs móveis com acionamento diferencial. No entanto, podem ser utilizadas leis de controle não suaves para contornar estas restrições, como aquela apresentada na Seção 3.4.2.

O sistema de controle proposto nesta seção, baseia-se na estrutura em cascata apresentada na Figura 20. A malha de controle interna é composta pelo controlador de velocidade apresentado na seção 5.1, responsável por assegurar a convergência de $\boldsymbol{u}$ para $\boldsymbol{u}_r$. Portanto, pressupõe-se que, com a abordagem de controle independente por junta, seja possível rastrear as referências de velocidade comandadas ao robô e que, comandando $\boldsymbol{u}_r$, é possível deslocar o robô até uma pose desejada.

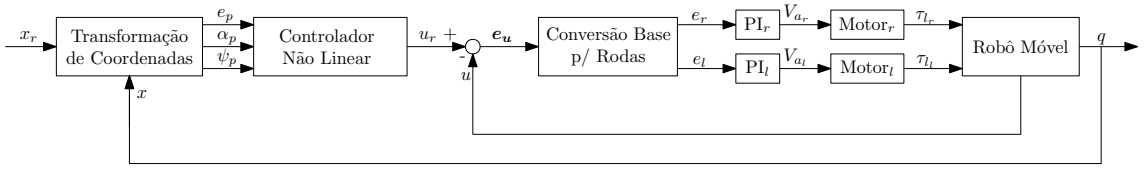


Figura 20: Controle de pose utilizando transformação descontínua e PI independente por junta.

A malha externa consiste no controlador de pose, representado pelos blocos Transformação de Coordenadas e Controlador Não Linear, que foi apresentado na seção 3.4.2. Este controlador segue a estrutura apresentada em (LAGES, 2017), é não linear e baseia-se na teoria de Lyapunov e na transformação do modelo para coordenadas polares.

Nesta malha, é tratada a parcela do modelo relacionada à cinemática do robô, caracterizada pela primeira linha de (175). Contudo, para aplicações de controle em geral, o interesse reside nas coordenadas de pose do robô, ou seja, $\boldsymbol{x} = \begin{bmatrix} x_c & y_c & \theta_c \end{bmatrix}^T$. Portanto, alguns componentes de $\boldsymbol{q}$ não são de interesse e podem ser ignorados. Desta forma, é utilizado o modelo cinemático de pose (27), caracterizado por $\dot{\boldsymbol{x}} = \boldsymbol{B}(\boldsymbol{x})\boldsymbol{u}$.

Considerando-se que a malha de controle de velocidade tenha sido propriamente parametrizada para ser estável e que o tempo de assentamento (T_s) seja suficientemente rápido em relação à malha de controle de pose, então a dinâmica (197) pode ser desprezada e a expressão (100) mantém-se válida. A malha de controle de pose recebe o vetor de coordenadas $\boldsymbol{x}_r = \begin{bmatrix} x_{cr} & y_{cr} & \theta_{cr} \end{bmatrix}^T$ como referência e utiliza a pose do robô ($\boldsymbol{x}$) como sinal de realimentação. Possui como saída o vetor de velocidades $\boldsymbol{u}_r$ cujas componentes devem ser rastreadas pela malha de controle interna.

Na síntese da lei de controle de pose, assim como na maioria dos controladores baseados na teoria de Lyapunov, assume-se que o sistema deve convergir para sua origem (LAGES, 2017). No entanto, é de interesse estabilizar o robô em uma pose arbitrária $\boldsymbol{x}_r = \begin{bmatrix} x_{cr} & y_{cr} & \theta_{cr} \end{bmatrix}^T$, o que pode ser alcançado com a transformação de coordenadas (101), que desloca a origem do novo sistema para a pose de referência ($\boldsymbol{x}_r$), dada por (BARROS, 2014, p. 80):

$$\overline{\boldsymbol{x}} \triangleq \begin{bmatrix} \cos\theta_{cr} & \sin\theta_{cr} & 0 \\ -\sin\theta_{cr} & \cos\theta_{cr} & 0 \\ 0 & 0 & 1 \end{bmatrix} (\boldsymbol{x} - \boldsymbol{x}_r)$$

onde $\overline{\boldsymbol{x}} = \begin{bmatrix} \overline{x}_c & \overline{y}_c & \overline{\theta}_c \end{bmatrix}^T$ representa o vetor de pose no novo sistema de coordenadas. Então, aplica-se a $\overline{\boldsymbol{x}}$ a transformação não suave (102):

$$\begin{aligned} e_p &= \sqrt{\overline{x}_c^2 + \overline{y}_c^2} \\ \psi_p &= atan2(\overline{y}_c, \overline{x}_c) \\ \alpha_p &= \overline{\theta}_c - \psi_p \end{aligned}$$

Por fim, são calculadas as referências de velocidade ($\boldsymbol{u}_r$) utilizando-se as leis de controle (107) e (108):

$$\begin{aligned} u_{r_1} &\triangleq -\gamma_1 e_p \cos\alpha_p \\ u_{r_2} &\triangleq -\gamma_2 \alpha_p - \gamma_1 \cos\alpha_p \sin\alpha_p + \gamma_1 \frac{\lambda_3}{\lambda_2} \cos\alpha_p \frac{\sin\alpha_p}{\alpha_p} \psi_p \end{aligned}$$

onde os valores de λ_i e γ_i devem ser positivos para que o sistema de controle de pose (Figura 20) seja estável. O detalhamento desta lei de controle, bem como a prova de estabilidade, foram apresentados na seção 3.4.2.

5.4 Controle de Pose com Linearização da Dinâmica

As considerações da proposta anterior não são efetivas para robôs móveis de alto desempenho, que são submetidos a altas velocidades ou variações elevadas de carga. Nestes casos, é necessário que sejam empregados modelos matemáticos que considerem os efeitos da dinâmica do robô.

O sistema de controle proposto nesta seção, baseia-se na estrutura em cascata apresentada na Figura 21. No controlador de velocidade, proposto nesta seção, responsável por assegurar a convergência de $\boldsymbol{u}$ para $\boldsymbol{u}_r$, são considerados estes efeitos. Este controlador foi apresentado na seção 5.2 e utiliza uma realimentação de estados para linearizar a parcela do modelo relacionada à dinâmica do robô.

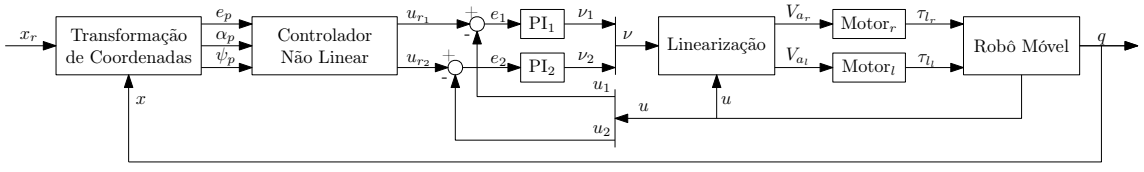


Figura 21: Controle de pose utilizando transformação descontínua e linearização da dinâmica.

Similarmente à proposta na seção 5.3, a malha externa consiste no controlador de pose, representado pelos blocos Transformação de Coordenadas e Controlador Não Linear, que foi apresentado na seção 3.4.2. Nesta malha, é tratada a parcela do modelo relacionada à cinemática do robô, caracterizada pela primeira linha de (175), mais especificamente o modelo cinemático de pose.

Com a linearização por realimentação aplicada à malha de controle interna e a utilização somente das coordenadas de pose, nesta proposta é considerado o modelo dinâmico de pose (70):

$$\begin{cases} \dot{\boldsymbol{x}} &= \boldsymbol{B}(\boldsymbol{x})\boldsymbol{u} \\ \dot{\boldsymbol{u}} &= \boldsymbol{\nu} \end{cases}$$

Novamente, considerando-se que o sistema de controle de velocidade tenha sido apropriadamente parametrizado para ser estável e que o tempo de assentamento (T_s) seja suficientemente rápido em relação à malha de controle de pose, a dinâmica (205) pode ser desprezada. Desta forma, a expressão (100) mantém-se válida.

A aplicação das transformações de coordenadas (101) e (102) bem como o cálculo das leis de controle (107) e (108) são idênticos aos da proposta anterior e, portanto, não serão detalhados novamente aqui.

Por fim, para que o sistema de controle de pose completo (Figura 21) seja estável, os valores de λ_i e γ_i devem ser positivos. O detalhamento desta lei de controle, bem como a prova de estabilidade, foram apresentados na seção 3.4.2.

6 IMPLEMENTAÇÃO DE *HARDWARE* E *SOFTWARE*

Neste capítulo, serão apresentadas as ferramentas utilizadas no desenvolvimento deste trabalho. Inicialmente, serão caracterizadas as interfaces responsáveis pela leitura dos sensores e acionamento dos motores. Em seguida, serão detalhados os aspectos da implementação de controladores, utilizando o pacote `ros_control` (MEEUSSEN, 2018a), do *Robot Operating System* (ROS) (QUIGLEY et al., 2009).

6.1 Estrutura de Acionamento dos Motores

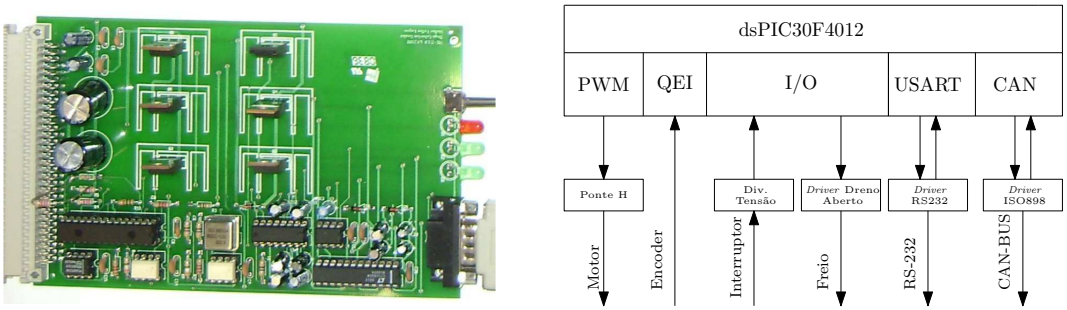
O robô móvel Twil possui atuadores compostos por motores de corrente contínua, com ímãs permanentes e sistemas de transmissão. Estes atuadores serão comandados pela tensão de armadura. Portanto, são necessárias estruturas eletrônicas capazes de fornecer aos enrolamentos dos motores os valores de tensão calculados pelos controladores. Para este propósito, optou-se por utilizar conversores CC-CC do tipo ponte completa (*full-bridge*) (MOHAN; UNDELAND; ROBBINS, 1995, p. 387), também denominados ponte-H.

Por outro lado, o Twil possui *encoders* incrementais de quadratura (EIQ), acoplados aos eixos das rodas. Este tipo de sensor permite a medição do deslocamento e do sentido de giro do eixo ao qual é fixado. O EIQ consiste em um disco dotado de ranhuras, pelas quais ocorre a passagem da luz, com dois sensores ópticos acoplados, posicionados com uma defasagem de noventa graus elétricos entre si. A decodificação em quadratura permite a obtenção de uma resolução quatro vezes maior que o número de ranhuras do disco (BORENSTEIN; EVERETT; FENG, 1996).

Neste trabalho, o acionamento dos motores e a decodificação em quadratura dos *encoders* serão realizados por placas *Actuator Interface Card* (AIC), desenvolvidas na UFRGS para o controle das juntas de robôs manipuladores. Estas placas foram inicialmente utilizadas por LAGES; BRACARENSE (2003) e LIMA et al. (2004) no *retrofitting* de um robô ASEA IRB6 e, posteriormente, por SANTINI (2009), no *retrofitting* do robô JANUS.

A arquitetura de *hardware* da AIC (Figura 22b) baseia-se no microcontrolador dsPIC 30F4012 (MICROCHIP, 2010) e dispositivos periféricos (*drivers*, *transceptores* e conversores de tensão). Maiores detalhes são apresentados em SANTINI (2009, pp. 57-74).

Originalmente, a comunicação de controle entre as AICs e o computador (*Host PC*) era realizada somente por meio de um barramento CAN, conforme a topologia apresentada na Figura 23, sendo a interface RS-232 empregada apenas a tarefas de *debug*. No entanto, para poder utilizá-las com computadores portáteis, optou-se por modificar o *firmware* desenvolvido por SANTINI (2009), de modo a possibilitar que a comunicação dos sinais de controle fosse executada, também, pela interface RS-232, conforme a topologia apresentada na Figura 24. Assim, podem ser utilizados conversores USB-Serial (juntamente com transceptores RS-232) para comunicação com as placas em computado-



(a) (b)

Figura 22: Fotografia da placa de circuito impresso AIC, na versão 2.1.9, em (a), e o diagrama de blocos do projeto em (b).
Fonte: SANTINI (2009)

res onde não há possibilidade de utilizar uma interface CAN.

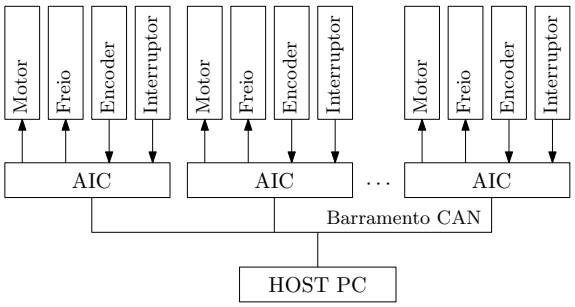


Figura 23: Topologia de comunicação com as placas AIC.
Fonte: SANTINI (2009, p. 57)

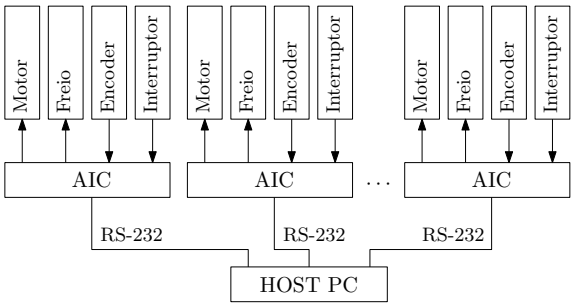


Figura 24: Topologia de comunicação com as placas AIC.

De forma similar, foi desenvolvida uma biblioteca de comunicação (`aic_lib`) em linguagem C++, para o envio de comandos e leitura das respostas das placas, a partir do *PC Host*. Esta biblioteca possibilita a comunicação com as AIC por meio de interfaces CAN e RS-232, utilizando-se o protocolo `aic_net`, apresentado na tabela 3. Este protocolo foi desenvolvido, inicialmente, para ser executado através do protocolo CAN (BOSCH, 1991), na versão 2.0A. Deste modo, o campo *identifier* de um *frame* CAN, que possui 11 *bits* na versão 2.0A, é utilizado para enviar o comando e o endereço da placa que deverá ser comandada. Os comandos que possuem informações adicionais, utilizam o campo *Data Length Code* (DLC) para definir o tamanho do *payload*, que será enviado no campo de dados (*Data Field*).

Tabela 3: Protocolo `aic_net`
Fonte: SANTINI (2009, p. 68)

Bits	Descrição	
10-5	Comando	Dados
	000000 = Reservado	0 *bytes*
	000001 = Reinicia AIC	0 *bytes*
	000010 = Desligar motor	0 *bytes*
	000011 = Aplicar freio	0 *bytes*
	010000 = Aplicar tensão no motor	8 *bytes*
	100000 = Ler *status*	8 *bytes*
	110000 = Desligar freio	0 *bytes*
	110001 = Ligar motor	0 *bytes*
4-0	Endereço da AIC	
	00000 = Reservado	
	00001 = AIC 01	
	00010 = AIC 02	
	⋮	
	11111 = AIC 31	

Quando é utilizada a interface RS-232, os *bits* de endereço não são enviados, uma vez que se tem uma comunicação ponto-a-ponto, dispensando o endereçamento. Além disto, a implementação utilizando o CAN possui uma taxa de transmissão máxima de 1Mbit/s, enquanto a versão por RS-232 utiliza uma taxa máxima de 115.2Kbit/s. Embora seja possível utilizar taxas maiores com RS-232, os transceptores empregados nas placas limitam esta taxa aos valores referidos. Utilizou-se, para ambos os meios físicos, as funções dos *drivers* disponíveis no *Kernel* 4.4.0-119 do Linux, onde a comunicação CAN baseou-se no driver SocketCan. Maiores detalhes sobre os protocolos `aic_net` e CAN 2.0A podem ser encontrados em SANTINI (2009) e BOSCH (1991).

A biblioteca foi implementada em classes, permitindo que sejam expostas ao usuário apenas funções de alto-nível, que executam chamadas para funções mais complexas, definidas nas classes relacionadas aos dispositivos periféricos. Deste modo, foram desenvolvidas as seguintes classes:

- `aic_serial`: implementa as funções de configuração, leitura e escrita com o *driver* de comunicação serial do Linux.

- `aic_can`: implementa as funções de configuração, leitura e escrita com o *driver* SocketCan do Linux.

- `aic_com`: esta classe é utilizada para realizar a troca de informações com as placas, através dos métodos `send_command()` e `get_status()`. O acesso ao meio de comunicação (RS-232 ou CAN) é realizado a partir de um objeto `aic_serial` ou `aic_can`, inicializado conforme os parâmetros recebidos no construtor da classe.

- `aic_brake`: esta classe permite a manipulação do sistema de freio ligado à placa, através dos métodos `brake_apply()` e `brake_release()`.

- `aic_motor`: esta classe permite a manipulação do atuador ligado à placa, através dos métodos `motor_off()`, `motor_on()` e `set_motor_voltage()`.

- `aic_sensors`: esta classe permite a leitura do deslocamento da roda, medido pelo *encoder*, e do estado da entrada digital. Embora estes sejam os únicos sensores utilizados no momento, optou-se por implementar uma classe, pois há a perspectiva de medição de força e de corrente elétrica na próxima versão da placa. Contudo, foi implementado somente o método `read_displacement()`, que fornece o estado de ambos os sensores utilizados.

- `aic`: esta é a classe mais geral, onde são definidas as funções de alto nível, disponíveis no protocolo `aic_net`. Para isto, são instanciados objetos das classes `aic_sensors`, `aic_brake` e `aic_motor`, cujos métodos são utilizados para a execução das funções de alto nível. Estes objetos recebem por referência um objeto do tipo `aic_com`, inicializado no construtor da classe `aic`, que é utilizado para as tarefas de comunicação.

Este tipo de implementação simplifica o processo de integração, necessitando-se somente incluir a *header* `aic.h` e instanciar um objeto `aic`. Este objeto recebe como argumento uma *struct* do tipo `aic_comm_config_t`, com os parâmetros de comunicação da respectiva placa. Por fim, o diagrama de classes é apresentado na Figura 25 e um exemplo de código de testes é demonstrado na Listagem 6.1.

Listagem 6.1: *Software* de teste da AIC, utilizando o barramento RS232.

```
#include <iostream>
#include "aic.h"
int main(int argc, char *argv[])
{
  aic_comm_config_t param;
  param.aic_comm_device = rs232;
  param.aic_serial_port = "/dev/ttyUSB0";
  aic * board = new aic(param);
  if (board->status_ok()){
    board->brake_off();
    board->set_motor_voltage(5);
    usleep(1000000);
    aic_displacement_msg dp = board->read_displacement_sensors();
    std::cout << "Meas.: " << dp.joint_displacement << std::endl;
    board->brake_on();
  }
  board->~aic();
  return 0;
}
```

6.2 Estrutura de Controle Utilizando o ROS

As estruturas tradicionais do ROS, como nodos, mensagens, tópicos, serviços e ações (QUIGLEY; GERKEY; SMART, 2015), embora adequadas para tarefas de alto nível, não são apropriadas para a implementação de controladores cuja execução deva ocorrer em tempo-real. Isto se deve ao fato de que os nodos não operam em tempo-real, e os tópicos são mecanismos de comunicação, para os quais não há garantia temporal na entrega das mensagens (LAGES, 2016).

