



Centro de Tecnologia e Urbanismo  
Departamento de Engenharia Elétrica

**Edno Gentilho Junior**

# **Instrumentação e Controle PID Embarcado para Cadeiras de Rodas Elétricas Acionada por Sopro e Sucção**

Dissertação apresentada ao Programa de  
Pós-Graduação em Engenharia Elétrica  
da Universidade Estadual de Londrina  
para obtenção do Título de Mestre em  
Engenharia Elétrica.

Londrina, PR  
2014



**Edno Gentilho Junior**

# **Instrumentação e Controle PID Embarcado para Cadeiras de Rodas Elétricas Acionada por Sopro e Sucção**

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Estadual de Londrina para obtenção do Título de Mestre em Engenharia Elétrica.

Área de concentração: Sistemas Eletrônicos  
Especialidade: Sistemas de Controle

Orientador:  
Prof. Dr. Ruberlei Gaino

Londrina, PR  
2014

### **Ficha Catalográfica**

Gentilho Junior, Edno

Instrumentação e Controle PID Embarcado para Cadeiras de Rodas Elétricas Acionada por Sopro e Sucção. Londrina, PR, 2014. 97 p.

Dissertação (Mestrado) – Universidade Estadual de Londrina, PR. Centro de Tecnologia e Urbanismo. Programa de Pós-Graduação em Engenharia Elétrica .

1. Sistemas de Controle. 2. Instrumentação Eletrônica. 3. Engenharia Biomédica I. Universidade Estadual de Londrina. Departamento de Engenharia Elétrica. Centro de Tecnologia e Urbanismo. Programa de Pós-Graduação em Engenharia Elétrica . II. Título.

**Edno Gentilho Junior**

# **Instrumentação e Controle PID Embarcado para Cadeiras de Rodas Elétricas Acionada por Sopro e Sucção**

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Estadual de Londrina para obtenção do Título de Mestre em Engenharia Elétrica.

Área de concentração: Sistemas Eletrônicos  
Especialidade: Sistemas de Controle

## **Comissão Examinadora**

---

Prof. Dr. Ruberlei Gaino  
Depto. de Engenharia Elétrica  
Universidade Estadual de Londrina  
Orientador

---

Prof. Dr. Márcio Roberto Covacic  
Depto. de Engenharia Elétrica  
Universidade Estadual de Londrina

---

Prof. Dr. Paulo Laerte Natti  
Depto. de Matemática  
Universidade Estadual de Londrina

Londrina, 22 de agosto de 2014



“Deixe o futuro dizer a verdade e avaliar cada um de acordo com seus trabalhos  
e suas conquistas.”

*Nikola Tesla*

# Agradecimentos

Primeiramente, agradeço a Deus por proporcionar condições para finalizar este trabalho. Ao orientador Ruberlei Gaino, aos Professores Márcio Roberto Covacic, Leonimer Flávio de Melo, Taufik Abrão, Newton da Silva e todos os demais com quem tive o privilégio de aprender muitas coisas.

Aos meus queridos pais Edno e Aparecida, pelo grande apoio e compreensão neste período.

À minha namorada Patrícia pela compreensão e apoio em tantos momentos de dificuldade.

A todos os meus amigos e colegas de laboratório, que tiveram grande importância no desenvolvimento deste trabalho. Em especial, aos amigos Julio, Kleber e Aislan, que estiveram sempre próximos, ajudando-me. Não posso deixar de agradecer aos meus amigos da Unesp de Ilha Solteira, Marcelo Sanches, Marcos Junqueira, Renan Kozan e Luiz (Perdido), que contribuíram grandemente aos meus conhecimentos. Também ao Prof. Aparecido Augusto de Carvalho, pela recepção em seu laboratório de pesquisa.

E por fim, agradeço à Fundação Araucária do Paraná e CAPES pelo apoio financeiro.