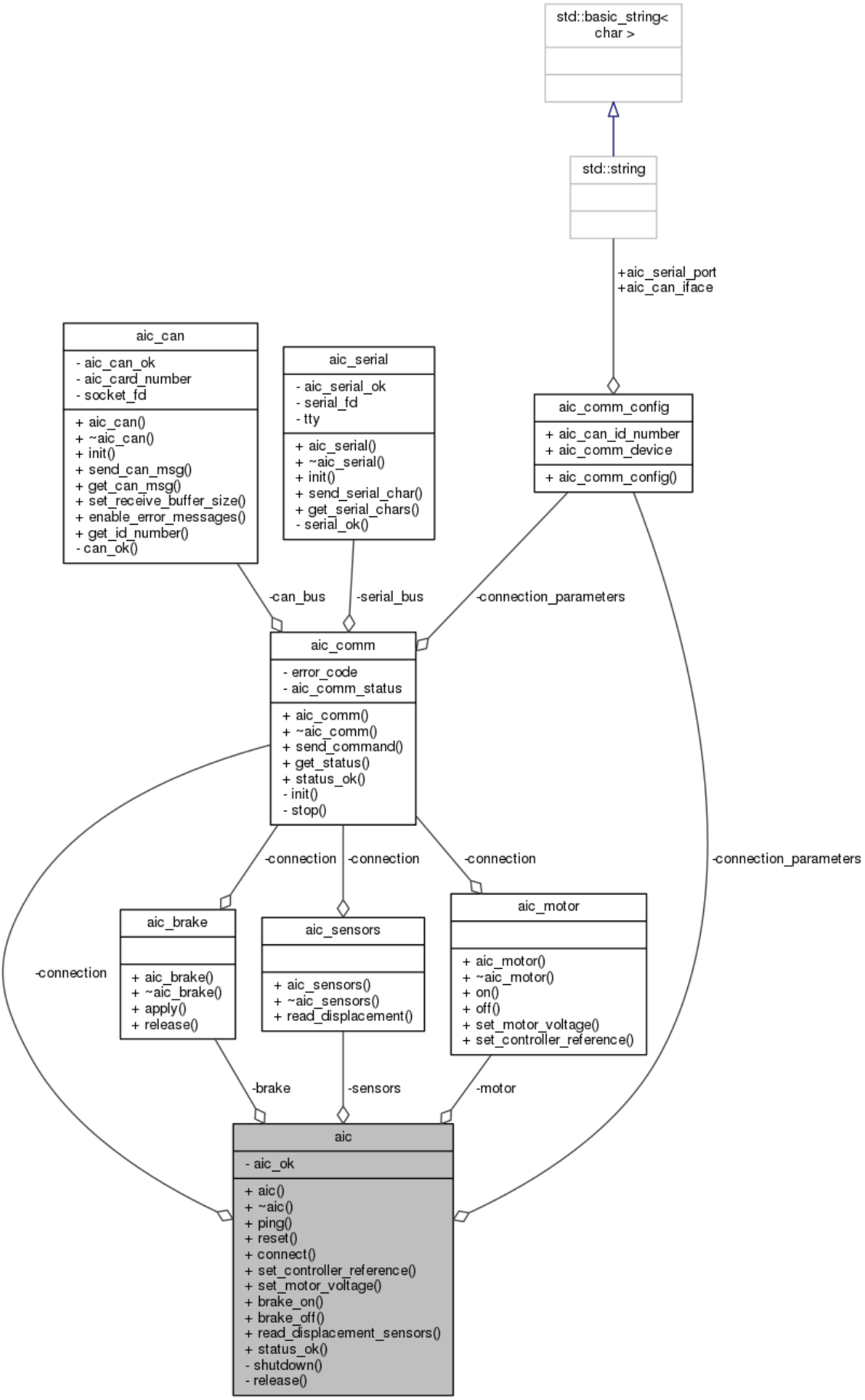


Figura 25: Diagrama de classes da `aic_lib`.

Para contornar estas limitações, pode-se utilizar o *framework* `ros_control` (MEEUSSEN, 2018a), fornecido como um pacote do ROS, que provê um conjunto de ferramentas para implementação e gerenciamento de controladores, bem como para a execução deles em tempo-real. Neste *framework*, os controladores são implementados em uma abordagem *realtime-safe*, e a comunicação é baseada em chamadas de funções (LAGES, 2016).

Somente essas características não são suficientes para garantir a execução em tempo-real, uma vez que o ROS depende do sistema operacional, sobre o qual é executado. Para alcançar resultados de tempo-real, devem ser modificados os limites de usuário, em relação às políticas do escalonador (*scheduler*) do sistema e, também, realizar a aplicação de um *patch* ao *Kernel* padrão do *Linux*, como o `PREEMPT_RT` *kernel patch*, para torná-lo totalmente preemptivo. Em LAGES (2016, pp. 681-683) são detalhados os processos de aplicação do `PREEMPT_RT` *kernel patch* e de ajuste dos limites de usuário.

No *framework* `ros_control`, representado pela Figura 26, os controladores e o *hardware* do robô, que pode ser físico ou simulado, são definidos em camadas (*layers*) distintas. O acesso aos sensores e atuadores é realizado através de classes, que implementam a classe-base `hardware_interface::RobotHW`. Em contrapartida, se o objetivo for a simulação no *software* Gazebo, deve ser utilizada a classe `RobotHWSim`, que é derivada da `hardware_interface::RobotHW`, e implementa as interfaces com o simulador. Os objetos da classe `hardware_interface::RobotHW` são acessados pelos controladores, através de *hardware interfaces*[1].

O gerenciador de controladores (*Controller Manager*) é a entidade responsável por gerenciar o ciclo de vida dos controladores e por lidar com conflitos entre eles (LAGES, 2016). As *hardware interfaces*, por outro lado, compõem a camada intermediária, entre os controladores e os objetos da classe `RobotHW`, denominada *Hardware Resource Interface Layer* (HRIL).

A implementação de um controlador, sob esta abstração, não depende do *hardware* empregado no robô, mas apenas dos tipos de *hardware interfaces*, registrados pela classe que implementa a `RobotHW`. Deste modo, se houver modificação na configuração de *hardware* do robô e se as mesmas *hardware interfaces* forem registradas na classe que implementa a `RobotHW`, não haverá necessidade de modificar o *software* de controle. Maiores detalhes sobre os pacotes e funcionalidades do *framework* são apresentados em MEEUSSEN (2018a), MEEUSSEN (2018b) e ROS (2018a).

A implementação de controladores apropriados para serem executados em tempo-real, utilizando o *framework* `ros_control`, foi demonstrada por MACIEL (2014), BARROS (2014), JOSEPH (2015), LAGES (2016) e LAGES (2017). Neste conjunto de trabalhos, são detalhadas as características do *framework*, bem como os aspectos de implementação dos controladores. São também apresentados os conceitos de modelagem utilizando o *Unified Robot Description Format* (URDF) (SUCAN, 2018), que permite descrever as propriedades cinemáticas e dinâmicas de robôs, e possibilita a simulação do modelo em *softwares* como o Gazebo. Contudo, em nenhum deles é realizado o controle de um robô real, sendo a avaliação dos controladores realizada por simulação, através do Gazebo.

No entanto, a diferença entre a utilização do *framework* para simulação ou para o robô real, encontra-se somente na implementação da classe `RobotHW`. Na simulação com o Gazebo, a base de tempo é gerada pelo simulador, e a sincronização com o ROS, mais precisamente com o gerenciador de controladores, é realizada utilizando-se o plugin `GazeboRosControlPlugin`. Por outro lado, quando é utilizado um robô real, deve-

[1] Os tipos de *hardware interfaces* fornecidas no pacote podem ser verificados em MEEUSSEN (2018a).

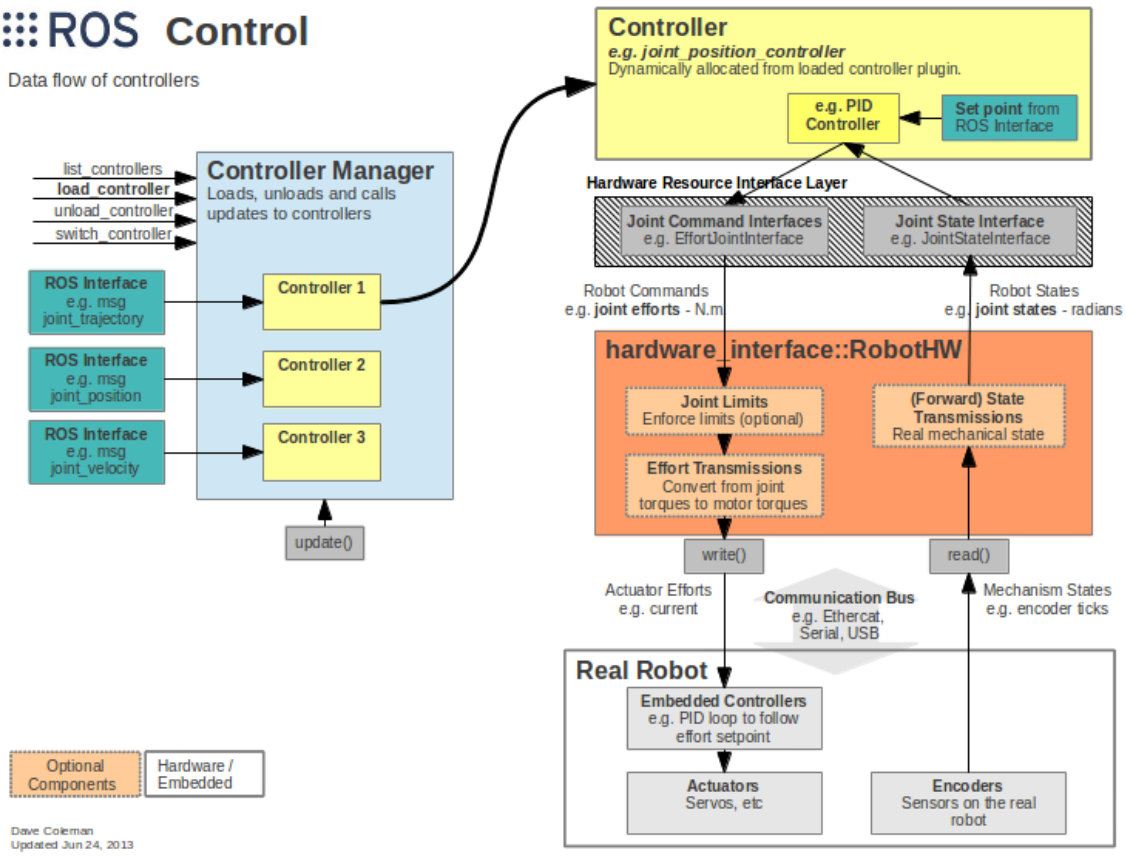


Figura 26: Visão geral do *framework ros_control*.
Fonte: MEEUSSEN (2018a)

se implementar um nodo que será responsável por gerar a sincronização com o *hardware* do robô, e com o gerenciador de controladores. Esta discussão será retomada na Seção 6.2.3.

Os tópicos relacionados à abstração de *hardware*, para um robô real, são apresentados na seção 6.2.1, e o laço de tempo-real na seção 6.2.3. Maiores detalhes sobre a integração com o Gazebo são apresentados em MACIEL (2014), BARROS (2014) e LAGES (2016).

6.2.1 Abstração do *hardware*

A classe `hardware_interface::RobotHW` abstrai a interface entre o ROS e o *hardware* do robô (Figura 26), seja ele simulado ou real. No caso de simulação com o Gazebo, esta classe é derivada para a classe `RobotHWSim`, que implementa as interfaces com o simulador, e por sua vez é derivada da classe `RobotHW`. Considerando-se um robô real, a comunicação com o *hardware* do robô deve ser realizada por uma classe que implemente a `RobotHW`. Em ambos os casos, a comunicação com o simulador ou com o *hardware* do robô real, é realizada invocando-se os os métodos virtuais `read()` e `write()` da classe `RobotHW`, implementados nas classes derivadas.

No caso do robô móvel TWIL, foi criado o pacote `twil_hardware`, composto pela classe `TwilRobotHW`, que implementa a `RobotHW`, e possui interfaces com os *encoders* e atuadores. A leitura dos dados dos sensores e a aplicação dos comandos aos atuadores, são realizadas através da `aic_lib`, o que será ilustrado mais adiante. Uma versão simplificada do arquivo de *headers*, da classe `TwilRobotHW`, é demonstrada no Apêndice A.

Os dados adquiridos no método `read()`, da classe `TwilRobotHW`, são armazenados em variáveis privadas, para serem utilizados posteriormente. Similarmente, são utilizadas variáveis privadas, para os comandos a serem aplicados aos atuadores, no método `write()`, da classe derivada. Com isso, os controladores não se comunicam diretamente com o *hardware* físico do robô. O acesso dos controladores a estas variáveis é realizado a partir de *handles*. Neste trabalho, serão consideradas somente as interfaces de junta, que refletem o conjunto de sensores e atuadores utilizados pelo Twil. Com este tipo de arranjo, utilizam-se dois tipos de *handles*:

- `JointStateHandle`: este tipo de *handle* é utilizado para a leitura dos estados de uma única junta. A classe `JointStateHandle` abstrai a leitura dos sensores de posição, velocidade e esforço, ligados à uma junta. Os valores dos estados são fornecidos através dos métodos `getPosition()`, `getVelocity()` e `getEffort()`, que utilizam ponteiros para acessar as respectivas variáveis, no objeto da classe `RobotHW`.

- `JointHandle`: este tipo de *handle* é utilizado não somente para a leitura dos estados, mas também para aplicar comandos, a uma única junta. Por este motivo, implementa a classe `JointStateHandle`, que permite a leitura dos sensores e abstrai a escrita nos atuadores. Além dos métodos herdados da `JointStateHandle`, a atuação é feita através da função `setCommand()`, que utiliza um ponteiro para acessar a variável de comando no objeto da classe `RobotHW`.

A classe que implementa o `RobotHW` deve registrar, no caso de interfaces de junta, um objeto da classe `JointStateHandle` para as variáveis relacionadas aos sensores de cada uma das juntas. Similarmente, um objeto da classe `JointHandle` deve ser registrado para a variável relacionada ao atuador de cada uma das juntas, permitindo que sejam recebidos os comandos, que serão aplicados a eles.

Um *handle* é um mapeamento para uma única junta. No entanto, é de interesse uma interface que forneça um mapeamento para um conjunto de juntas. São utilizadas, para esta finalidade, as *hardware interfaces*, que podem ser de dois tipos:

- `JointStateInterface`: *hardware interface* concebida para permitir a leitura dos estados de um *array* de juntas, a partir dos *handles* registrados.
- `JointCommandInterface`: *hardware interface* concebida para permitir que sejam comandadas as saídas de um *array* de juntas, a partir dos *handles* registrados.

Basicamente, a classe `JointStateInterface`, implementa um *template* da classe `HardwareResourceManager`, com um argumento do tipo `JointStateHandle`. Similarmente, a classe `JointCommandInterface` implementa um *template* da classe `HardwareResourceManager`, porém com um argumento do tipo `JointHandle`.

A classe que implementa o `RobotHW` deve, portanto, registrar, em um objeto do tipo `JointStateInterface`, todos os objetos do tipo `JointStateHandle`. O processo de registro dos *handles* dos sensores e da `JointStateInterface`, utilizados na classe `TwilRobotHW`, são apresentados na Listagem 6.2.

Listagem 6.2: Registro dos objetos das classes `StateHandle` e `StateInterface` na classe `TwilRobotHW`.

```
hardware_interface::JointStateHandle state_handle_left(
          "left_wheel_joint", &pos[0], &vel[0], &eff[0]
          );
joint_state_interface.registerHandle(state_handle_left);

hardware_interface::JointStateHandle state_handle_right(
          "right_wheel_joint", &pos[1], &vel[1], &eff[1]
          );
joint_state_interface.registerHandle(state_handle_right);

registerInterface(&joint_state_interface);
```

Analogamente, devem ser registrados todos os objetos do tipo `JointHandle`, em um objeto do tipo `JointCommandInterface`. Como pode ser observado, a nomenclatura aplicada à classe `JointCommandInterface` é genérica, e o comando não evidencia a grandeza física que está sendo manipulada. Afim de tornar mais evidente qual o tipo de variável sendo atuada, foram implementadas as seguintes derivações desta classe:

- `EffortJointInterface`: utilizada para comandar juntas baseadas em esforço;
- `VelocityJointInterface`: utilizada para comandar juntas baseadas em velocidade;
- `PositionJointInterface`: utilizada para comandar juntas baseadas em posição;

Estas classes derivadas, na maioria das vezes, são utilizadas para comandar juntas com servos. Contudo, o Twil possui atuadores comandados em tensão elétrica e, portanto, a classe `JointCommandInterface` foi derivada para `VoltageJointInterface`, no pacote `twil_hardware_interfaces`, criado para esta finalidade. O processo de registro dos objetos da classe `JointCommandInterface` e dos *handles* relacionados aos atuadores, utilizados na classe `TwilRobotHW`, são apresentados na Listagem 6.3.

Listagem 6.3: Registro dos objetos das classes `JointHandle` e `CommandInterface` na classe `TwilRobotHW`.

```
hardware_interface::JointHandle
 eff_handle_left(joint_state_interface.getHandle("left_wheel_joint"),
    &cmd[0]);
voltage_interface.registerHandle(eff_handle_left);

hardware_interface::JointHandle
 eff_handle_right(joint_state_interface.getHandle("right_wheel_joint"),
    &cmd[1]);
voltage_interface.registerHandle(eff_handle_right);

registerInterface(&voltage_interface);
```

Para que um controlador acesse as variáveis do objeto da classe `RobotHW`, ele recebe o endereço do objeto da *hardware interface* registrada. Posteriormente, devem ser obtidos deste objeto os *handles* para as juntas de interesse, baseando-se no nome utilizado no processo de registro. Nas listagens 6.2 e 6.3, os nomes utilizados foram *left_wheel_joint* e *right_wheel_joint*.

Por fim, a aquisição das medidas dos sensores das juntas é realizada pelo método `TwilRobotHW::read()`, e as variáveis de comando são aplicadas com o método `TwilRobotHW::write()`. Um exemplo simplificado de leitura dos sensores, utilizando a `aic_lib`, é apresentado na Listagem 6.4, e um exemplo de atuação é apresentado na Listagem 6.5.

Listagem 6.4: Implementação do método `read()` da classe `RobotHW`.

```
void TwilRobotHW::read(const ros::Time &time, const ros::Duration &
    period)
{
  for (int i = 0; i < 2; i++)
  {
    joint_sensor[i] = jnt_hw[i]->read_displacement_sensors();
    pos[i] += joint_sensor[i].joint_displacement;
    vel[i] =  joint_sensor[i].joint_displacement/period.toSec();
  }
}
```

Listagem 6.5: Implementação do método `write()` da classe `RobotHW`.

```
void TwilRobotHW::write(const ros::Time &time, const ros::Duration &
    period)
{
  for (int i = 0; i < 2; i++) joint_hw[i]->set_motor_voltage(cmd[i]);
}
```

6.2.2 Implementação de Controladores

A implementação de controladores no ROS apresenta algumas particularidades, do ponto de vista de sistemas de controle. A primeira delas refere-se à nomenclatura, uma vez que um controlador no ROS não representa, necessariamente, um controlador na teoria de sistemas de controle. Neste *framework*, um controlador é um *plugin* para o *controller manager*, que implementa a classe `Controller` (BARROS, 2014).

Tipicamente, um controlador atua sobre uma junta, por meio de um objeto que expõe uma `JointCommandInterface`, e obtém dados dos sensores, por meio de um objeto

que expõe uma `JointStateInterface`. No entanto, um controlador em ROS pode executar funções diferentes daquelas de um controlador em um sistema de controle. Pode ser utilizado como exemplo o `JointStateController`, cuja nomenclatura remete a um controlador de junta, no espaço de estados, mas trata-se apenas de um *publisher* para as variáveis de junta, obtidas através de uma interface do tipo `JointStateInterface` (LAGES, 2016). Outro exemplo é o `ImuSensorController`, que é utilizado para ler as medidas de um sensor inercial, utilizando-se uma interface do tipo `ImuSensor Interface`, e publicá-las em um tópico, de modo análogo ao controlador `JointState Controller`.

Além disto, aparentemente, a infraestrutura do ROS foi concebida para controladores SISO. Por outro lado, leis de controle avançadas para robôs móveis, são intrinsecamente do tipo *multiple-input, multiple-output* (MIMO), e não podem ser decompostas em um conjunto de leis do tipo SISO, tendo em vista que em um sistema MIMO, todas as saídas são fornecidas no mesmo instante de tempo, o que dependeria de algum tipo de mecanismo de sincronização, para que um conjunto de controladores SISO apresentasse este resultado (BARROS, 2014; LAGES, 2016).

Do ponto de vista da teoria de controle digital, o paradigma é o de um controlador em tempo contínuo, implementado por um computador digital, com uma taxa de amostragem rápida o suficiente para que sejam negligenciados os efeitos de discretização (LAGES, 2016). Esta abordagem é apropriada para a implementação dos controladores não lineares apresentados no capítulo anterior (SLOTINE; LI, 1991).

Na Figura 26 é apresentada a arquitetura do *framework* `ros_control`. Nesta arquitetura, os controladores são *plugins* carregados pelo gerenciador de controladores (*Controller Manager*), que pode, também, descarregar, ativar e desativar os controladores. Basicamente, quatro funções virtuais da classe `controller`, implementadas nas classes derivadas, são invocadas pelo *Controller Manager*. Estas funções são apresentadas em vermelho no diagrama de estados da Figura 27.

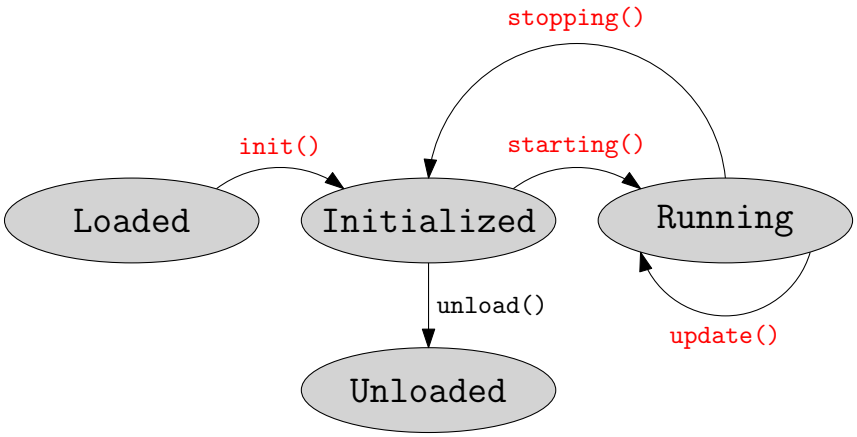


Figura 27: Diagrama de estados de um controlador.

A função `init()`, a primeira delas, é invocada quando o controlador é carregado pelo *Controller Manager* e não possui limite de tempo para ser executada, uma vez que está fora do laço de tempo-real (JOSEPH, 2015). Geralmente, é neste método que são obtidos os parâmetros do *Parameter Server* e configurados os *handles* para as juntas do robô, conforme apresentado no exemplo da Listagem 6.6 (ROS, 2018b).

Listagem 6.6: Exemplo de utilização do método `init()` de um controlador.

```
    bool NewController::init(twil_hardware_interfaces::
   VoltageJointInterface* hw, ros::NodeHandle &n)
    {
    // get joint name from the parameter server
    std::string joint_name;
    if (!n.getParam("left_wheel_joint", joint_name)){
    ROS_ERROR("Could not find joint name");
    return false;
    }
    // get the joint object to use in the realtime loop
    joint_ = hw->getHandle(joint_name);  // throws on failure
    sub_ = n.subscribe("/controller/command", 1,
    &NewController::commandCB,this);
    return true;
    }
%
```

Na Listagem 6.6, as variáveis `joint_` e `sub_` são objetos dos tipos `JointHandle` e `ros::Subscriber`. Retornando à sequência da Figura 27, o método `starting()` é executado dentro do laço de tempo-real, apenas uma vez, após o controlador ser carregado (JOSEPH, 2015). Este método faz parte da filosofia do ROS de poder carregar e descarregar controladores, enquanto o robô está em funcionamento e, portanto, nem sempre é implementado.

A função `update()` é a mais importante na sequência da Figura 27. A lei de controle deve ser calculada neste método, conforme o exemplo apresentado na Listagem 6.7 (JOSEPH, 2015; ROS, 2018b).

Listagem 6.7: Exemplo de utilização do método `update()` para implementar um controlador proporcional.

```
void NewController::update(const ros::Time& time, const ros::Duration&
    period)
{
  double error = setpoint_ - joint_.getPosition();
  joint_.setCommand(Kp_*error);
}
```

Por fim, o método `stopping()` é invocado apenas uma vez, quando o controlador é finalizado. Similarmente à função `starting()`, este método nem sempre é implementado.

Na Listagem 6.6, é inicializado um *subscriber* no tópico `/controller/command`, cuja função de *callback* é `commandCB()`. Neste tópico, são recebidas as referências para o controlador. Na Listagem 6.8 é apresentado um exemplo de função de *callback* e na Listagem 6.9 uma classe básica de controlador.

Listagem 6.8: Exemplo de função de *callback* para um valor escalar.

```
void NewController::commandCB(const std_msgs::Float32ConstPtr &msg){
  setpoint_ = msg->data;
}
```

Neste trabalho, foram implementados quatro controladores. Deste modo, foram contempladas as quatro abordagens apresentadas no Capítulo 5. Tendo em vista que neste capítulo foram abordadas as principais etapas para o desenvolvimento de controladores,

Listagem 6.9: Declaração da classe que implementa um controlador proporcional.

```
namespace controller_ns{

class NewController : public controller_interface::Controller<
    twil_hardware_interfaces::VoltageJointInterface>
{
public:
  bool init(twil_hardware_interfaces::VoltageJointInterface* hw,
  ros::NodeHandle &n);
  void update(const ros::Time& time, const ros::Duration& period);
  void starting(const ros::Time& time);
  void stopping(const ros::Time& time);

private:
  std::vector<hardware_interface::JointHandle> joint_;
  ros::Subscriber sub_;
  double setpoint_, Kp_;
  void commandCB(const std_msgs::Float32ConstPtr &msg);
};
}
```

tema que já havia sido amplamente debatido por BARROS (2014), MACIEL (2014), JOSEPH (2015), LAGES (2016) e LAGES (2017), não seria significativo expor, ao longo deste texto, a implementação de cada um dos controladores desenvolvidos.

6.2.3 Laço de Controle em Tempo Real

Na Figura 28 é apresentado um diagrama de blocos do laço de tempo real, do *framework* `ros_control`. Os blocos retangulares representam as classes dos objetos utilizados para implementar o laço e as setas caracterizam chamadas de funções ou acessos às variáveis, por meio de ponteiros. No caso dos ponteiros, a seta aponta para a classe que armazena a variável. No caso das funções, as setas apontam para a classe onde estão implementadas, enquanto a base representa a classe que as invoca. As elipses representam os tópicos, que podem ser publicados pelas classes, ou subscritos por elas (BARROS, 2014; LAGES, 2016).

No diagrama da Figura 28, os blocos em cinza representam as classes que são implementadas neste trabalho. A classe `RobotHW` abstrai o acesso do ROS ao *hardware* do robô. Por outro lado, a classe `controller` abstrai a estrutura de um controlador no ROS. Estas classes devem ser derivadas para cada caso particular. Isto significa que os controladores, utilizados neste *framework*, devem implementar a classe `controller`, e a comunicação com o *hardware* do robô, deve ser realizada por uma classe que implementa a `RobotHW`[2] (BARROS, 2014; LAGES, 2016). Neste trabalho, conforme apresentado na Seção 6.2.1, foi desenvolvida a classe `TwilRobotHW`, que implementa a comunicação com o *hardware* do Twil.

Na classe `ControllerManager`, é implementado o gerenciador de controladores, cuja principal função, no laço de tempo-real, é executar os controladores ativos a cada período de amostragem. O gerenciador executa estes controladores sequencialmente,

[2]Nos casos em que o robô é simulado pelo Gazebo, é utilizada a classe `RobotHWSim`, que é derivada da classe `RobotHW` e implementa a interface com o simulador.

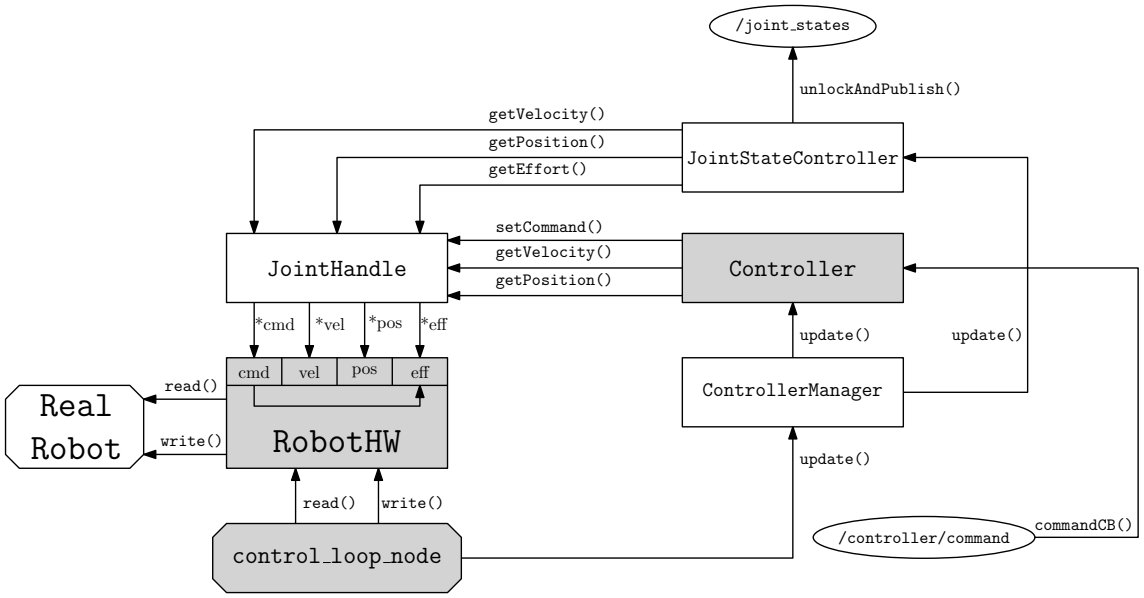


Figura 28: Laço de tempo-real do ROS.
Fonte: Adaptado de MACIEL (2014)

através de chamadas ao método `update()` de cada um deles. Conforme mencionado anteriormente, um controlador no ROS é um *plugin* para o gerenciador de controladores, e pode executar funções diferentes das apresentadas por um controlador na teoria de controle. Na Figura 28, tanto a classe `JointStateController` quanto o objeto da classe `controller` são controladores para o *framework ros_control*. O objeto da classe `controller` pode apresentar o comportamento de um controlador, conforme a teoria de controle, ou apresentar outra funcionalidade, o que depende da implementação. A classe `JointStateController`, por exemplo, implementa um objeto da classe `controller`, mas apenas publica os estados dos sensores, ligados às juntas do robô, no tópico `/joint_states` (BARROS, 2014; LAGES, 2016).

No diagrama apresentado na Figura 28, o sinal de referência para o controlador é recebido através do tópico `/controller/command`. No entanto, o modo como o sinal de referência é recebido depende da maneira como é implementada a classe do controlador. Para esta finalidade, pode-se utilizar, por exemplo, uma estrutura do tipo serviço, quando é necessária uma resposta do controlador, ou do tipo ação (através da *actionlib*), mas esta última é mais apropriada para funções de alto-nível (BARROS, 2014; LAGES, 2016), como a geração de trajetórias.

A classe `JointHandle` implementa a abstração de uma junta do robô, expondo funções que possibilitam a leitura das medidas de posição, velocidade e esforço, e a aplicação de comandos à junta. Estas funções utilizam ponteiros para acessar as respectivas variáveis, nos objetos da classe `RobotHW`, que implementam o acesso ao *hardware* do robô (BARROS, 2014; LAGES, 2016).

Quando o robô é simulado pelo Gazebo, a sincronização temporal entre o simulador e o ROS é realizada pelo *plugin* `GazeboRosControlPlugin` (LAGES, 2016, p. 688). No entanto, quando é utilizado um robô real, deve-se implementar um nodo, que será responsável por gerar esta sincronização. Este nodo, representado pelo bloco `control_loop_node` na Figura 28, deve ter uma implementação *realtime-safe*.

Para que seja realizada a sincronização com os controladores e com o *hardware* do robô (amostragem e atuação), o `control_loop_node` precisa instanciar objetos das

classes que implementam a RobotHW e, também, do ControllerManager. Para o caso particular do Twil, isto é feito conforme a Listagem 6.10.

Listagem 6.10: Objetos instanciados pelo control_loop_node.

```
twil_hardware::TwilRobotHW real_robot_hw;
real_robot_hw.init(nh, pnh);
controller_manager::ControllerManager cm(&real_robot_hw);
```

O control_loop_node implementa um laço de repetição onde é executada a sincronização com os controladores e com o *hardware* do robô. Para isto, são invocados os métodos RobotHW::read(), ControllerManager::update() e RobotHW::write() dos objetos das classes RobotHW e ControllerManager. Os métodos relacionados à classe RobotHW são invocados através dos objetos que a implementam, representado pelo real_robot_hw na Listagem 6.10. Um exemplo de implementação do control_loop_node é demonstrado no Apêndice B, e uma versão simplificada, do laço de repetição deste nodo, é apresentada na Listagem 6.11.

Listagem 6.11: Sincronização realizada pelo control_loop_node.

```
real_robot_hw.read(current_time,dt);
cm.update(current_time, dt);
real_robot_hw.write(current_time,dt);
```

Na nomenclatura do *framework ros_control*, a estrutura apresentada na Figura 28 é denominada laço de tempo-real. É importante ressaltar que existe apenas um laço de tempo-real, onde todos os controladores são executados em sequência, e todos os objetos da RobotHW são amostrados e atuados. Portanto, se houver a necessidade de amostrar um sensor com uma frequência menor ou de executar um controlador com uma taxa mais baixa, cada um destes elementos deve implementar uma subamostragem.

Por fim, o laço de repetição do control_loop_node é executado a cada período de amostragem e segue a seguinte sequência (MACIEL, 2014; BARROS, 2014; LAGES, 2016):

1. O método TwilRobotHW::read() é invocado a partir do laço de repetição do control_loop_node. A classe TwilRobotHW é a implementação da classe RobotHW, para o robô móvel Twil.

2. O método TwilRobotHW::read() realiza a leitura dos dados dos sensores, ligados às juntas do robô, através das placas AIC, e armazena-os em variáveis privadas, que serão utilizadas posteriormente.

3. O método ControllerManager::update() é invocado a partir do laço de repetição do control_loop_node.

4. O método ControllerManager::update() invoca, em sequência, a função update() de cada controlador ativo.

5. O método update() de cada controlador ativo obtém as leituras dos sensores. Estes dados são obtidos, para cada uma das juntas registradas, através dos métodos JointHandle::getEffort(), JointHandle::getVelocity() e JointHandle::getPosition(), que acessam, utilizando ponteiros, as variáveis privadas da TwilRobotHW. Em seguida, é executada a funcionalidade

para a qual o controlador foi implementado. Se for o caso, os esforços de controle são gravados nas variáveis privadas da `TwilRobotHW`, através dos métodos `JointHandle::setCommand()` relacionados aos *handles* das juntas.

6. O método `TwilRobotHW::write()` é invocado a partir do laço de repetição do `control_loop_node`. Este método aplica os esforços de controle através das placas AIC, a partir dos valores armazenados nas variáveis privadas, finalizando o ciclo de controle.

Os métodos `getPosition()`, `getVelocity()` e `getEffort()`, da classe `JointHandle`, apenas retornam os valores que foram armazenados pela função `read()` da classe `TwilRobotHW`, nas variáveis `pos`, `vel` e `eff`. Em contrapartida, o método `setCommand()`, da classe `JointHandle`, apenas armazena o esforço de controle na variável `cmd`, para ser utilizado pelo método `write()` da classe `TwilRobotHW`.

No próximo capítulo serão apresentados os resultados obtidos com os métodos implementados nesta dissertação.

7 RESULTADOS EXPERIMENTAIS

O estudo de caso foi realizado com o robô móvel Twil, desenvolvido por LAGES (1998), e que, ao longo dos anos, recebeu atualizações somente nas estruturas eletrônica e de processamento. Portanto, os atuadores e sistemas de transmissão apresentam várias não idealidades.

Em todos os experimentos a frequência de amostragem dos sensores foi de $100\,\mathrm{Hz}$, sendo esta a taxa sob a qual as malhas de controle de velocidade operam. As malhas de controle de pose são executadas com uma frequência de $20\,\mathrm{Hz}$. As informações de deslocamento foram obtidas a partir do método de *Dead-Reckoning* (Seção 3.2), utilizando-se as medidas dos *encoders* do robô, que fornecem 2048 pulsos por rotação, resultando em 8192 contagens quando decodificados em quadratura. Por fim, a velocidade máxima das rodas é de $30\,\mathrm{rpm}$, quando são aplicados $12\,\mathrm{V}$ na armadura dos motores.

7.1 Identificação de Parâmetros

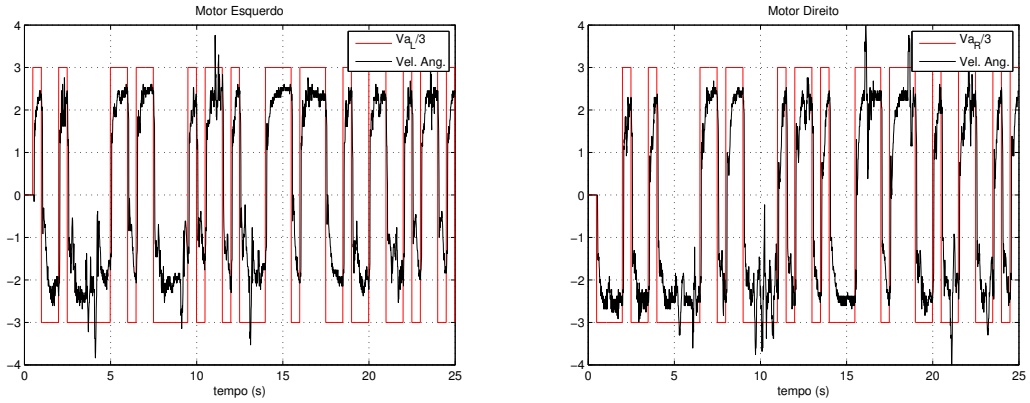
A identificação de parâmetros foi dividida em duas partes: a) identificação do modelo dinâmico completo do robô e b) identificação do modelo dinâmico dos atuadores. O modelo completo é utilizado pelo sistema de controle de velocidade, baseado na linearização da dinâmica, que foi apresentado na Seção 5.2. O modelo dos atuadores, em contrapartida, é utilizado no sistema de controle de velocidade, com atuação independente por junta, apresentado na Seção 5.1.

7.1.1 Identificação de Parâmetros do Modelo Dinâmico Completo

O primeiro passo deste processo consistiu na escolha dos sinais de excitação. Optou-se por utilizar como entradas dois sinais do tipo *Pseudo-Random Binary Sequence* (PRBS), não correlacionados, com amplitude de $9\,\mathrm{V}$ e frequência de $2\,\mathrm{Hz}$. O sinal PRBS possui propriedades espectrais próximas às de um ruído branco e possibilita que o sistema seja persistentemente excitado (LJUNG, 1999, p. 418).

Para a realização dos experimentos, os sinais foram aplicados simultaneamente nas duas rodas. As medidas dos *encoders* foram adquiridas com uma frequência de $100\,\mathrm{Hz}$ e registradas em um arquivo ASCII, juntamente com as informações de temporização e dos sinais aplicados. Nas Figuras 29a e 29b são apresentados os sinais de tensão aplicados aos motores e as velocidades medidas nos eixos das rodas.

A identificação das contantes K_A a K_F foi realizada com o método RLS, utilizando-se os modelos recursivos (182) e (183), e os regressores (186) e (187). Na Figura 30 é apresentada a evolução dos parâmetros no processo de identificação e, na Figura 31, as diagonais das matrizes de covariância relacionadas a estas variáveis. A Figura 32 apresenta



(a) Atuador esquerdo.

(b) Atuador direito.

Figura 29: Excitação do tipo PRBS aplicada aos atuadores do robô.

as respostas obtidas com o modelo identificado e as respostas do sistema. Deve-se salientar que, os sistemas de transmissão dos atuadores do Twil são suportados por mancais, sem a utilização de rolamentos, o que ocasiona ruídos nas medidas dos *encoders*. Estes ruídos tendem a ser amplificados pelas derivadas numéricas em (182) e (183) e, como consequência, reduzem a precisão dos parâmetros identificados. Por fim, as constantes identificadas no processo foram:

$$\begin{aligned} K_A &= -11{,}252 \\ K_B &= 0{,}012 \\ K_C &= 0{,}098 \\ K_D &= -7{,}551 \\ K_E &= -8{,}405 \\ K_F &= 0{,}498 \end{aligned} \tag{208}$$

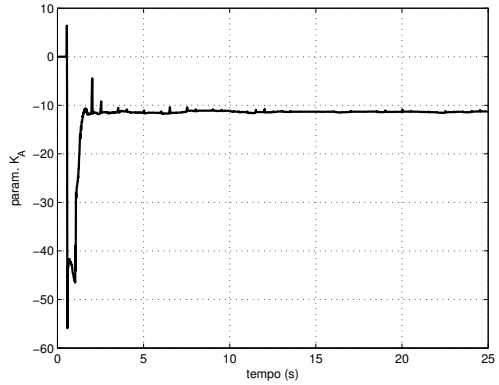
A validação foi realizada com os sinais do tipo degrau apresentados na Figura 33. O resultado da aplicação destes sinais ao robô e ao modelo identificado é apresentado na Figura 34. O modelo ajusta apropriadamente a resposta do sistema. No entanto, existem vários efeitos físicos que não são modelados, e, portanto, não são verificados na resposta do modelo.

7.1.2 Identificação de Parâmetros dos Atuadores

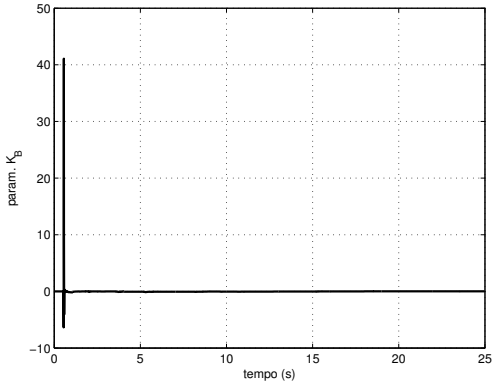
Neste trabalho, assumiu-se que os atuadores são idênticos, conforme os modelos apresentados nas Seções 4.2 e 5.1. Além disto, conforme discutido na Seção 5.1, estes atuadores podem ser descritos por uma função de transferência de primeira ordem (195). Os parâmetros da função de transferência foram identificados utilizando-se o método `tfest` (*Transfer function estimation*) (Mathworks, 2018), que é disponibilizado na *System Identification Toolbox* do *software MATLAB*.

Neste procedimento foram utilizados os mesmos sinais de excitação (Figuras 29a e 29b), e medidas de velocidade das rodas, que haviam sido empregados na seção anterior. Foram identificados somente os parâmetros do motor esquerdo, que serão utilizados nos modelos de ambos os atuadores.