# Resumo

Ultimamente, os novos conceitos de acessibilidade vêm estimulando diversas organizações no mundo e a comunidade científica a interessar-se no desenvolvimento de cadeiras de rodas que possam habilitar deficientes físicos a terem uma mobilidade maior. A principal necessidade de adaptação nesse tipo de equipamento é a adaptação de interfaces diferenciadas para acionamento que permitem a manipulação por pessoas com baixa mobilidade motora e um controle eficaz, garantindo movimentos suaves e seguros. A proposta deste trabalho é o projeto de implementação de um controle PID (Proporcional, Integral e Derivativo) embarcado em DSP (Processador Digital de Sinais) para uma cadeira de rodas com interface de acionamento por sopro e sucção. Este projeto consistiu de três etapas: identificação dos motores elétricos, projeto do controlador PID e implementação do sistema. Foram realizadas simulações do controlador e uma prévia implementação do projeto em uma plataforma em escala reduzida, que viabilizaram a posterior implementação na cadeira de rodas. Com o projeto implementado na cadeira de rodas, obteve-se um controlador simples e eficaz com resultados satisfatórios. Em testes de condução a cadeira apresentou resultados condizentes com o esperado, sendo conduzida por sopro e sucção e apresentando movimentos precisos e suaves.

# Abstract

At present, new concepts of accessibility have stimulated various international organizations and the scientific community to take interest in the development of wheelchairs that can grant physically challenged individuals greater mobility. The primary challenges in adapting this type of equipment are the implementation of different control interfaces that can be manipulated by individuals with limited motor skills and effective control, ensuring smooth and safe movements. The purpose of this study is to design and implement a PID controller embedded in a DSP for a wheelchair with a control interface based on inhaling and exhaling air (sip and puff control). This project includes the following tasks: identifying the electrical motors, designing the PID controller, and implementing the system. Simulations were performed with the controller and a previous implementation of the design on a reduced-scale platform, which permitted later implementation in the wheelchair. With the design implemented in the wheelchair, a simple and effective controller was obtained and yielded satisfactory results. Test maneuvers were performed, and the wheelchair showed results that were consistent with our expectations-guided by sips and puffs of air, the chair's movements were precise and smooth.

# Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

Convenções e Lista de Símbolos

|          |                                                                                                 |           |
|----------|-------------------------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introdução</b>                                                                               | <b>1</b>  |
| 1.1      | Justificativa . . . . .                                                                         | 5         |
| 1.2      | Revisão Bibliográfica . . . . .                                                                 | 5         |
| 1.3      | Metodologia . . . . .                                                                           | 6         |
| 1.4      | Objetivos . . . . .                                                                             | 7         |
| 1.5      | Resultados Preliminares . . . . .                                                               | 7         |
| 1.6      | Organização da Dissertação . . . . .                                                            | 8         |
| <br>     |                                                                                                 |           |
| <b>2</b> | <b>Conceitos Teóricos</b>                                                                       | <b>10</b> |
| 2.1      | Modelo Matemático Linear . . . . .                                                              | 10        |
| 2.2      | Controle de Sistemas . . . . .                                                                  | 12        |
| 2.2.1    | Sistemas de Controle Embarcado . . . . .                                                        | 12        |
| 2.2.2    | Controladores PID . . . . .                                                                     | 14        |
| 2.2.3    | Estabilidade de Polinômios com Dependência Polinomial<br>através do Critério de Routh . . . . . | 16        |
| 2.2.4    | Estabilidade de Sistemas com Realimentação Estática da<br>Saída . . . . .                       | 18        |
| 2.2.5    | Estabilidade de Sistemas com Controladores PID e PI . . . .                                     | 19        |

|          |                                                                            |           |
|----------|----------------------------------------------------------------------------|-----------|
| 2.3      | Identificação de Sistemas . . . . .                                        | 20        |
| 2.3.1    | Conceitos de Identificação de Sistemas . . . . .                           | 21        |
| 2.3.2    | Modelos . . . . .                                                          | 21        |
| 2.3.3    | Modelos Paramétricos . . . . .                                             | 23        |
| 2.3.4    | Modelo ARX . . . . .                                                       | 24        |
| 2.3.5    | O Estimador dos Mínimos Quadrados . . . . .                                | 26        |
| <b>3</b> | <b>Descrição do Projeto Proposto</b>                                       | <b>31</b> |
| 3.1      | Dispositivos Eletrônicos . . . . .                                         | 32        |
| 3.1.1    | DSP <i>Texas Instruments</i> TMS320F28335 . . . . .                        | 32        |
| 3.1.2    | Motores Elétricos e Acionamento de Potência . . . . .                      | 35        |
| 3.1.3    | Sensor <i>Encoder</i> . . . . .                                            | 37        |
| 3.1.4    | Interface de Sopro e Sucção . . . . .                                      | 40        |
| 3.2      | Identificação da Função de Transferência dos Motores . . . . .             | 44        |
| 3.2.1    | Aquisição de Dados . . . . .                                               | 44        |
| 3.2.2    | Obtenção das Funções de Transferências dos Motores com<br>Matlab . . . . . | 51        |
| 3.3      | Implementação do Controle . . . . .                                        | 57        |
| 3.3.1    | Controlador Proposto . . . . .                                             | 58        |
| 3.3.2    | Obtenção dos Ganhos do Controlador PID . . . . .                           | 59        |
| 3.3.3    | Implementação do Controlador no DSP . . . . .                              | 61        |
| <b>4</b> | <b>Resultados</b>                                                          | <b>68</b> |
| 4.1      | Simulações do Controlador PID . . . . .                                    | 68        |
| 4.2      | Resposta do Sistema Implemento no DSP . . . . .                            | 70        |
| <b>5</b> | <b>Conclusão</b>                                                           | <b>75</b> |
|          | <b>Referências</b>                                                         | <b>77</b> |
|          | <b>Apêndice A – Disseminações</b>                                          | <b>81</b> |

|     |                                                           |    |
|-----|-----------------------------------------------------------|----|
| A.1 | Patente Requerida . . . . .                               | 81 |
| A.2 | Publicações Diretamente Relacionadas ao Tema . . . . .    | 81 |
| A.3 | Publicações Relacionadas com a Área de Pesquisa . . . . . | 82 |
| A.4 | Projetos de Pesquisa Envolvidos . . . . .                 | 83 |

|                                                                                               |           |
|-----------------------------------------------------------------------------------------------|-----------|
| <b>Apêndice B – Código do Bloco “ACIONAMENTO”(MatLab) do<br/>Programa Implementado no DSP</b> | <b>84</b> |
|-----------------------------------------------------------------------------------------------|-----------|

|                                                                          |           |
|--------------------------------------------------------------------------|-----------|
| <b>Apêndice C – Programa de Controle Desenvolvido em Linguagem<br/>C</b> | <b>86</b> |
|--------------------------------------------------------------------------|-----------|

# Lista de Figuras

|      |                                                                                     |    |
|------|-------------------------------------------------------------------------------------|----|
| 1.1  | Projeto da Cadeira de Rodas na UEL. . . . .                                         | 3  |
| 1.2  | Diagrama do sistema. . . . .                                                        | 4  |
| 1.3  | Mini protótipo da cadeira de rodas . . . . .                                        | 8  |
| 1.4  | Comparação da simulação e dados captados dos motores em movimento linear . . . . .  | 9  |
| 1.5  | Comparação da simulação e dados captados dos motores em movimento angular . . . . . | 9  |
| 2.1  | Modelo linear da cadeira . . . . .                                                  | 11 |
| 2.2  | V e $\Omega$ em função de $\omega_l$ e $\omega_r$ . . . . .                         | 11 |
| 2.3  | Diagrama de blocos básico de um sistema de controle embarcado. . . . .              | 13 |
| 2.4  | Sistema em malha fechada. . . . .                                                   | 15 |
| 2.5  | Sistema realimentado com controlador $G_c(s)$ . . . . .                             | 18 |
| 2.6  | Fluxograma de Identificação de Sistemas. . . . .                                    | 22 |
| 2.7  | Representação do modelo ARX . . . . .                                               | 26 |
| 3.1  | Diagrama de blocos do desenvolvimento do projeto. . . . .                           | 31 |
| 3.2  | Menu do CCS. . . . .                                                                | 33 |
| 3.3  | Menu de gravação do CCS. . . . .                                                    | 34 |
| 3.4  | Estação de trabalho e DSP TMS320F28335. . . . .                                     | 34 |
| 3.5  | Diagrama funcional do DSP TMS320F28335. . . . .                                     | 36 |
| 3.6  | Motor da cadeira de rodas. . . . .                                                  | 37 |
| 3.7  | Gráfico da curva do motor. . . . .                                                  | 37 |
| 3.8  | Esquema da ponte-H. . . . .                                                         | 38 |
| 3.9  | Foto do circuito eletrônico de controle da cadeira de rodas. . . . .                | 38 |
| 3.10 | a) <i>Encoder</i> incremental, b) <i>Encoder</i> absoluto . . . . .                 | 39 |