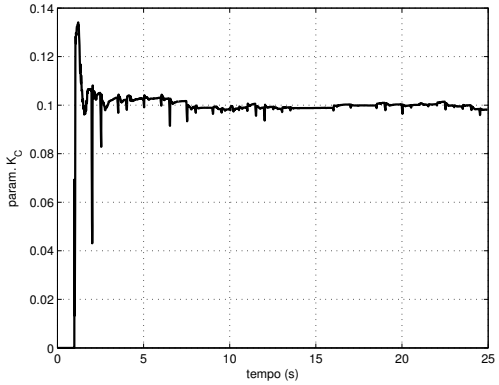
Na Figura 35, são apresentadas as respostas obtidas no processo de identificação do



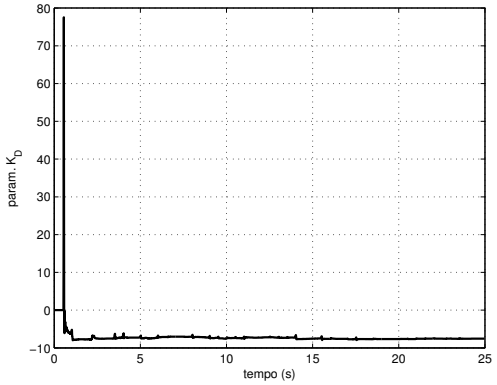
(a) Parâmetro K_A.



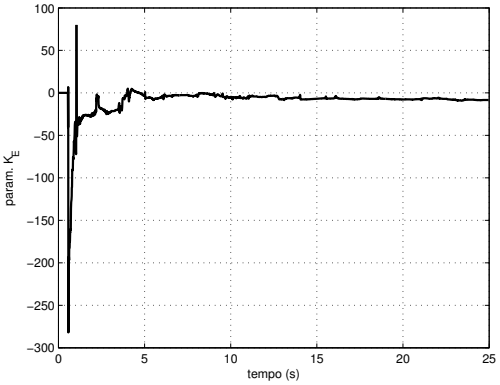
(b) Parâmetro K_B.



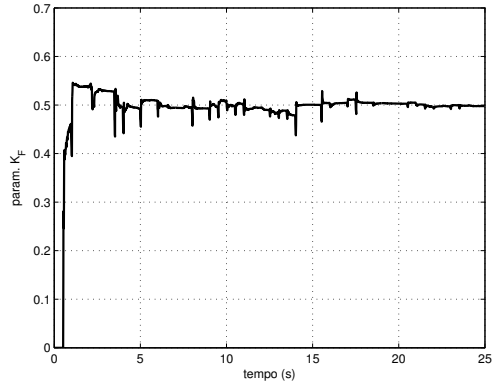
(c) Parâmetro K_C.



(d) Parâmetro K_D.

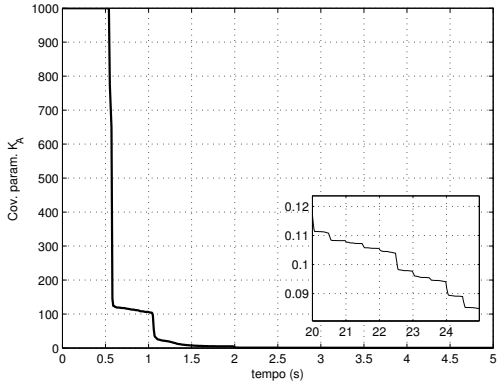


(e) Parâmetro K_E.

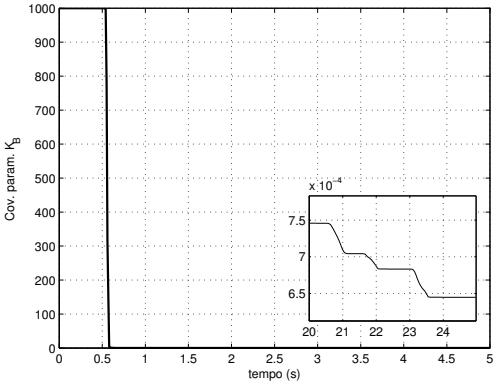


(f) Parâmetro K_F.

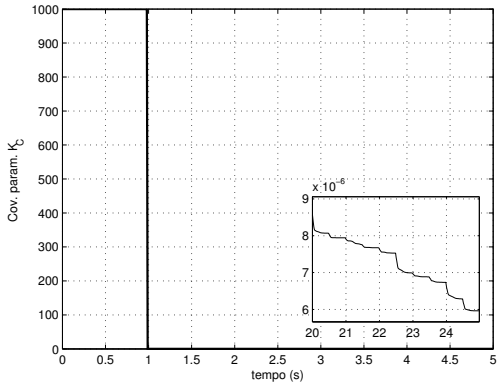
Figura 30: Evolução dos valores estimados para os parâmetros do modelo dinâmico completo.



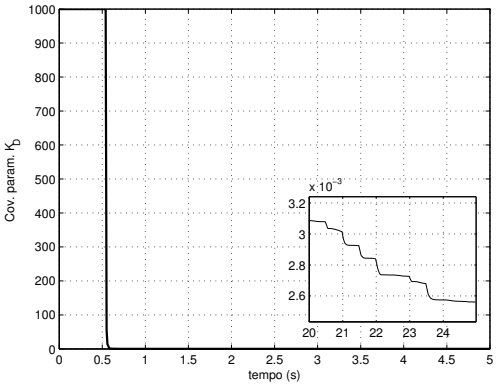
(a) Covariância do parâmetro K_A.



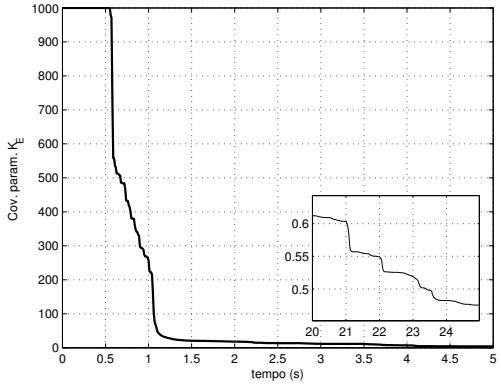
(b) Covariância do parâmetro K_B.



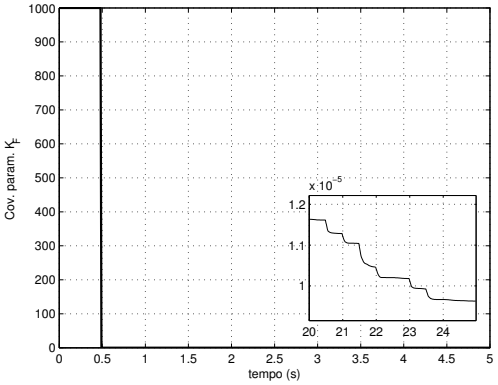
(c) Covariância do parâmetro K_C.



(d) Covariância do parâmetro K_D.

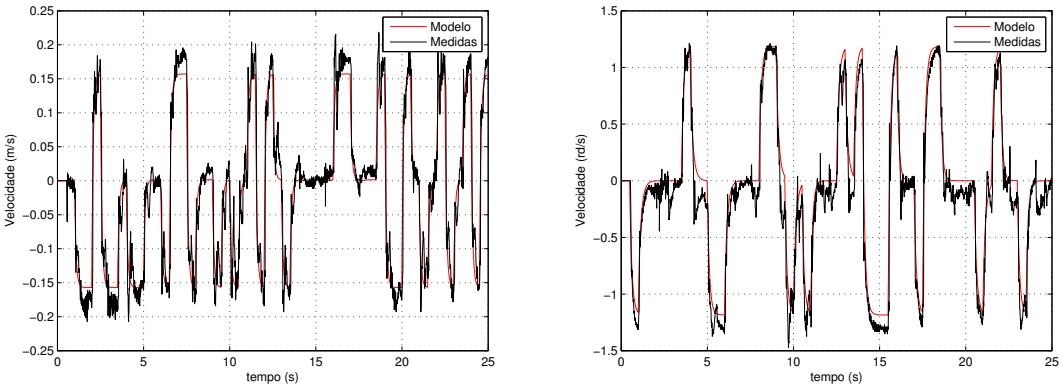


(e) Covariância do parâmetro K_E.



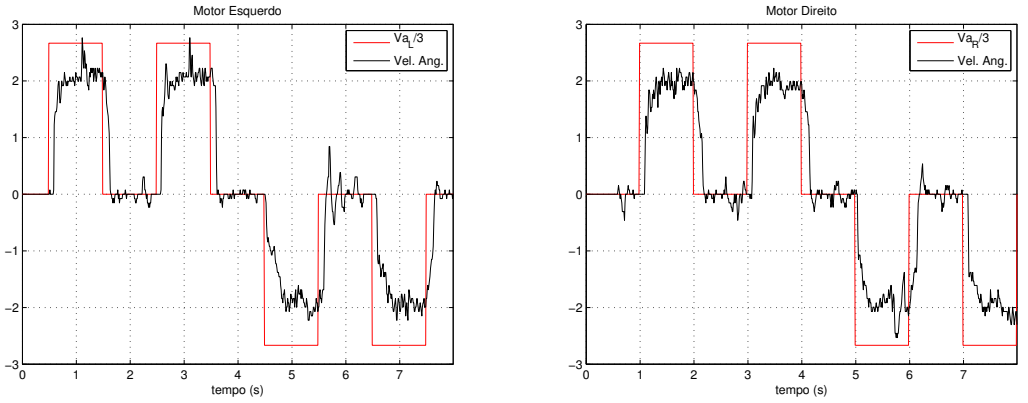
(f) Covariância do parâmetro K_F.

Figura 31: Valores da diagonal da matriz de covariância, relacionados aos parâmetros do modelo dinâmico completo.



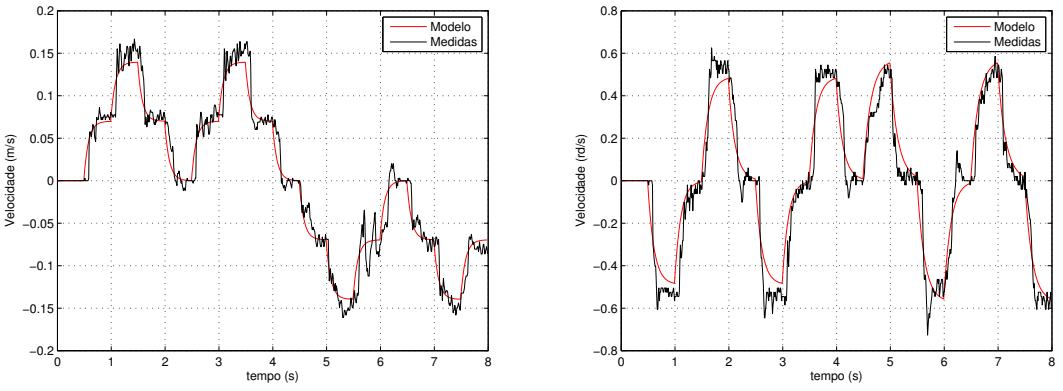
(a) Velocidade linear. (b) Velocidade angular.

Figura 32: Velocidades do robô móvel para uma entrada do tipo PRBS.



(a) Atuador esquerdo. (b) Atuador direito.

Figura 33: Excitação do tipo degrau aplicada aos atuadores do robô.



(a) Velocidade linear. (b) Velocidade angular.

Figura 34: Velocidades do robô móvel para entradas do tipo degrau.

atuador esquerdo. A função de transferência obtida no processo foi:

$$G(s) = \frac{K_G}{s+a} = \frac{2{,}06}{s+8{,}14} \tag{209}$$

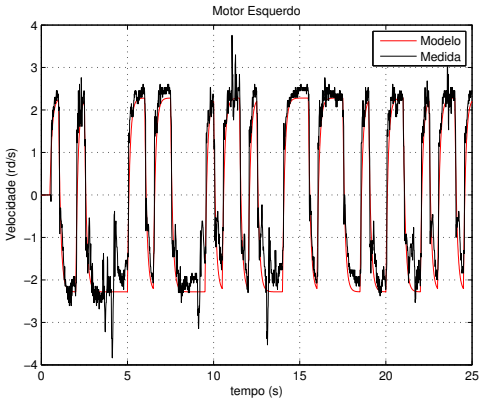
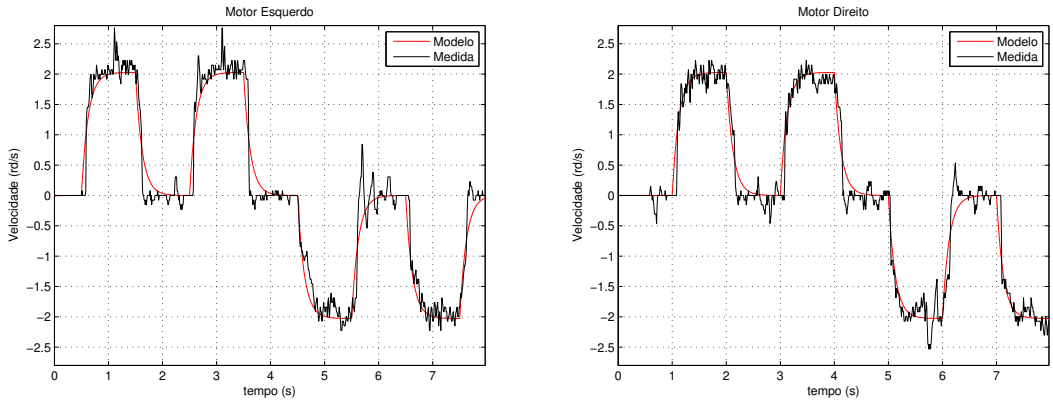


Figura 35: Velocidade da roda esquerda para uma entrada do tipo PRBS.

Novamente, a validação foi realizada com os sinais do tipo degrau apresentados na Figura 33. O resultado da aplicação destes sinais ao modelo e aos atuadores é apresentado na Figura 36. O modelo ajusta apropriadamente as respostas de velocidade dos atuadores. No entanto, os efeitos físicos não modelados são observados nas medidas dos atuadores, mas não na resposta do modelo.



(a) Velocidade da roda esquerda.

(b) Velocidade da roda direita.

Figura 36: Excitação do tipo PRBS aplicada ao atuador esquerdo do robô.

7.2 Controladores de Velocidade

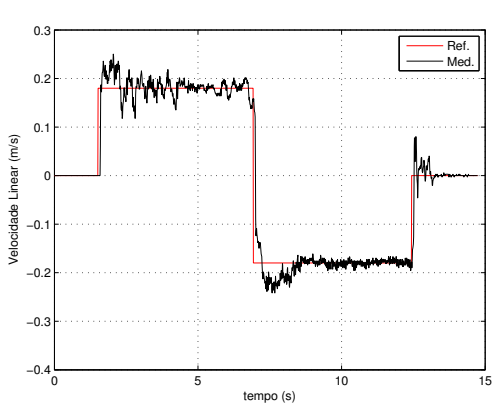
Nesta seção são avaliados os controladores de velocidade apresentados nas Seções 5.1 e 5.2. A sintonia dos ganhos, de ambos os controladores de velocidade propostos, baseou-se nos seguintes critérios de projeto: a) coeficiente de amortecimento unitário ($\xi = 1$) e b) tempo de assentamento (T_s) de $500\,\mathrm{ms}$. Foi empregado um valor elevado de T_s devido ao comportamento dinâmico dos atuadores, pois a utilização de constantes de tempo inferiores tende a excitar a dinâmica não modelada, instabilizando os controladores.

7.2.1 Controlador de Velocidade com Linearização da Dinâmica

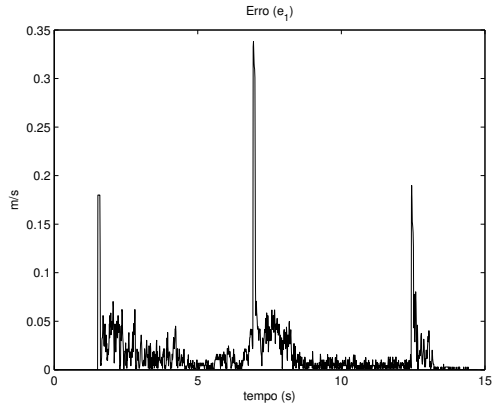
Embora seja possível utilizar características de desempenho diferentes para o controle das velocidades linear (v) e angular (ω), neste trabalho foram aplicados os mesmo critérios de projeto, conforme discutido acima. Então, aplicando-se estes valores às equações (198), (206) e (207), são obtidos os seguintes ganhos:

$$\begin{aligned} k_p &= 16 \\ k_i &= 64 \end{aligned}$$

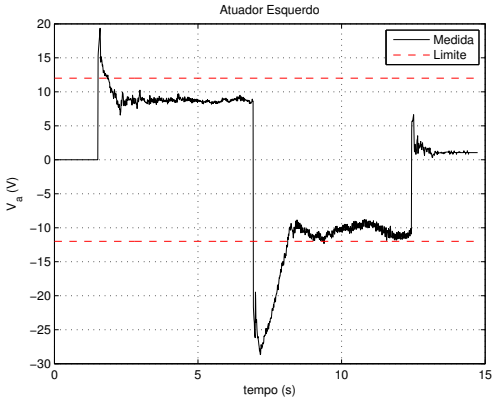
A avaliação de desempenho do controlador, aplicado à velocidade linear do robô, foi realizada utilizando-se a referência $\boldsymbol{u}_r = \begin{bmatrix} 0{,}18 & 0 \end{bmatrix}^T$, que após $5\,\mathrm{s}$ foi revertida para $\boldsymbol{u}_r = \begin{bmatrix} -0{,}18 & 0 \end{bmatrix}^T$. Os resultados deste experimento encontram-se na Figura 37. Similarmente, a avaliação de desempenho do controlador, aplicado à velocidade angular do robô, foi realizada utilizando-se a referência $\boldsymbol{u}_r = \begin{bmatrix} 0 & 1{,}3 \end{bmatrix}^T$, que após $5\,\mathrm{s}$ foi revertida para $\boldsymbol{u}_r = \begin{bmatrix} 0 & -1{,}3 \end{bmatrix}^T$. Os resultados deste experimento encontram-se na Figura 38.



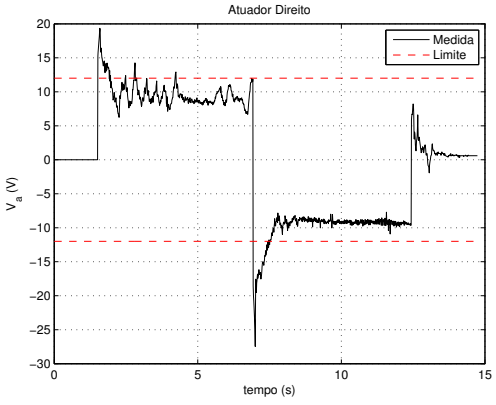
(a) Velocidade linear do robô.



(b) Erro (e_{u_1}).



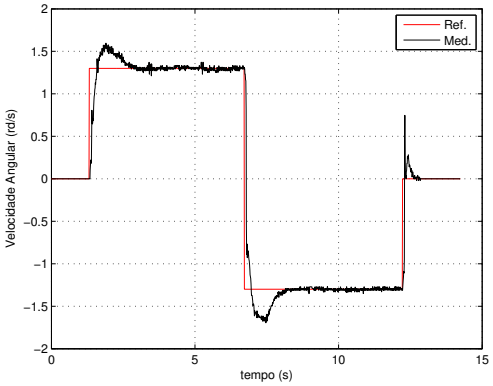
(c) Tensão aplicada ao atuador esquerdo.



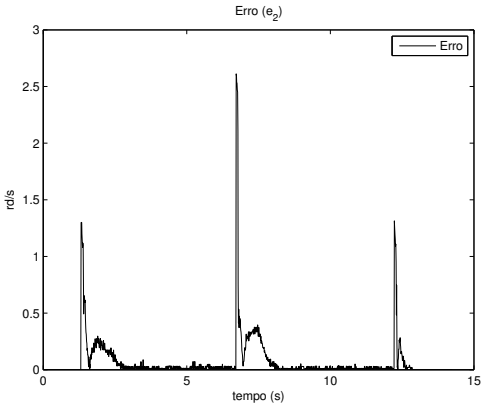
(d) Tensão aplicada ao atuador direito.

Figura 37: Resposta de velocidade linear.

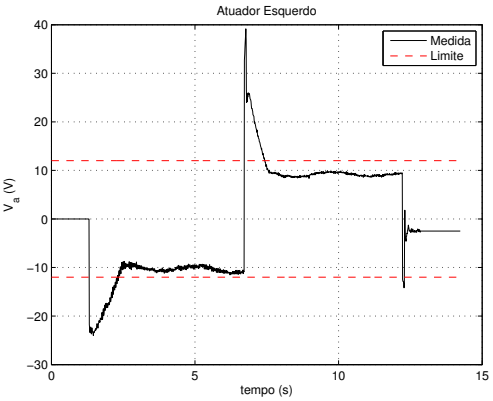
Nestes experimentos foram aplicadas referências do tipo degrau, que levam a esforços de controle próximos dos limites dos atuadores. Contudo, como pode ser observado nas Figuras 37 e 38, os controladores conseguem recuperar-se do estado de saturação e rastrear as velocidades de referência.