|      |                                                                                                        |    |
|------|--------------------------------------------------------------------------------------------------------|----|
| 3.11 | <i>Encoder Avago</i> HEDS-5645 . . . . .                                                               | 39 |
| 3.12 | a) Sensor de pressão <i>Honeywell</i> 24PCEFA6G, b) Circuito esquemático interno do sensor. . . . .    | 40 |
| 3.13 | Diagrama da interface de sopro e sucção com sensor de pressão. . .                                     | 41 |
| 3.14 | Circuito de condicionamento de sinais dos sensores de pressão . .                                      | 41 |
| 3.15 | Diagrama da interface de sopro e sucção com sensor de fluxo. . . .                                     | 42 |
| 3.16 | Sensor de fluxo de ar - AWM2100V. . . . .                                                              | 43 |
| 3.17 | Curva característica do sensor de fluxo de ar. . . . .                                                 | 43 |
| 3.18 | Circuito de condicionamento e ajuste de sinal do sensor de fluxo. .                                    | 44 |
| 3.19 | Esquema de identificação dos motores. . . . .                                                          | 45 |
| 3.20 | a) Placa de aquisição <i>National Instruments</i> PCI6024E, b) Placa de conexões. . . . .              | 45 |
| 3.21 | Diagrama de blocos das camadas do software. . . . .                                                    | 46 |
| 3.22 | Interface Gráfica do programa de aquisição de dados ( <i>LabView</i> ). .                              | 48 |
| 3.23 | Parte do programa responsável pela geração de estímulo elétrico para o motor. . . . .                  | 48 |
| 3.24 | Parte do programa responsável pela leitura de tensão do motor e frequência do <i>encoder</i> . . . . . | 49 |
| 3.25 | Parte do programa responsável pela sincronização dos dados. . . .                                      | 50 |
| 3.26 | Parte do programa responsável pelo armazenamento dos dados. . .                                        | 51 |
| 3.27 | Dados de tensão do motor (coletados e filtrados). . . . .                                              | 52 |
| 3.28 | Dados de velocidade angular do motor (coletados e filtrados). . . .                                    | 53 |
| 3.29 | <i>Toolbox</i> de identificação de sistemas do Matlab . . . . .                                        | 54 |
| 3.30 | Janela para carregar os dados na <i>Toolbox</i> do Matlab . . . . .                                    | 54 |
| 3.31 | Janela para determinação da ordem do modelo . . . . .                                                  | 55 |
| 3.32 | Modelos estimados . . . . .                                                                            | 55 |
| 3.33 | Diagrama principal do sistema implementado. . . . .                                                    | 58 |
| 3.34 | Diagrama de blocos do sistema de controle . . . . .                                                    | 58 |
| 3.35 | Diagrama de blocos do controlador. . . . .                                                             | 59 |

|      |                                                                                           |    |
|------|-------------------------------------------------------------------------------------------|----|
| 3.36 | Fluxograma do Algoritmo de Ganho PID . . . . .                                            | 61 |
| 3.37 | Diagrama de blocos do controlador. . . . .                                                | 64 |
| 3.38 | Fluxograma de funcionamento geral do programa da cadeira de<br>rodas. . . . .             | 65 |
| 3.39 | Diagrama de blocos do programa implementado. . . . .                                      | 66 |
| 3.40 | Diagrama interno do programa de controle. . . . .                                         | 67 |
| 3.41 | Sinal PWM gerado pelo controlador. . . . .                                                | 67 |
| 4.1  | Resposta do motores em movimento linear (simulação). . . . .                              | 69 |
| 4.2  | Resposta do motores em movimento angular (simulação). . . . .                             | 69 |
| 4.3  | Erro de velocidade entre os motores em movimento linear e angular<br>(simulação). . . . . | 70 |
| 4.4  | Resposta do motor direito para movimento linear. . . . .                                  | 71 |
| 4.5  | Resposta do motor esquerdo para movimento linear com carga. . .                           | 72 |
| 4.6  | Erro de velocidade entre os motores em movimento linear com carga.                        | 72 |
| 4.7  | Resposta do motor direito para movimento angular com carga. . .                           | 73 |
| 4.8  | Resposta do motor esquerdo para movimento angular com carga. .                            | 73 |
| 4.9  | Erro de velocidade entre os motores em movimento angular com<br>carga. . . . .            | 74 |