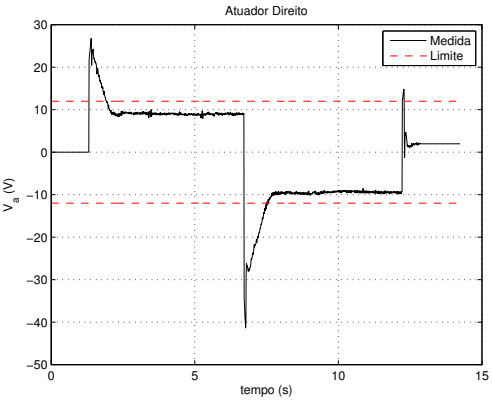
(a) Velocidade angular do robô.



(b) Erro (e_{u_2}).



(c) Tensão aplicada ao atuador esquerdo.



(d) Tensão aplicada ao atuador direito.

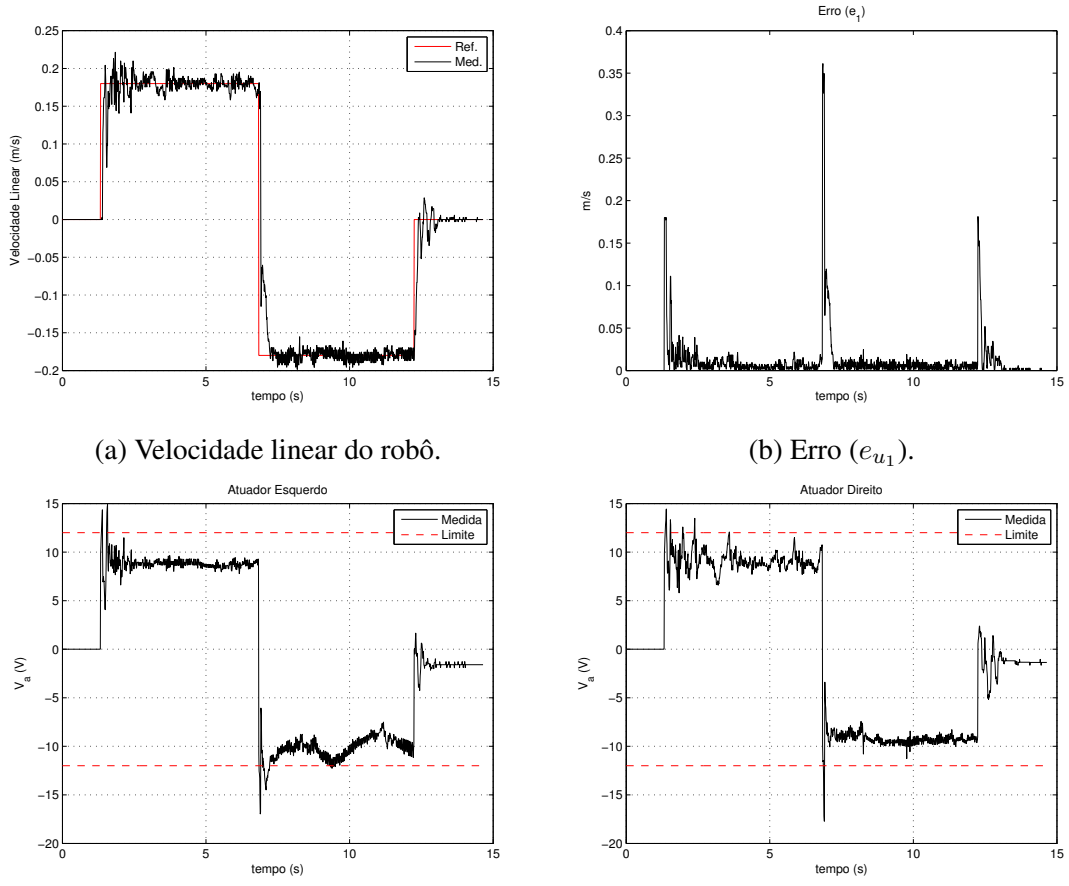
Figura 38: Resposta de velocidade angular.

7.2.2 Controlador de Velocidade Independente por Junta

Aplicando-se os critérios de projeto e as constantes identificadas na Seção 7.1.2 às equações (198), (199) e (200), são obtidos os seguintes ganhos:

$$\begin{aligned} k_p &= 3{,}81 \\ k_i &= 31{,}02 \end{aligned}$$

A avaliação de desempenho do controlador, aplicado à velocidade linear do robô, foi realizada utilizando-se a referência $\boldsymbol{u}_r = \begin{bmatrix} 0{,}18 & 0 \end{bmatrix}^T$, que após $5\,\mathrm{s}$ foi revertida para $\boldsymbol{u}_r = \begin{bmatrix} -0{,}18 & 0 \end{bmatrix}^T$. Os resultados deste experimento encontram-se na Figura 39. Similarmente, a avaliação de desempenho do controlador, aplicado à velocidade angular do robô, foi realizada utilizando-se a referência $\boldsymbol{u}_r = \begin{bmatrix} 0 & 1{,}3 \end{bmatrix}^T$, que após $5\,\mathrm{s}$ foi revertida para $\boldsymbol{u}_r = \begin{bmatrix} 0 & -1{,}3 \end{bmatrix}^T$. Os resultados deste experimento encontram-se na Figura 40.



(a) Velocidade linear do robô.

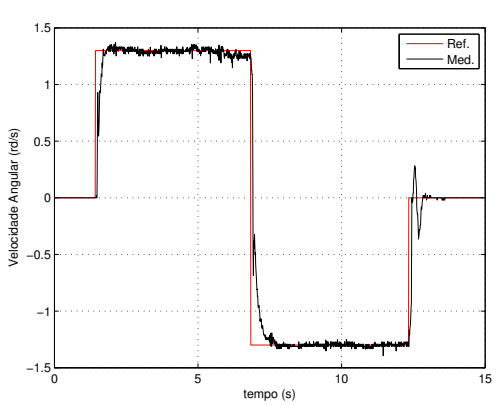
(b) Erro (e_{u_1}).

(c) Tensão aplicada ao atuador esquerdo.

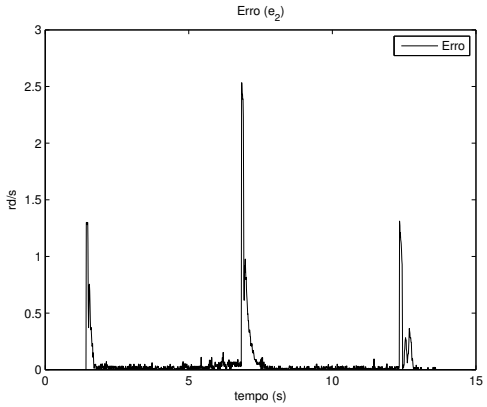
(d) Tensão aplicada ao atuador direito.

Figura 39: Resposta de velocidade linear.

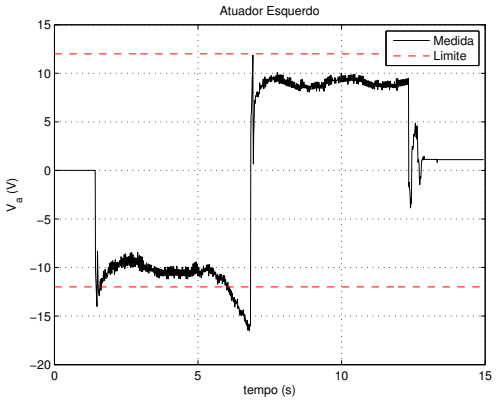
De forma análoga ao caso anterior, foram aplicadas referências do tipo degrau próximas dos limites dos atuadores. Contudo, como pode ser observado nas Figuras 39 e 40, os controladores conseguem recuperar-se do estado de saturação e rastrear as velocidades de referência.



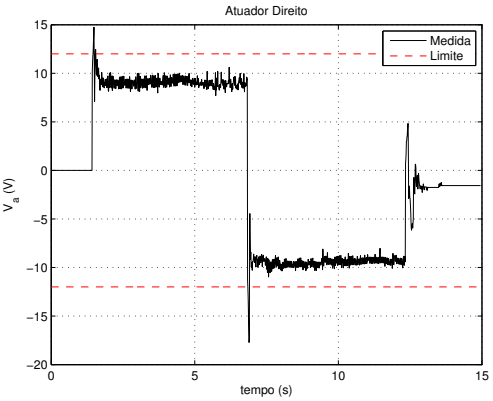
(a) Velocidade angular do robô.



(b) Erro (e_{u_2}).



(c) Tensão aplicada ao atuador esquerdo.



(d) Tensão aplicada ao atuador direito.

Figura 40: Resposta de velocidade angular.

7.3 Estabilização em uma Pose (*Parking*)

Nesta seção serão avaliados os controladores propostos nas Seções 5.3 e 5.4, aplicados à tarefa de estabilização em uma pose (*Parking*). Tendo em vista que o ambiente onde seriam realizados os experimentos possui uma área útil de $2.5\,\mathrm{m}\times4\,\mathrm{m}$, foram planejados deslocamentos confinados a um circulo com dois metros de diâmetro.

7.3.1 Controlador de Pose com Linearização da Dinâmica

Diferentemente dos controladores lineares, como os PIs usados nas malhas de velocidade, não há um método intuitivo para o ajuste dos ganhos do controlador de pose. Portanto, estes parâmetros foram determinados por simulação (ALVES; LAGES; HENRIQUES, 2018b), baseando-se na estrutura de controle completa (Figura 21), incluindo os efeitos de dinâmica da base móvel e dos atuadores e, também, os controladores.

Além disto, devido às características dinâmicas dos atuadores, e ao tempo de assentamento elevado dos controladores de velocidade, optou-se por parâmetros com taxa de convergência menos acentuada, mas que evitam a ocorrência de saturação do controlador de velocidade. Assim, os parâmetros determinados para a tarefa de estabilização em um ponto (*Parking*) foram:

$$\begin{aligned}\boldsymbol{\lambda} &= \begin{bmatrix}1{,}0 & 1{,}0 & 1{,}0\end{bmatrix}\\ \boldsymbol{\gamma} &= \begin{bmatrix}0{,}2658 & 0{,}4781\end{bmatrix}\end{aligned}$$

A avaliação de desempenho foi realizada a partir de quatro trajetórias de teste, cujas poses iniciais são:

$$\begin{aligned}A &= \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T\\ B &= \begin{bmatrix}-0{,}707 & -0{,}707 & 3{,}927\end{bmatrix}^T\\ C &= \begin{bmatrix}-0{,}707 & 0{,}707 & 2{,}356\end{bmatrix}^T\\ D &= \begin{bmatrix}0{,}707 & 0{,}707 & 0{,}785\end{bmatrix}^T\end{aligned}$$

A pose de referência para ambas as trajetórias foi definida por $\boldsymbol{x}_r = \begin{bmatrix}0{,}0 & 0{,}0 & 0{,}0\end{bmatrix}^T$. A convergência das componentes de pose pode ser verificada a partir das Figuras 41 a 43 e a trajetória no plano $X \times Y$ a partir da Figura 44.

Nas Figuras 41 a 43 são apresentadas ampliações de partes das trajetórias, que demonstram o erro de regime permanente. Este erro deve-se à zona-morta inserida, propositalmente, no controlador de pose, devido a problemas gerados pelo atrito estático no atuador. Estes problemas ocorrem quando o esforço de controle aplicado ao atuador encontra-se na zona-morta deste. Nesta situação, o integrador do controlador de velocidade integra o erro até vencer o atrito estático. Quando isto ocorre, a velocidade aumenta rapidamente, uma vez que o atrito viscoso é menor do que o atrito estático, gerando um grande erro de pose. As características não-holônomicas do robô forçam um movimento amplo para corrigir este erro de pose, dificultando a convergência para a pose de referência.

Para lidar com este problema, foi inserida uma zona-morta no erro de pose, conforme apresentado na Figura 45. Com isto, quando o erro de pose estiver dentro desta zona-morta, a referência para o controlador de velocidade será nula, fazendo com que o robô permaneça na pose atual. Os valores selecionados para esta região foram de $5\,\mathrm{mm}$ e $0{,}0175\mathrm{rad}$, respectivamente.

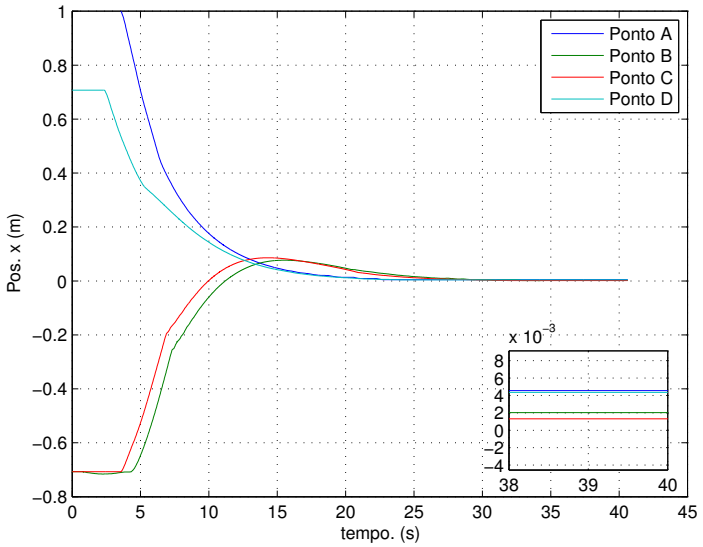


Figura 41: Convergência da variável x_c para x_r.

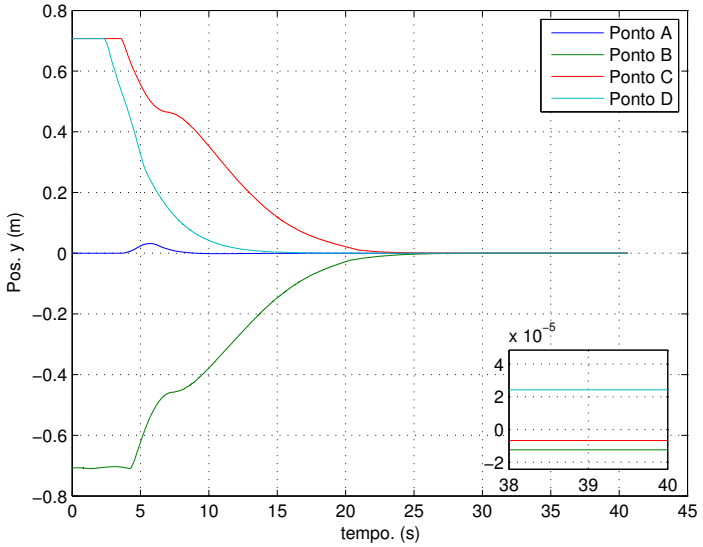


Figura 42: Convergência da variável y_c para y_r.

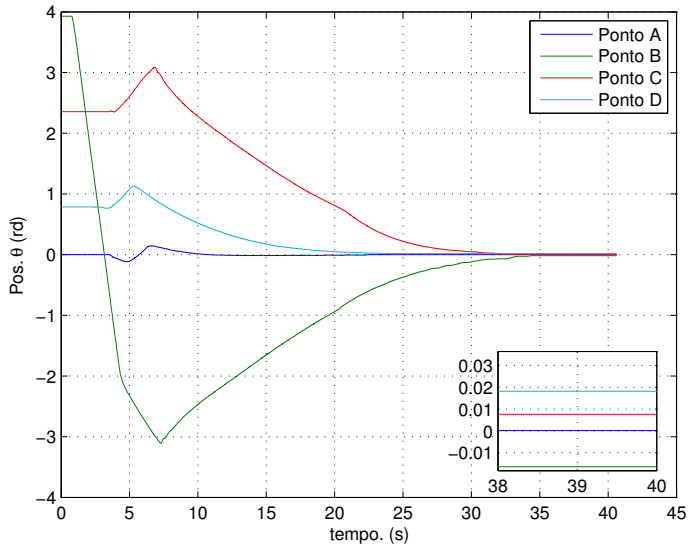


Figura 43: Convergência da variável θ_c para θ_r.

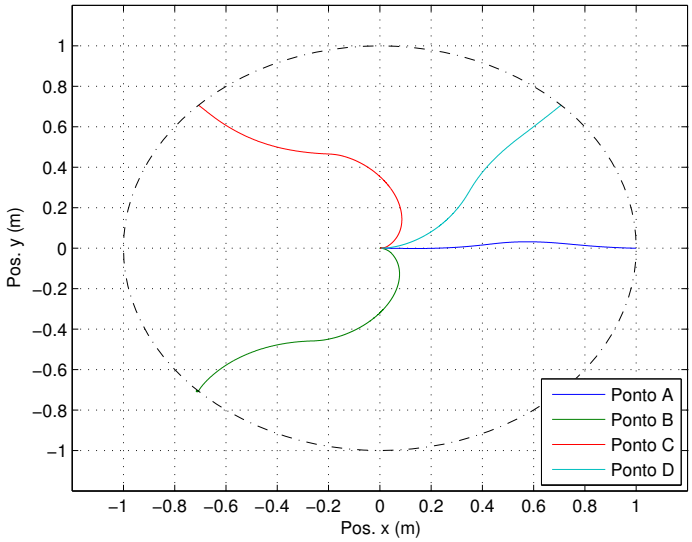


Figura 44: Trajetórias percorridas no plano $X \times Y$.

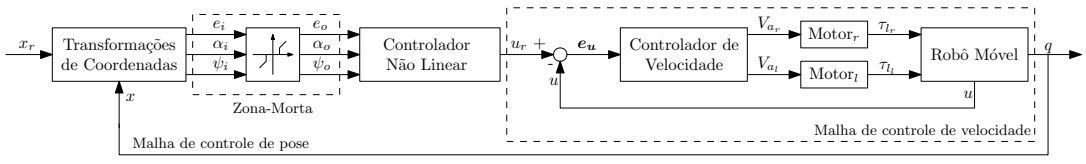


Figura 45: Controle de pose por transformação descontínua com zona-morta no erro.

7.3.2 Controlador de Pose com PI Independente por Junta

A avaliação deste sistema de controle de pose seguiu os mesmos procedimentos da Seção 7.3.1. Embora a malha de velocidade seja diferente (Figura 20), os ganhos da malha de pose foram mantidos os mesmos da Seção 7.3.1, que são:

$$\begin{aligned} \boldsymbol{\lambda} &= \begin{bmatrix} 1{,}0 & 1{,}0 & 1{,}0 \end{bmatrix} \\ \boldsymbol{\gamma} &= \begin{bmatrix} 0{,}2658 & 0{,}4781 \end{bmatrix} \end{aligned}$$

De forma análoga ao caso anterior, a avaliação de desempenho foi realizada a partir de quatro trajetórias de teste, cujas poses iniciais são:

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \\ B &= \begin{bmatrix} -0{,}707 & -0{,}707 & 3{,}927 \end{bmatrix}^T \\ C &= \begin{bmatrix} -0{,}707 & 0{,}707 & 2{,}356 \end{bmatrix}^T \\ D &= \begin{bmatrix} 0{,}707 & 0{,}707 & 0{,}785 \end{bmatrix}^T \end{aligned}$$

A pose de referência para ambas as trajetórias foi de finida por $\boldsymbol{x}_r = \begin{bmatrix} 0{,}0 & 0{,}0 & 0{,}0 \end{bmatrix}^T$. A convergência das componentes de pose pode ser verificada a partir das Figuras 46 a 48 e a trajetória no plano $X \times Y$ a partir da Figura 49. Cabe mencionar que, para ambos os controladores, as referências foram geradas de forma manual e, portanto, as trajetórias iniciam em instantes de tempo diferentes.

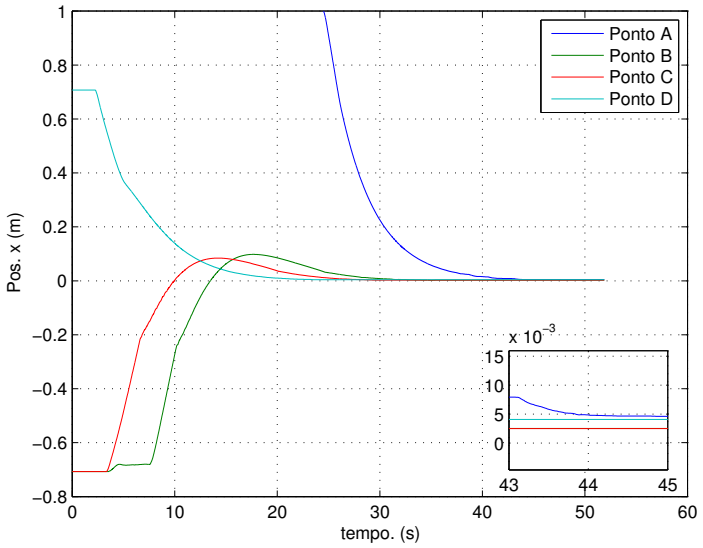


Figura 46: Convergência da variável x_c para x_r.

Nas Figuras 46 a 48 são apresentadas ampliações de partes das trajetórias, que demonstram o erro de regime permanente. Neste experimento, também, aplicou-se uma zona-morta ao controlador de pose, conforme a Figura 45, e os valores selecionados para esta região foram os mesmos do caso anterior.