# Lista de Tabelas

|     |                                                                                          |    |
|-----|------------------------------------------------------------------------------------------|----|
| 2.1 | Tabela de Routh. . . . .                                                                 | 17 |
| 3.1 | Dados de funcionamento da interface de sopro e sucção (sensor de pressão de ar). . . . . | 42 |
| 3.2 | Dados de funcionamento da interface de sopro e sucção (sensor de fluxo de ar). . . . .   | 44 |
| 3.3 | Percentual de aproximação dos modelos estimados e do sistema real.                       | 56 |
| 3.4 | Exemplo de faixas de valores para o PID dos motores. . . . .                             | 62 |
| 3.5 | Faixa de valores obtidas pelo algoritmo escolhidas para o projeto.                       | 62 |
| 4.1 | Valores dos ganhos utilizados no controlador PID discreto. . . . .                       | 70 |

# Lista de Abreviaturas

**AR** *Auto Regressivo*

**ARARX** *AutoRegressive AutoRegressive with Exogeneous inputs* - Autorregressivo autorregressivo com entrada exógena

**ARARMAX** *AutoRegressive AutoRegressive Moving Average with Exogenous inputs* - Autorregressivo autorregressivo com média móvel e com entrada exógena

**ARMAX** *AutoRegressive Moving Average with Exogenous inputs* - Autorregressivo com média móvel e entrada exógena

**ARMA** *AutoRegressive Moving Average* - Autorregressivo com média móvel

**ARX** *AutoRegressive Exogeneous* - Autorregressivo com entrada exógena

**CCS** *Code Composer Studio*

**DAQ** *Data Acquisition* - Aquisição de dados

**DSP** *Digital Signal Processor* - Processador digital de sinais

**DVD** *Digital Versatile Disc* - Disco Digital Versátil

**EMG** *Eletromiografia*

**eQEP** *Enhanced Quadrature Encoder Pulse* - Encoder de quadratura de pulsos avançado

**FIFO** *First In First Out* - Primeiro a entrar primeiro a sair

**FIR** *Finite Impulse Response* - Resposta Finita ao Impulso

**FT** *Transfer Function* - Função de transferência

**GPS** *Global Position System* - Sistema de posicionamento global

**IDE** *Integrated Development Environment* - Ambiente de desenvolvimento integrado

**MIMO** *Multiple-Input Multi-Output* - Múltiplas entradas e Múltiplas saídas

**MISO** *Multiple-Input Single-Output* - Múltiplas entradas e uma saída

**PCI** *Peripheral Component Interconnect* - Interconector de Componentes Periféricos

**PID** Proporcional Integral Derivativo

**PWM** *Pulse Width Modulation* - Modulação de Largura de Pulso

**RAM** *Random Access Memory* - Memória de acesso randômico

**RPM** Rotações por Minuto

**SIMO** *Single-Input Multi-Output* - Uma entrada e múltiplas saídas

**SISO** *Single Input and Single output* - Uma entrada e Uma saída

**sRAM** *Static Random Access Memory* - Memória estática de acesso randômico

**UEL** Universidade Estadual de Londrina

**USB** *Universal Serial Bus* - Barramento serial universal

# Convenções e Lista de Símbolos

Os seguintes símbolos serão utilizados:

| símbolo    | descrição                                                   |
|------------|-------------------------------------------------------------|
| $H_l(s)$   | função de transferência do motor esquerdo do mini protótipo |
| $H_r(s)$   | função de transferência do motor direito do mini protótipo  |
| $V$        | velocidade linear da cadeira de rodas                       |
| $\Omega$   | velocidade angular da cadeira de rodas                      |
| $\omega_l$ | velocidade angular da roda esquerda                         |
| $\omega_r$ | velocidade angular da roda direita                          |
| $\theta$   | ângulo de rotação da cadeira de rodas                       |
| $R$        | raio das rodas motorizadas                                  |
| $D$        | distância entre as rodas motorizadas                        |
| $B(s)$     | transformada de Laplace do sinal de realimentação           |
| $E(s)$     | transformada de Laplace do sinal de erro                    |
| $G(s)$     | função de transferência                                     |
| $H(s)$     | elemento de realimentação                                   |
| $C(s)$     | saída do sistema                                            |
| $U(s)$     | saída do controlador                                        |
| $k_p$      | ganho proporcional                                          |
| $k_i$      | ganho integral                                              |
| $k_d$      | ganho derivativo                                            |
| $T_i$      | constante de tempo integrativo                              |
| $T_d$      | constante de tempo derivativo                               |
| $d(s, k)$  | polinômio característico                                    |
| $u(t)$     | lei de controle                                             |
| $e(t)$     | sinal de erro                                               |
| $G_c(s)$   | função de transferência do controlador PID                  |
| $y(k)$     | sinal de saída                                              |
| $\psi$     | vetor de medida                                             |