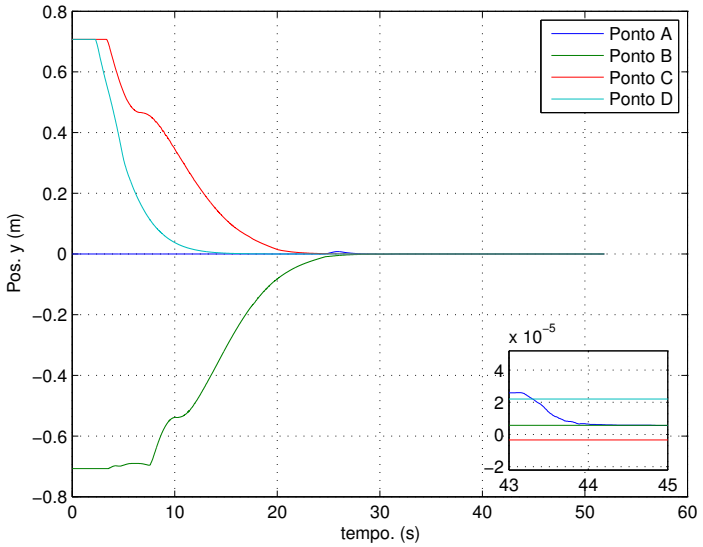


Figura 47: Convergência da variável y_c para y_r.

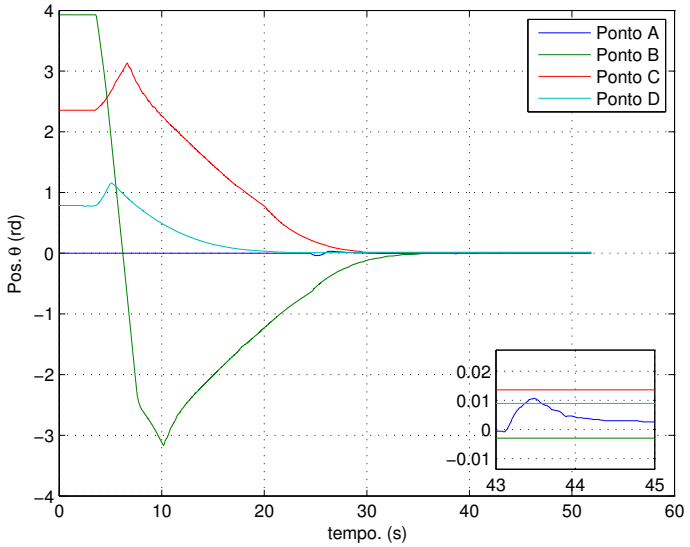


Figura 48: Convergência da variável θ_c para θ_r.

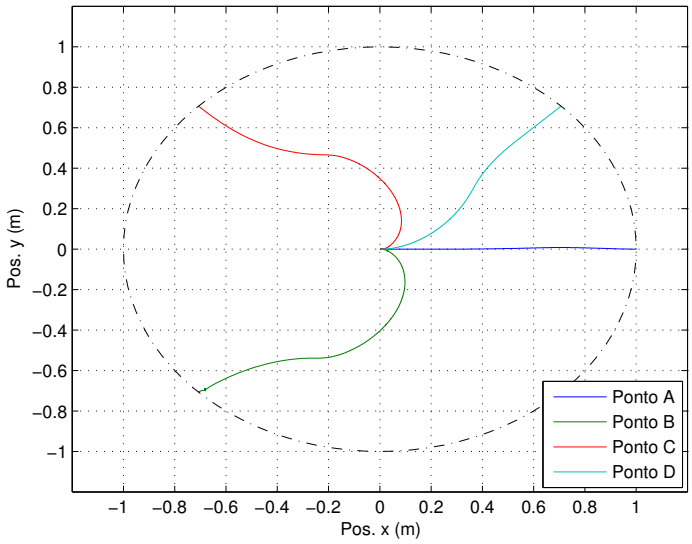


Figura 49: Trajetórias percorridas no plano $X \times Y$.

7.4 Rastreamento de Trajetória (*Tracking*)

Nesta seção, serão avaliados os controladores propostos nas Seções 5.3 e 5.4, aplicados à tarefa de rastreamento de trajetórias (*Tracking*). Neste objetivo de controle, embora os controladores de velocidade possam ser os mesmos que para a tarefa de *Parking*, para que seja obtido um melhor desempenho, os ganhos do controlador de pose devem ser novamente ajustados. Isto decorre das características de não-holonomicidade dos RMRs diferenciais. Por este motivo, para que haja rastreamento de posição, faz-se necessário reduzir a precisão do rastreamento da orientação. Caso contrário, a trajetória resultante tornar-se-ía pouco natural, pois o robô realizaria muitas curvas para reduzir o erro de orientação.

7.4.1 Controlador de Pose com Linearização da Dinâmica

Novamente, os parâmetros do controlador de pose foram determinados por simulação, considerando-se as mesmas condições da Seção 7.3.1, com a exceção de que a escolha dos ganhos deveria priorizar a redução do erro de posição, resultando em:

$$\begin{aligned} \boldsymbol{\lambda} &= \begin{bmatrix} 1{,}0 & 1{,}0 & 1{,}0 \end{bmatrix} \\ \boldsymbol{\gamma} &= \begin{bmatrix} 0{,}8 & 0{,}01 \end{bmatrix} \end{aligned}$$

A avaliação de desempenho foi realizada a partir de uma trajetória de referência, caracterizada por dois círculos lado a lado, de modo a formar um oito. Optou-se por colocar o robô em uma pose inicial $\boldsymbol{x}_i = \begin{bmatrix} 0{,}0 & -0{,}2500 & 1{,}57 \end{bmatrix}^T$, fora da trajetória, para permitir a análise da convergência para a trajetória de teste. A convergência das componentes de pose pode ser verificada a partir das Figuras 50 a 52, e a trajetória no plano $X \times Y$, a partir da Figura 53.

Observando-se as Figuras 50 a 52, fica evidente o comportamento do controlador no sentido de priorizar a posição. No entanto, a orientação também converge, embora

necessite um tempo maior. Cabe salientar que isto se deve, neste caso, à escolha dos ganhos do controlador de pose. Por fim, como pode ser observado, as variáveis convergem para as respectivas referências.

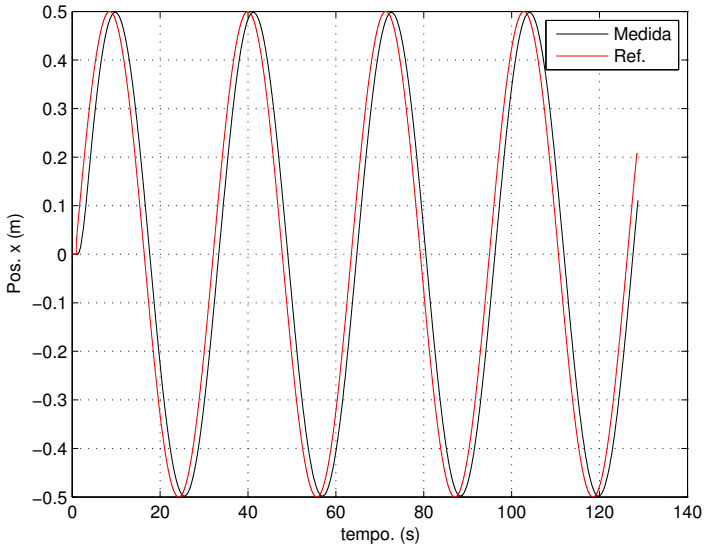


Figura 50: Rastreamento da variável x_r por x_c.

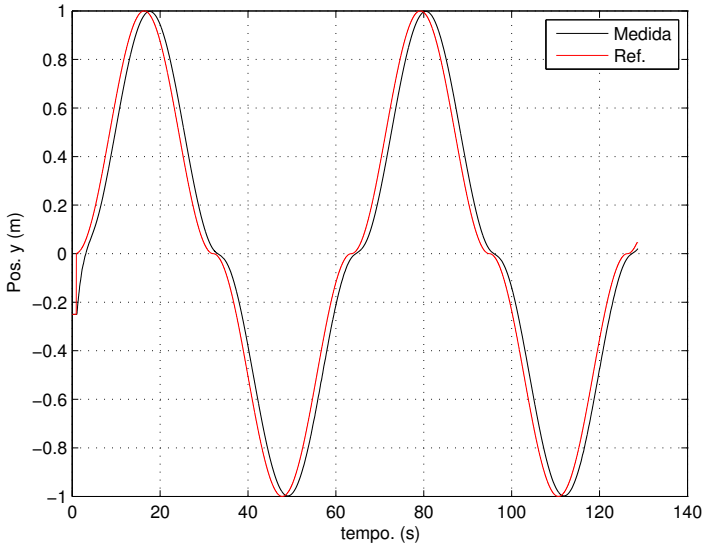


Figura 51: Rastreamento da variável y_r por y_c.

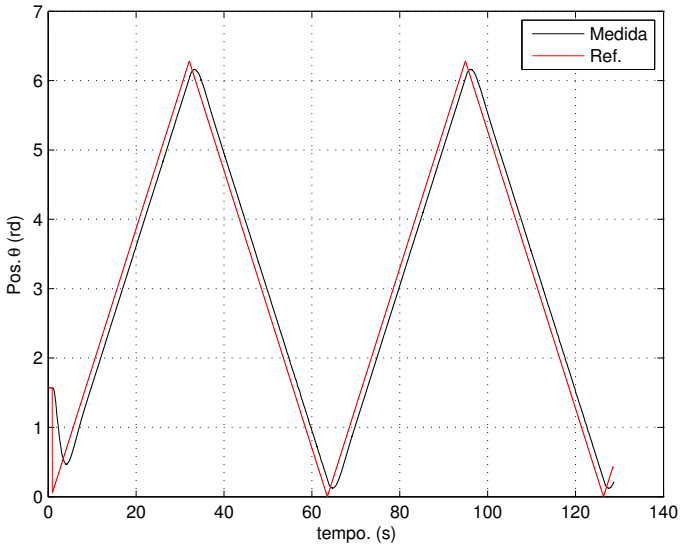


Figura 52: Rastreamento da variável θ_r para θ_c.

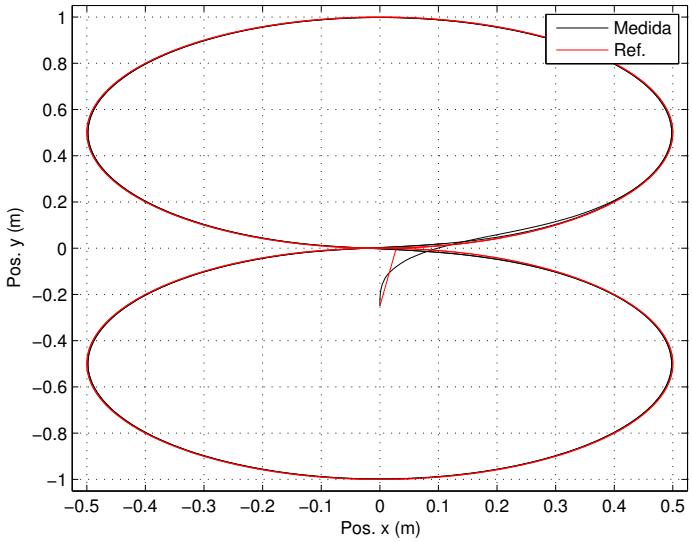


Figura 53: Trajetórias percorridas no plano $X \times Y$.

7.4.2 Controlador de Pose com PI Independente por Junta

A avaliação deste sistema de controle de pose seguiu os mesmos procedimentos da Seção 7.4.1. Portanto, os ganhos da malha de pose são:

$$\begin{aligned} \boldsymbol{\lambda} &= \begin{bmatrix} 1{,}0 & 1{,}0 & 1{,}0 \end{bmatrix} \\ \boldsymbol{\gamma} &= \begin{bmatrix} 0{,}8 & 0{,}01 \end{bmatrix} \end{aligned}$$

Utilizou-se aqui a mesma trajetória de referência, caracterizada por dois círculos lado a lado, de modo a formar um oito. Similarmente, optou-se por colocar o robô em uma pose inicial $\boldsymbol{x}_i = \begin{bmatrix} 0{,}0 & -0{,}2500 & 1{,}57 \end{bmatrix}^T$, fora da trajetória, para permitir a análise da convergência para a trajetória de teste. A convergência das componentes de pose pode ser verificadas a partir das Figuras 54 a 56, e a trajetória no plano $X \times Y$, a partir da Figura 57. Assim como no caso anterior, observando-se as Figuras 54 a 56, fica evidente o comportamento do controlador no sentido de priorizar a posição. Contudo, a orientação também converge, embora necessite um tempo maior. Novamente, este comportamento decorre da escolha dos ganhos do controlador de pose. Por fim, assim como na Seção 7.4.1, as variáveis convergem para as respectivas referências.

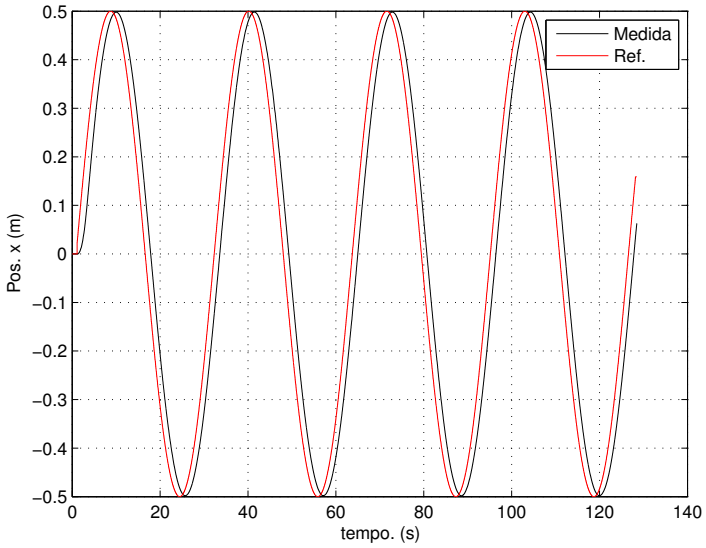


Figura 54: Rastreamento da variável x_r por x_c.

7.5 Considerações

Os experimentos realizados possibilitam avaliar a resposta dos controladores, mas uma comparação entre eles está fora do escopo deste trabalho. Consequentemente, os dados não foram capturados com os devido cuidados para tal. No entanto, diante dos resultados apresentados acima, fica evidente que os controladores de velocidade apresentaram desempenho regular. Isto se deve, principalmente, ao elevado tempo de assentamento, mas também às imperfeições dos atuadores e sistemas de transmissão.

Por fim, os resultados obtidos na seção 7.4.1 foram comparados com resultados obtidos por simulação. Os resultados para as variáveis relacionadas às coordenadas de pose

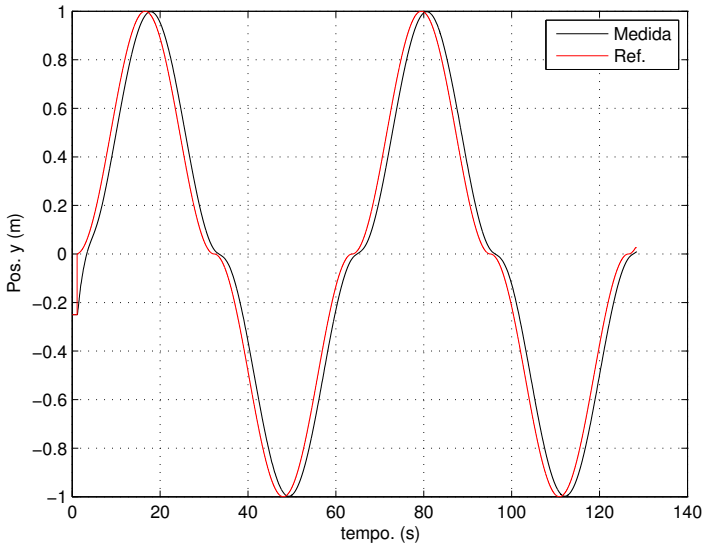


Figura 55: Rastreamento da variável y_r por y_c.

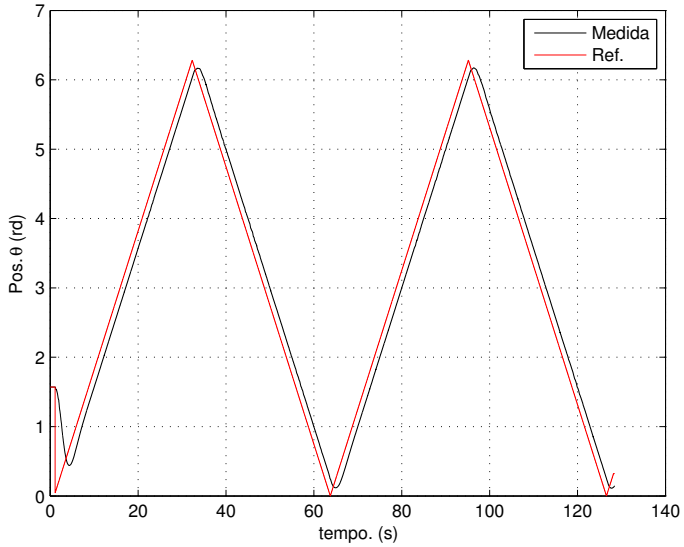


Figura 56: Rastreamento da variável θ_r para θ_c.

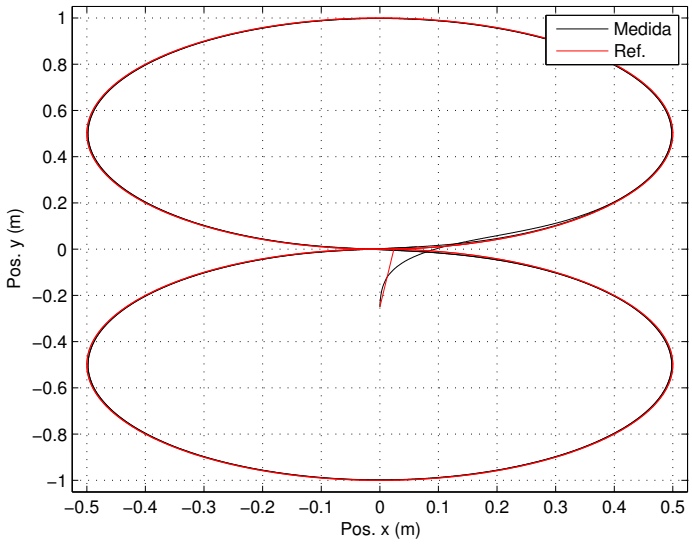
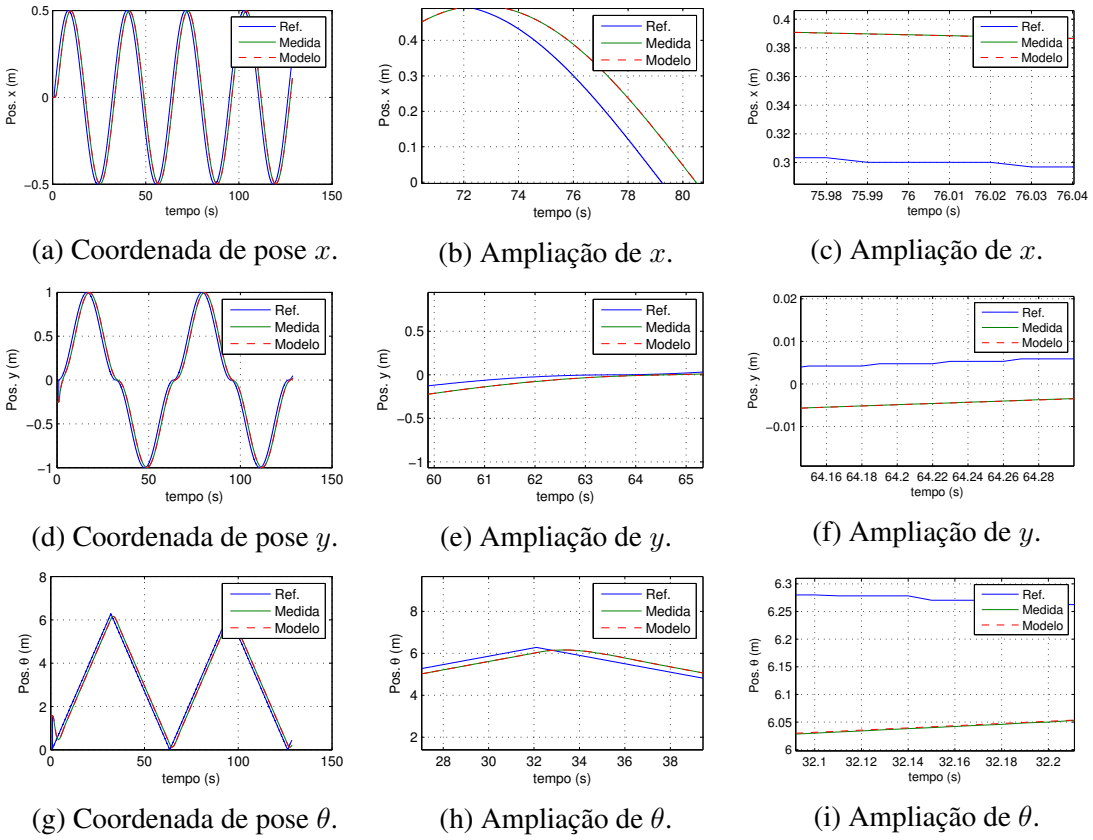


Figura 57: Trajetórias percorridas no plano $X \times Y$.

x, y, e θ são dispostos, respectivamente, nas Figuras 58a a 58c, 58d a 58f e 58g a 58i, demonstrando que o modelo representa satisfatoriamente a dinâmica do robô.

Os resultados obtidos permitem afirmar que aplicação destes controladores às cadeiras de rodas motorizadas possibilitarão ao usuário uma movimentação suave e precisa.



(a) Coordenada de pose x. (b) Ampliação de x. (c) Ampliação de x.

(d) Coordenada de pose y. (e) Ampliação de y. (f) Ampliação de y.

(g) Coordenada de pose θ. (h) Ampliação de θ. (i) Ampliação de θ.

Figura 58: Comparação entre os resultado dos controladores de pose real e simulado, para a tarefa de *Tracking*.

8 CONCLUSÕES

O presente trabalho apresentou uma metodologia para a implementação de um sistema de controle de pose para uma cadeira de rodas inteligente, capaz de levar o veículo de uma posição inicial até uma posição desejada ou fazer com este siga uma trajetória especificada a partir de comandos recebidos pelo sistema de controle.

Inicialmente, foi realizado um estudo sobre as cadeiras de rodas inteligentes, cujo desenvolvimento tem como objetivo o aumento da autonomia de indivíduos que não possuem capacidade de comandar uma cadeira de rodas motorizada convencional. Observou-se que estes veículos são encontrados, em sua grande maioria, em centros de pesquisa e universidades. Por outro lado, as cadeiras de rodas motorizadas comerciais vêm apresentando avanços referentes à concepção, mas necessitam de melhorias significativas nos sistemas de comando e controle para proporcionar mobilidade e conforto a mais usuários.

Em seguida, as modelagens cinemática e dinâmica de robôs móveis foram analisadas. Por meio das definições dos graus de mobilidade e dirigibilidade, concluiu-se que é possível a existência de apenas cinco classes de modelos de RMRs. Verificou-se que, posicionando apropriadamente o sistema de coordenadas do robô, todos os robôs pertencentes a uma determinada classe podem ser descritos pelo mesmo modelo cinemático de postura. Estas definições aplicam-se, também, às cadeiras de rodas motorizadas.

Observou-se que a grande maioria das abordagens de controle de RMRs apresentadas na literatura baseiam-se somente no modelo cinemático do robô, o que se deve, principalmente, ao fato de que considerar o modelo dinâmico eleva a complexidade do problema. No entanto, para aplicações de alto desempenho, quando são necessários movimentos rápidos ou transporte de cargas pesadas, é essencial considerar os efeitos de dinâmica do robô, além de sua cinemática.

Posteriormente, um modelo dinâmico particularizado para um robô móvel com acionamento diferencial foi avaliado. A grande maioria das cadeiras de rodas comerciais possui esta configuração cinemática e pode ser representada pelos modelos de pose desta classe. Este modelo foi aumentado para a inclusão dos efeitos dinâmicos dos atuadores. O modelo aumentado foi, então, parametrizado em um formato apropriado para que os parâmetros do robô fossem identificados utilizando-se o método RLS.

Foram, também, demonstradas duas estruturas de controle em cascata, compostas por um controlador para o rastreamento de velocidade e um controlador de pose não linear, voltadas à estabilização de posição e orientação. A primeira delas, utiliza um controlador de velocidade com PIs independentes por junta, enquanto a segunda, utiliza um controlador de velocidade baseado no modelo dinâmico do robô, linearizado por realimentação de estados.

Os métodos propostos foram implementados utilizando-se o *Robot Operating System* (ROS). Foram estudados os aspectos de implementação de controladores, apropria-

dos para execução em tempo-real, bem como a utilização do *framework* `ros_control` com um robô real. Demonstrou-se a implementação de um nodo responsável por gerar a sincronização com o *hardware* do robô e com o gerenciador de controladores. Foram, também, discutidas as características da classe que implementa a comunicação com o *hardware* do robô.

Em seguida, os resultados para os métodos propostos nesta dissertação foram apresentados. Inicialmente, foram verificados os resultados da identificação de parâmetros do modelo dinâmico completo, e dos atuadores. Em seguida, avaliou-se a convergência da linearização por realimentação de estados e a resposta dos controladores de velocidade. Posteriormente, avaliou-se o sistema de controle de pose nas tarefas de estabilização em uma pose e rastreamento de trajetórias.

Os resultados demonstraram que, mesmo sob condições mecânicas não ideais, os objetivos de controle são alcançados. Por fim, demonstrou-se que a lei de controle de pose proposta pode, ajustando-se os ganhos de forma apropriada, ser aplicada tanto à estabilização de posição e orientação, quanto ao rastreamento de trajetórias.

A partir dos resultados obtidos com a utilização dos métodos propostos, sugere-se como trabalhos futuros:

1. Aplicar os métodos de identificação e controle, propostos nesta dissertação, a uma cadeira de rodas motorizada comercial;

2. Avaliar os métodos aplicados nesta dissertação juntamente com a *stack* de navegação do ROS;

3. Estudar uma técnica para ajuste automático dos ganhos do controlador de pose. Além de reduzir a necessidade de sucessivos ensaios e simulações, poderiam ser utilizados como objetivos de sintonia, ganhos que melhorem o desempenho do sistema, bem como, que minimizem a energia gasta para o desenvolvimento das tarefas;

4. Avaliar leis de controle adaptativas como aquelas apresentadas por LAGES; HEMERLY (1998), MARTINS et al. (2008) e DE LA CRUZ; BASTOS; CARELLI (2011), que possibilitam o ajuste *online* dos parâmetros do modelo. Com este tipo de método, a degradação do desempenho do sistema de controle poderia ser evitada, quando houvessem modificações de parâmetros do veículo ou mudança de usuário.

REFERÊNCIAS

AGUIAR, A.; ATASSI, A.; PASCOAL, A. Regulation of a nonholonomic dynamic wheeled mobile robot with parametric modeling uncertainty using Lyapunov functions. In: IEEE CONFERENCE ON DECISION AND CONTROL (CAT. NO.00CH37187), 39., 2000, Sydney. **Proceedings...** New York: IEEE, 2000. v.3, p.2995–3000.

AGUIRRE, L. A. **Introdução à Identificação de Sistemas**: técnicas lineares e não-lineares aplicadas a sistemas reais. 3.ed. Belo Horizonte: UFMG, 2007. 730 p.

ALATISE, M.; HANCKE, G. Pose Estimation of a Mobile Robot Based on Fusion of IMU Data and Vision Data Using an Extended Kalman Filter. **Sensors**, Basel, v.17, n.10, p.2164, Sept. 2017.

ALEXANDER, J.; MADDOCKS, J. On the Kinematics of Wheeled Mobile Robots. **The International Journal of Robotics Research**, Thousand Oaks, v.8, n.5, p.15–27, Oct. 1989.

ALVES DE OLIVEIRA NETO, I. **Desenvolvimento de uma cadeira de rodas robótica para transporte de portador de necessidades especiais.** 2013. 89 p. Dissertação (Mestrado em Ciências de Engenharia Elétrica) — Universidade Federal do Rio Grande do Norte, Natal, 2013.

ALVES, T. G.; LAGES, W. F.; HENRIQUES, R. V. B. Parametric Identification and Controller Design for a Differential-Drive Mobile Robot. In: IFAC SYMPOSIUM ON SYSTEM IDENTIFICATION (SYSID 2018), 18., 2018, Stockholm. **Proceedings...** New York: Elsevier, 2018. p.6.

ALVES, T. G.; LAGES, W. F.; HENRIQUES, R. V. B. Non-linear Pose Stabilization Controller for a Differential-Drive Mobile Robot: optimization-based controller tuning. In: IFAC SYMPOSIUM ON ROBOT CONTROL, 12., 2018, Budapest. **Proceedings...** New York: Elsevier, 2018. p.6.

AQEL, M. O. A. et al. Review of visual odometry: types, approaches, challenges, and applications. **SpringerPlus**, Gewerbestrasse, v.5, n.1, p.1897, Dec. 2016.

ASTOLFI, A. On the stabilization of nonholonomic systems. In: IEEE CONFERENCE ON DECISION AND CONTROL, 33., 1994, Lake Buena Vista. **Proceedings...** New York: IEEE, 1994. p.3481–3486.

ÅSTRÖM, K. J.; HÄGGLUND, T. **PID Controllers**: theory, design, and tuning. 2.ed. North Carolina: International Society of Automation, 1995. 343 p.

BARROS, T. T. T. **Modelagem e Implementaçao no ROS de um Controlador para Manipuladores Moveis**. 2014. 157 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2014.

BARROS, T. T. T.; LAGES, W. F. A Mobile Manipulator Controller Implemented in the Robot Operating System. In: INTERNATIONAL SYMPOSIUM ON ROBOTICS, 41., 2014, Munich. **Proceedings...** Berlin: VDE, 2014. p.1–8.

BARSHAN, B.; DURRANT-WHYTE, H. Inertial navigation systems for mobile robots. **IEEE Transactions on Robotics and Automation**, New York, v.11, n.3, p.328–342, June 1995.

BARZAMINI, R.; YAZDIZADEH, A.; RAHMANI, A. A New Adaptive Tracking Control For Wheeled Mobile Robot. In: IEEE CONFERENCE ON ROBOTICS, AUTOMATION AND MECHATRONICS, 2006, Bangkok. **Proceedings...** New York: IEEE, 2006. p.1–6.

BASTOS-FILHO, T. F. et al. Towards a New Modality-Independent Interface for a Robotic Wheelchair. **IEEE Transactions on Neural Systems and Rehabilitation Engineering**, New York, v.22, n.3, p.567–584, May 2014.

BASTOS-FILHO, T. F.; KUMAR, D.; ARJUNAN, S. P. **Devices for Mobility and Manipulation for People with Reduced Abilities**. 1.ed. Boca Raton: CRC, 2014. 222 p.

BAUMGARTNER, E.; SKAAR, S. An autonomous vision-based mobile robot. **IEEE Transactions on Automatic Control**, New York, v.39, n.3, p.493–502, Mar. 1994.

BAZANELLA, A. S.; GOMES DA SILVA JR., J. M. **Sistemas de Controle**: princípios e métodos de projeto. 1.ed. Porto Alegre: UFRGS, 2005. 299 p.

BELL, D. A. et al. Design criteria for obstacle avoidance in a shared-control system. In: RESNA INTERNATIONAL CONFERENCE, 1994, Nashville. **Proceedings...** Arlington: RESNA, 1994. p.17–24.

BLOCH, A.; REYHANOGLU, M.; MCCLAMROCH, N. Control and stabilization of nonholonomic dynamic systems. **IEEE Transactions on Automatic Control**, New York, v.37, n.11, p.1746–1757, 1992.

BOK, Y.; CHOI, D.-G.; KWEON, I. S. Generalized laser three-point algorithm for motion estimation of camera-laser fusion system. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2013, Karlsruhe. **Proceedings...** New York: IEEE, 2013. p.2880–2887.

BORENSTEIN, J.; EVERETT, H. R.; FENG, L. **Where am I?**: sensors and methods for mobile robot positioning. Ann Arbor: University of Michigan, 1996. 282 p.

BORENSTEIN, J.; FENG, L. **UMBmark**: a method for measuring, comparing, and correcting dead-reckoning errors in mobile robots. Ann Arbor: University of Michigan, 1994. 81 p.

BORENSTEIN, J.; KOREN, Y. Real-time obstacle avoidance for fast mobile robots. **IEEE Transactions on Systems, Man, and Cybernetics**, New York, v.19, n.5, p.1179–1187, Sept. 1989.

BORENSTEIN, J.; KOREN, Y. Histogramic in-motion mapping for mobile robot obstacle avoidance. **IEEE Transactions on Robotics and Automation**, New York, v.7, n.4, p.535–539, 1991.

BORENSTEIN, J.; KOREN, Y. The vector field histogram-fast obstacle avoidance for mobile robots. **IEEE Transactions on Robotics and Automation**, New York, v.7, n.3, p.278–288, June 1991.

BORENSTEIN, J.; KOREN, Y. Noise rejection for ultrasonic sensors in mobile robot applications. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1992, Nice. **Proceedings...** New York: IEEE, 1992. p.1727–1732.

BORENSTEIN, J.; RASCHKE, U. Real-time Obstacle Avoidance for Non-Point Mobile Robots. In: WORLD CONFERENCE ON ROBOTICS RESEARCH, 4., 1991, Pittsburgh. **Proceedings...** Dearborn: SME, 1991. p.1–9.

BOSCH. **CAN Specification**: version 2.0. Stuttgart: Rober Bousch GmbH, 1991. 72 p.

BOZMA, O.; KUC, R. A physical model-based analysis of heterogeneous environments using sonar-ENDURA method. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.16, n.5, p.497–506, May 1994.

BRASIL. **Cartilha do Censo 2010**: pessoas com deficiências. Brasília: Secretaria Nacional de Promoção dos Direitos da Pessoa com Deficiência (SNPD), 2012. 36 p.

BRASIL. **Estatuto da Pessoa com Deficiência**. 2.ed. Brasília: Senado Federal, 2016. 65 p.

BROCKETT, R. Asymptotic stability and feedback stabilization. In: BROCKETT, R. W.; MILLMANN, R. S.; SUSSMANN, H. J. (Ed.). **Differential Geometric Control Theory**. 1.ed. Boston: Birkhauser, 1983. p.181–191.

BROCKETT, R. W. Control Theory and Singular Riemannian Geometry. In: HILTON, P. J.; YOUNG, G. S. (Ed.). **New Directions in Applied Mathematics**. 1.ed. New York: Springer-Verlag, 1982.

BROWN, K.; INIGO, R.; JOHNSON, B. Design, implementation, and testing of an adaptable optimal controller for an electric wheelchair. In: CONFERENCE RECORD OF THE IEEE INDUSTRY APPLICATIONS SOCIETY ANNUAL MEETING, 1989, San Diego. **Proceedings...** New York: IEEE, 1989. p.2332–2339.

CALABRESE, L. **Robust Odometry , Localization and Autonomous Navigation on a Robotic Wheelchair**. 2013. 109 p. Tesi di Laurea — POLITECNICO DI MILANO, Milano, 2013.

CAMPION, G.; BASTIN, G.; DANDREA-NOVEL, B. Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. **IEEE Transactions on Robotics and Automation**, New York, v.12, n.1, p.47–62, 1996.

CAMPION, G.; CHUNG, W. Wheeled Robots. In: SICILIANO, B.; KHATIB, O. (Ed.). **Springer Handbook of Robotics**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p.391–410.

CAMPION, G.; D'ANDREA-NOVEL, B.; BASTIN, G. Modelling and state feedback control of nonholonomic mechanical systems. In: IEEE CONFERENCE ON DECISION AND CONTROL, 30., 1991, Brighton. **Proceedings...** New York: IEEE, 1991. p.1184–1189.

CANOZ, V. et al. Embedded hardware for closing the gap between research and industry in the assistive powered wheelchair market. In: IEEE/SICE INTERNATIONAL SYMPOSIUM ON SYSTEM INTEGRATION (SII), 2016, Sapporo. **Proceedings...** New York: IEEE, 2016. p.108–113.

CANUDAS DE WIT, C.; SICILIANO, B.; BASTIN, G. (Ed.). **Theory of Robot Control**. London: Springer London, 1996. 392 p.

CANUDAS DE WIT, C.; SØRDALEN, O. J. Exponential Stabilization of Mobile Robots with Nonholonomic Constraints. **IEEE Transactions on Automatic Control**, New York, v.37, n.11, p.1791–1797, 1992.

CARELLI, R.; OLIVEIRA FREIRE, E. Corridor navigation and wall-following stable control for sonar-based mobile robots. **Robotics and Autonomous Systems**, New York, v.45, n.3-4, p.235–247, Dec. 2003.

CELESTE, W. C. et al. Dynamic model and control structure for an autonomous wheelchair. In: IEEE INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS, 2008, Cambridge. **Proceedings...** New York: IEEE, 2008. p.1359–1364.

CHEN, M. et al. Real-time 3D mapping using a 2D laser scanner and IMU-aided visual SLAM. In: IEEE INTERNATIONAL CONFERENCE ON REAL-TIME COMPUTING AND ROBOTICS (RCAR), 2017, Okinawa. **Proceedings...** New York: IEEE, 2017. p.297–302.

CHWA, D. Sliding-Mode Tracking Control of Nonholonomic Wheeled Mobile Robots in Polar Coordinates. **IEEE Transactions on Control Systems Technology**, New York, v.12, n.4, p.637–644, July 2004.

COELHO, A. A. R.; COELHO, L. d. S. **Identificação de Sistemas Dinâmicos Lineares**. 1.ed. Florianópolis: UFSC, 2004. 181 p.

CORRADINI, M.; ORLANDO, G. Control of mobile robots with uncertainties in the dynamical model: a discrete time sliding mode approach with experimental results. **Control Engineering Practice**, New York, v.10, n.1, p.23–34, Jan. 2002.

D'ANDREA-NOVEL, B.; BASTIN, G.; CAMPION, G. Dynamic feedback linearization of nonholonomic wheeled mobile robots. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1992, Nice. **Proceedings...** New York: IEEE, 1992. p.2527–2532.

D'ANDRÉA-NOVEL, B.; CAMPION, G.; BASTIN, G. Control of Nonholonomic Wheeled Mobile Robots by State Feedback Linearization. **The International Journal of Robotics Research**, Thousand Oaks, v.14, n.6, p.543–559, Dec. 1995.

DE LA CRUZ, C.; BASTOS, T. F.; CARELLI, R. Adaptive motion control law of a robotic wheelchair. **Control Engineering Practice**, New York, v.19, n.2, p.113–125, Feb. 2011.

DE LA CRUZ, C.; CARELLI, R. Dynamic Modeling and Centralized Formation Control of Mobile Robots. In: ANNUAL CONFERENCE ON IEEE INDUSTRIAL ELECTRONICS, 32., 2006, Paris. **Proceedings...** New York: IEEE, 2006. p.3880–3885.

DE LA CRUZ, C.; CELESTE, W. C.; BASTOS, T. F. A robust navigation system for robotic wheelchairs. **Control Engineering Practice**, New York, v.19, n.6, p.575–590, June 2011.

DE LA CRUZ, C.; CELESTE, W. C.; BASTOS, T. F. Dynamic Model-Based Adaptive Posture Controller for Robotic Wheelchairs. **Journal of Medical and Biological Engineering**, Heidelberg, v.32, n.1, p.61–69, 2012.

DHAOUADI, R.; HATAB, A. A. Dynamic Modelling of Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: a unified framework. **Advances in Robotics & Automation**, Hyderabad, v.02, n.02, 2013.

DING, D.; COOPER, R. Electric powered wheelchairs. **IEEE Control Systems Magazine**, New York, v.25, n.2, p.22–34, Apr. 2005.

DO, K.; JIANG, Z.; PAN, J. Simultaneous Tracking and Stabilization of Mobile Robots: an adaptive approach. **IEEE Transactions on Automatic Control**, New York, v.49, n.7, p.1147–1152, July 2004.

DUDEK, G.; JENKIN, M. Inertial Sensors, GPS, and Odometry. In: SICILIANO, B.; KHATIB, O. (Ed.). **Springer Handbook of Robotics**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p.477–490.

DUŠEK, F.; HONC, D.; ROZSÍVAL, P. Mathematical model of differentially steered mobile robot. In: INTERNATIONAL CONFERENCE ON PROCESS CONTROL, 18., 2011, Tatranská Lomnica. **Proceedings...** Bratislava: STU, 2011. p.221–229.

ENCARNAÇÃO, P.; PASCOAL, A. Combined Trajectory Tracking and Path Following Control For Dynamic Wheeled Mobile Robots. **IFAC Proceedings Volumes**, New York, v.35, n.1, p.451–456, 2002.

FIERRO, R.; LEWIS, F. Control of a nonholonomic mobile robot: backstepping kinematics into dynamics. In: IEEE CONFERENCE ON DECISION AND CONTROL, 34., 1995, New Orleans. **Proceedings...** New York: IEEE, 1995. p.3805–3810.

FILGUEIRA, P. N. D. S. **Robotização de Uma Cadeira de Rodas**. 2011. 305 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Espirito Santo, Vitória, 2011.

FREEDOM, F. **Cadeira de Rodas Freedom Compact 20**. Disponível em: <http://www.freedom.ind.br/produto/saude/cadeiras-de-rodas-motorizadas/freedom-compact/>. Acesso em: 20 fev. 2018.

FU, K. S.; GONZALES, R. C.; LEE, C. S. G. **Robotics**: control, sensing, vision, and intelligence. 1.ed. New York: McGraw-Hill, 1987. 580 p.

GEIGER, A.; ZIEGLER, J.; STILLER, C. StereoScan: dense 3d reconstruction in real-time. In: IEEE INTELLIGENT VEHICLES SYMPOSIUM, 4., 2011, Baden-Baden. **Proceedings...** New York: IEEE, 2011. n.Iv, p.963–968.

GONÇALVES, D. M. d. S. **RobChair 2.0**: simultaneous localization and mapping and hardware/software frameworks. 2013. 99 p. Thesis (Master of Science in Electrical Computer Engineering) — Universidade de Coimbra, Coimbra, 2013.

GUIZILINI, V. C. **Localização e mapeamento simultâneos com auxílio visual**. 2008. 114 p. Dissertação (Mestrado em Engenharia) — Universidade de São Paulo, São Paulo, 2008.

HAMANAKA, M. H. M. O. **Projeto e Desenvolvimento de Circuito de Controle para Cadeira de Rodas**. 2002. 81 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Estadual de Campinas, Campinas, 2002.

HUANG, A. S. et al. Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera. In: CHRISTENSEN, H.; KHATIB, O. (Ed.). **Robotics Research**. 1.ed. Cham: Springer, 2017. p.235–252. (Springer Tracts in Advanced Robotic, v.100).

HUANG, J. et al. Adaptive output feedback tracking control of a nonholonomic mobile robot. **Automatica**, New York, v.50, n.3, p.821–831, Mar. 2014.

ISIDORI, A. **Nonlinear Control Systems**. 3.ed. London: Springer-Verlag London, 1995. (Communications and Control Engineering).

JAFFE, D. L. An ultrasonic head position interface for wheelchair control. **Journal of Medical Systems**, Dordrecht, v.6, n.4, p.337–342, Aug. 1982.

JOHNSON, B. W.; AYLOR, J. H. Dynamic Modeling of an Electric Wheelchair. **IEEE Transactions on Industry Applications**, New York, v.IA-21, n.5, p.1284–1293, Sept. 1985.

JOSEPH, L. **Mastering ROS for Robotics Programming**. 1.ed. Birmingham: Packt Publishing, 2015. 480 p.

KANAYAMA, Y. et al. A stable tracking control method for an autonomous mobile robot. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1990, Cincinnati. **Proceedings...** New York: IEEE, 1990. p.384–389.

KILLOUGH, S.; PIN, F. Design of an omnidirectional and holonomic wheeled platform prototype. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1992, Nice. **Proceedings...** New York: IEEE, 1992. p.84–90.

KITAGAWA, H. et al. Motion Control Of Omnidirectional Wheelchair Considering Patient Comfort. **IFAC Proceedings Volumes**, New York, v.35, n.1, p.437–442, 2002.

KUC, R.; SIEGEL, M. W. Physically Based Simulation Model for Acoustic Sensor Robot Navigation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, New York, v.PAMI-9, n.6, p.766–778, Nov. 1987.

KÜHNE, F. **Controle Preditivo de Robôs Móveis Não Holonômicos**. 2005. 155 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2005.

KUHNE, F.; LAGES, W.; GOMES DA SILVA JR., J. Point stabilization of mobile robots with nonlinear model predictive control. In: IEEE INTERNATIONAL CONFERENCE MECHATRONICS AND AUTOMATION, 2005, Niagara Falls. **Proceedings...** New York: IEEE, 2005. p.1163–1168.

KUNDU, A. S. et al. Design and Performance Evaluation of 4 Wheeled Omni Wheelchair with Reduced Slip and Vibration. **Procedia Computer Science**, Amsterdam, v.105, p.289–295, 2017.

LAGES, W. F. **Controle e Estimação de Posição e Orientação de Robôs Móveis**. 1998. 180 p. Tese de Doutorado — Instituto Tecnológico de Aeronáutica, São José dos Campos, 1998.

LAGES, W. F. Implementation of Real-Time Joint Controllers. In: KOUBAA, A. (Ed.). **Robot Operating System (ROS)**. 1.ed. Cham: Springer International Publishing, 2016. p.671–702.

LAGES, W. F. Parametric Identification of the Dynamics of Mobile Robots and Its Application to the Tuning of Controllers in ROS. In: KOUBAA, A. (Ed.). **Robot Operating System (ROS)**. 1.ed. Cham: Springer International Publishing, 2017. p.191–229.

LAGES, W. F.; BRACARENSE, A. Q. Robot retrofitting: a perspective to small and medium size entreprizes. In: AUSTRIAN BRAZILIAN AUTOMATION DAY, 3., 2003, São Bernardo do Campo. **Proceedings...** São Bernardo do Campo: RECOPE/MANET, 2003. p.1–11.

LAGES, W. F.; HEMERLY, E. M. Adaptive Linearizing Control of Mobile Robots. **IFAC Proceedings Volumes**, New York, v.31, n.31, p.23–28, 1998.

LEAMAN, J.; MANH LA, H. A Comprehensive Review of Smart Wheelchairs: past, present, and future. **IEEE Transactions on Human-Machine Systems**, New York, v.47, n.4, p.486–499, Aug. 2017.

LEONARD, J. J.; DURRANT-WHYTE, H. F. **Directed Sonar Sensing for Mobile Robot Navigation**. Boston, MA: Springer US, 1992.

LEVINE, S. P.; BORENSTEIN, J.; KOREN, Y. The Navchair control system for automatic assistive wheelchair Navigation. In: RESNA ANNUAL CONFERENCE, 13., 1990, Washington. **Proceedings...** Arlington: RESNA, 1990. p.193–194.

LEVINE, S. P. et al. The NavChair Assistive Wheelchair Navigation System. **IEEE Transactions on Rehabilitation Engineering**, New York, v.7, n.4, p.443–451, 1999.

LIMA, E. J. et al. Sensoring for Retrofitting of an Industrial Robot. **IFAC Proceedings Volumes**, New York, v.37, n.4, p.545–550, 2004.

LJUNG, L. **System identification**: theory for the user. 2.ed. Upper Saddle River: Prentice Hall PTR, 1999. 609 p.

LOZAC'H, Y. et al. Design and evaluation of head unit for wheelchair control by quadriplegic patients. **Canadian Medical Association Journal**, Ottawa, v.115, n.5, p.429–432, 1976.

LUO, R.; CHEN, T.; HSIEN LIN, M. Automatic guided intelligent wheelchair system using hierarchical grey-fuzzy motion decision-making algorithms. In: IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS. HUMAN AND ENVIRONMENT FRIENDLY ROBOTS WITH HIGH INTELLIGENCE AND EMOTIONAL QUOTIENTS (CAT. NO.99CH36289), 1999, Kyongju. **Proceedings...** New York: IEEE, 1999. v.2, p.900–905.

LUO, R.; LIN, S.-C.; CHI LAI, C. Indoor autonomous mobile robot localization using natural landmark. In: ANNUAL CONFERENCE OF IEEE INDUSTRIAL ELECTRONICS, 34., 2008, Orlando. **Proceedings...** New York: IEEE, 2008. p.1626–1631.

MACIEL, E. H. **Desenvolvimento de um modelo simplificado dos membros inferiores de um robô bípede utilizando ros**. 2014. 105 p. Dissertação de Mestrado — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2014.

MADARASZ, R. et al. The design of an autonomous vehicle for the disabled. **IEEE Journal on Robotics and Automation**, New York, v.2, n.3, p.117–126, 1986.

MARQUES, P. J. **Proposta de sistema de determinação da posição de cadeira de rodas em ambiente inteligente**. 2014. 140 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, 2014.

MARTINS, F. N. **Modelagem e Compensação da Dinâmica de Robôs móveis e sua alicação em controle de formação**. 2009. 194 p. Tese (Doutorado em Engenharia Elétrica) — Universidade Federal do Espírito Santo, Vitória, 2009.

MARTINS, F. N. et al. An adaptive dynamic controller for autonomous mobile robot trajectory tracking. **Control Engineering Practice**, New York, v.16, n.11, p.1354–1363, Nov. 2008.

MARTINS, F. N.; SARCINELLI-FILHO, M.; CARELLI, R. A Velocity-Based Dynamic Model and Its Properties for Differential Drive Mobile Robots. **Journal of Intelligent & Robotic Systems**, Dordrecht, v.85, n.2, p.277–292, Feb. 2017.

MASCARO, S.; SPANO, J.; ASADA, H. A reconfigurable holonomic omnidirectional mobile bed with unified seating (RHOMBUS) for bedridden patients. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1997, Albuquerque. **Proceedings...** New York: IEEE, 1997. v.2, p.1277–1282.

Mathworks, T. M. **Documentation**: tfest - Transfer function estimation. Disponível em: <`https://www.mathworks.com/help/ident/ref/tfest.html`>. Acesso em: 20 fev. 2018.

MEEUSSEN, W. **The ros_control Wiki**. Disponível em: <`http://wiki.ros.org/ros_control`>. Acesso em: 20 fev. 2018.

MEEUSSEN, W. **The ros_controllers Wiki**. Disponível em: <`http://wiki.ros.org/ros_controllers`>. Acesso em: 20 fev. 2018.

MICROCHIP. **dsPIC30F4011/4012 Data Sheet**. Chandler: Microchip Technology, 2010.

MILLER, D. P.; SLACK, M. G. Design and testing of a low-cost robotic wheelchair prototype. **Autonomous Robots**, Boston, v.2, n.1, p.77–88, 1995.

MOHAN, N.; UNDELAND, T. M.; ROBBINS, W. P. **Power Electronics, Converters, Applications and Design**. 2.ed. New York: John Wiley, 1995. 802 p.

MUIR, P. F.; NEUMAN, C. P. Kinematic modeling of wheeled mobile robots. **Journal of Robotic Systems**, Hoboken, v.4, n.2, p.281–340, Apr. 1987.

MUIR, P.; NEUMAN, C. Kinematic modeling for feedback control of an omnidirectional wheeled mobile robot. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1987, Raleigh. **Proceedings...** New York: IEEE, 1987. v.4, p.1772–1778.

MURATA, S.; HIROSE, T. Onboard locating system using real-time image processing for a self-navigating vehicle. **IEEE Transactions on Industrial Electronics**, New York, v.40, n.1, p.145–154, 1993.

NASCIMENTO JÚNIOR, A. **Robotização de uma cadeira de rodas motorizada**: arquitetura, modelos, controle e aplicações. 2016. 122 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Estadual de Campinas, Campinas, 2016.

NEGAHDARIPOUR, S.; HAYASHI, B.; ALOIMONOS, Y. Direct motion stereo for passive navigation. **IEEE Transactions on Robotics and Automation**, New York, v.11, n.6, p.829–843, 1995.

NELLES, O. **Nonlinear System Identification**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

NGUYEN, N.; NGUYEN, H. T.; SU, S. Neuro-sliding mode multivariable control of a powered wheelchair. In: ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE ENGINEERING IN MEDICINE AND BIOLOGY SOCIETY, 30., 2008, Vancouver. **Proceedings...** New York: IEEE, 2008. p.3471–3474.

OGATA, K. **Engenharia de Controle Moderno**. 5.ed. São Paulo: PEARSON, 2010. 824 p.

ONYANGO, S. et al. Dynamic control of powered wheelchair with slip on an incline. In: INTERNATIONAL CONFERENCE ON ADAPTIVE SCIENCE & TECHNOLOGY (ICAST), 2., 2009, Accra. **Proceedings...** New York: IEEE, 2009. p.278–283.

ORIOLO, G.; DE LUCA, A.; VENDITTELLI, M. WMR control via dynamic feedback linearization: design, implementation, and experimental validation. **IEEE Transactions on Control Systems Technology**, New York, v.10, n.6, p.835–852, Nov. 2002.

PETROV, P. Modeling and adaptive path control of a differential drive mobile robot. In: WSEAS INTERNATIONAL CONFERENCE ON AUTOMATIC CONTROL, MODELLING AND SIMULATION, 12., 2010, Catania. **Proceedings...** Wisconsin: WSEAS, 2010. p.403–408.

POMET, J.-B. et al. A hybrid strategy for the feedback stabilization of nonholonomic mobile robots. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1992, Nice. **Proceedings...** New York: IEEE, 1992. p.129–134.

POPOV, V. M. **Hyperstability of Control Systems**. 1.ed. Nova York: Springer-Verlag Berlin Heidelberg, 1973. 400 p.

QUIGLEY, M. et al. ROS: an open-source robot operating system. In: ICRA WORKSHOP ON OPEN SOURCE SOFTWARE, 2009, Kobe. **Proceedings...** New York: IEEE, 2009. p.6.

QUIGLEY, M.; GERKEY, B.; SMART, W. D. **Programming Robots with ROS**: a practical introduction to the robot operating system. 1.ed. Sebastopol: O'Reilly Media, 2015. 448 p.

RÖFER, T. Building consistent laser scan maps. In: EUROPEAN WORKSHOP ON ADVANCED MOBILE ROBOTS, 4., 2001. **Proceedings...** Lund: Lund University Cognitive Studies, 2001. p.83–90.

ROMERO, R. A. F. et al. **Robótica Móvel**. 1.ed. Rio de janeiro: LTC, 2014. 316 p.

ROS, R. O. S. **The hardware_interface Wiki - Setting up a new robot**. Disponível em: <https://github.com/ros-controls/ros_control/wiki/hardware_interface>. Acesso em: 20 fev. 2018.

ROS, R. O. S. **The controller_interface Wiki - Writing a new controller**. Disponível em: <https://github.com/ros-controls/ros_control/wiki/controller_interface>. Acesso em: 20 fev. 2018.

SAMSON, C. Velocity and torque feedback control of a nonholonomic cart. In: CANUDAS DE WIT, C. (Ed.). **Advanced Robot Control**. Berlin/Heidelberg: Springer-Verlag, 1991. p.125–151.

SAMSON, C. Control of chained systems application to path following and time-varying point-stabilization of mobile robots. **IEEE Transactions on Automatic Control**, New York, v.40, n.1, p.64–77, 1995.

SAMSON, C.; AIT-ABDERRAHIM, K. Feedback control of a nonholonomic wheeled cart in Cartesian space. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1991, Sacramento. **Proceedings...** New York: IEEE, 1991. p.1136–1141.

SANTINI, D. C. **Arquitetura Aberta para Controle de Robôs Manipuladores**. 2009. 123 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

SARKAR, N.; XIAOPING, Y.; KUMAR, V. Dynamic path following: a new control algorithm for mobile robots. In: IEEE CONFERENCE ON DECISION AND CONTROL, 32., 1993, San Antonio. **Proceedings...** New York: IEEE, 1993. p.2670–2675.

SCARAMUZZA, D.; FRAUNDORFER, F. Visual Odometry [Tutorial]. **IEEE Robotics & Automation Magazine**, New York, v.18, n.4, p.80–92, Dec. 2011.

SICILIANO, B. et al. **Robotics**: modelling, planning and control. 1.ed. London: Springer-Verlag London, 2009. 632 p.

SIDEK, N.; SARKAR, N. Dynamic Modeling and Control of Nonholonomic Mobile Robot with Lateral Slip. In: INTERNATIONAL CONFERENCE ON SYSTEMS (ICONS 2008), 3., 2008, Cancun. **Proceedings...** New York: IEEE, 2008. p.35–40.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots**. 2.ed. Cambridge: MIT, 2011. 472 p.

SILVA, R. L. **Desenvolvimento de uma Interface Homem-Máquina aplicada a Uma Cadeira de Rodas Robótica por meio de PDA**. 2007. 145 p. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Espírito Santo, Vitória, 2007.

SIMPSON, R. C. Smart wheelchairs: a literature review. **The Journal of Rehabilitation Research and Development**, Washington DC, v.42, n.4, p.423, 2005.

SLOTINE, J.-J. E.; LI, W. **Applied Nonlinear Control**. 1.ed. Englewood Cliffs: Prentice-Hall, 1991. 461 p.

SOATTO, S.; FREZZA, R.; PERONA, P. Motion estimation via dynamic vision. **IEEE Transactions on Automatic Control**, New York, v.41, n.3, p.393–413, Mar. 1996.

SÖDERSTRÖM, T.; STOICA, P. **System Identification**. 1.ed. Upper Saddle River: Prentice Hall, 1989. 612 p.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. **Robot Modeling and Control**. 1.ed. New York: John Wiley & Sons, 2005. 496 p.

SUCAN, I. **The ROS URDF Wiki**. Disponível em: <http://wiki.ros.org/urdf>. Acesso em: 20 fev. 2018.

TEEL, A. R.; MURRAY, R. M.; WALSH, G. C. Non-holonomic control systems: from steering to stabilization with sinusoids. **International Journal of Control**, Abingdon, v.62, n.4, p.849–870, Oct. 1995.

TZAFESTAS, S. **Introduction to Mobile Robot Control**. 1.ed. New York: Elsevier, 2013. 750 p.

WENG, J.; COHEN, P.; REBIBO, N. Motion and structure estimation from stereo image sequences. **IEEE Transactions on Robotics and Automation**, New York, v.8, n.3, p.362–382, June 1992.

WHILL, W. I. **Model A - Personal Mobility Device - Personal EV**. Disponível em: <http://whill.us/model-a-personal-mobility-device-personal-ev>. Acesso em: 20 fev. 2018.

WOODS, B.; WATSON, N. A Short History of Powered Wheelchairs. **Assistive Technology**, Milton Park, v.15, n.2, p.164–180, Dec. 2003.

WORLD HEALTH ORGANIZATION. **World report on disability**. Malta, 2011. 350 p.

WORLD HEALTH ORGANIZATION. **WHO global disability action plan 2014-2021:** better health for all people with disability. Geneva, 2015. 25 p.

WU, W. et al. Adaptive exponential stabilization of mobile robots with uncertainties. In: IEEE CONFERENCE ON DECISION AND CONTROL (CAT. NO.99CH36304), 38., 1999, Phoenix. **Proceedings...** New York: IEEE, 1999. v.4, p.3484–3489.

YANCO, H. A. **Shared user-computer control of a robotic wheelchair system**. 2000. 152 p. PHD Thesis (Doctor of Philosophy) — Massachusetts Institute of Technology, Cambridge, 2000.

YUSIF, S.; SOAR, J.; HAFEEZ-BAIG, A. Older people, assistive technologies, and the barriers to adoption: a systematic review. **International Journal of Medical Informatics**, New York, v.94, p.112–116, Oct. 2016.

ZHANG, J.; SINGH, S. Visual-lidar odometry and mapping: low-drift, robust, and fast. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), 2015, Seattle. **Proceedings...** New York: IEEE, 2015. p.2174–2181.

ZHONG, X.; ZHOU, Y.; LIU, H. Design and recognition of artificial landmarks for reliable indoor self-localization of mobile robots. **International Journal of Advanced Robotic Systems**, Thousand Oaks, v.14, n.1, p.1–13, Jan. 2017.

APÊNDICE A DECLARAÇÃO DA CLASSE TWILROBOTHW

Este apêndice apresenta uma versão simplificada da classe `TwilRobotHW`, derivada da classe `hardware_interface::RobotHW`.

Listagem A.1: Versão simplificada da classe `TwilRobotHW`.

```
#include <ros/ros.h>
#include <hardware_interface/joint_state_interface.h>
#include <hardware_interface/robot_hw.h>

#include "aic.h"
#include "twil_hardware_interfaces/joint_command_extended_interface.h"

namespace twil_hardware
{

class TwilRobotHW : public hardware_interface::RobotHW
{
  public:
    TwilRobotHW();
    ~TwilRobotHW();
    bool init(ros::NodeHandle &root_nh, ros::NodeHandle &robot_hw_nh);
    void read(const ros::Time& time, const ros::Duration& period);
    void write(const ros::Time& time, const ros::Duration& period);

  private:
    ros::NodeHandle node_;
    hardware_interface::JointStateInterface joint_state_interface;
    twil_hardware_interfaces::VoltageJointInterface voltage_interface;

    double cmd[2]; // joint command variables
    double pos[2], vel[2], eff[2]; // joint variables

    aic *joint_hw[2];
    aic_displacement_msg_t joint_sensor[2];
    aic_comm_config_t connection_parameters[2];
};
}
```

APÊNDICE B IMPLEMENTAÇÃO DO CONTROL_LOOP_NODE

Este apêndice apresenta uma versão simplificada do `control_loop_node` utilizado com o robô móvel Twil.

Listagem B.1: Versão simplificada do `twil_control_loop_node`.

```
#include <ros/ros.h>
#include <std_msgs/Duration.h>
#include <controller_manager/controller_manager.h>
#include "twil_hardware/twil_hardware.h"

int main(int argc, char **argv)
{
  ros::init(argc, argv, "twil_control_loop_node");
  ros::NodeHandle nh;
  // parameters from ROS parameter server
  ...
  // scheduler policy configuration for realtime
  ...
  ros::Duration dt;
  twil_hardware::TwilRobotHW real_robot_hw;
  real_robot_hw.init(nh, pnh);
  controller_manager::ControllerManager cm(&real_robot_hw);
  ros::AsyncSpinner spinner(1); spinner.start();
  ros::Rate loop_rate(control_rate);

  ros::Time last_time = ros::Time::now(), current_time;
  while (ros::ok())
  {
    current_time = ros::Time::now();
    dt = current_time - last_time;
    last_time = current_time;

    real_robot_hw.read(current_time,dt);
    cm.update(current_time, dt);
    real_robot_hw.write(current_time,dt);

    loop_rate.sleep();
  }
  spinner.stop();
  real_robot_hw.~twil_hardware();

  return 0;
}
```