*continua...*

| símbolo    | descrição                                                                                        |
|------------|--------------------------------------------------------------------------------------------------|
| $e$        | ruídos                                                                                           |
| $na$       | número de polos                                                                                  |
| $nb$       | número de zeros                                                                                  |
| $nk$       | atraso de tempo                                                                                  |
| $H_1(s)$   | função de transferência do motor direito da cadeira de rodas sem carga                           |
| $H_2(s)$   | função de transferência do motor direito da cadeira de rodas com carga                           |
| $H_3(s)$   | função de transferência do motor esquerdo da cadeira de rodas sem carga                          |
| $H_4(s)$   | função de transferência do motor esquerdo da cadeira de rodas com carga                          |
| $H_5(s)$   | função de transferência do motor direito da cadeira de rodas com carga e em movimento giratório  |
| $H_6(s)$   | função de transferência do motor esquerdo da cadeira de rodas com carga e em movimento giratório |
| $H(s)$     | função de transferência simplificada                                                             |
| $\omega$   | velocidade angular                                                                               |
| $Va$       | tensão de armadura                                                                               |
| $K$        | ganho                                                                                            |
| $T$        | constante de tempo do motor                                                                      |
| $m$        | relação de polias                                                                                |
| $V(k)$     | velocidade em rotações por segundo                                                               |
| $\Delta x$ | quantidade de pulsos medidos                                                                     |

# 1 Introdução

A necessidade de maior mobilidade a deficientes físicos, tem estimulado a comunidade científica mundial a interessar-se no desenvolvimento de cadeiras de rodas que possam auxiliá-los nessa tarefa.

No Brasil este quadro não é diferente. É necessário estimular a acessibilidade, pois de acordo com o censo realizado em 2010<sup>1</sup>, há no Brasil, 13.273.969 pessoas com deficiência (IBGE, 2010).

Desta forma, uma contribuição tecnológica direcionada a uma cadeira de roda pode ser feita através da utilização de sistemas eletrônicos e conceitos de automação. Os primeiros modelos de cadeiras de rodas eram mecânicos. Com o tempo, foram desenvolvidos vários projetos para automatização de cadeira de rodas, com o objetivo de minimizar ou eliminar completamente o esforço físico do usuário na tarefa da locomoção (CHONG; HONG, 2008; TIAN; XU, 2009). Para condução de cadeiras de rodas automatizadas, existem várias formas de comando, uma das maneiras mais simples, dependendo do tipo de lesão, é por *joystick*.

Como alternativa para o comando por *joystick*, existem algumas interfaces de entrada adaptadas às incapacidades dos usuários, tais como: acionamento por piscar dos olhos (LIN; CHEN; YANG, 2010); acionamento por Eletromiografia (EMG) (HASHIMOTO; TAKAHASHI; SHIMADA, 2009); acionamento por mudança de posição do paciente na cadeira (ONISHI; KOBAYASHI; WATANABE, 2008); acionamento por leitura da expressão facial (FARIA et al., 2007); acionamento através de movimentos da língua (NAM; ZHAO; CICHOCKI, 2012); acionamento por ondas cerebrais (CHO; VINOD; CHENG, 2009); acionamento por movimentos dos ombros (SATO; YODA; INOUE, 2009); acionamento por sopro e sucção (em desenvolvimento na UEL), entre outras.

As cadeiras de rodas elétricas, em sua maioria, utilizam dois motores de corrente contínua que necessitam de um sistema eletrônico para acionar e ade-

---

<sup>1</sup>Sem distinguir os hemiplégicos, paraplégicos e tetraplégicos, a nomenclatura usada foi “Deficiência Motora”.



quar a velocidade de cada motor, independentemente da superfície em que a cadeira esteja se locomovendo. Para realização dessa tarefa, é necessário um sistema de controle com realimentação que ajusta a velocidade de cada motor de forma autônoma com o objetivo de sincronizar ambos motores. A diferença de velocidade dos motores causa desvio de trajetória. Desta forma, para realizar movimentos lineares, são necessárias velocidades iguais nos motores, e para movimentos angulares, a diferença de velocidade entre os motores deve ser controlada (SOBRINHO et al., 2003).

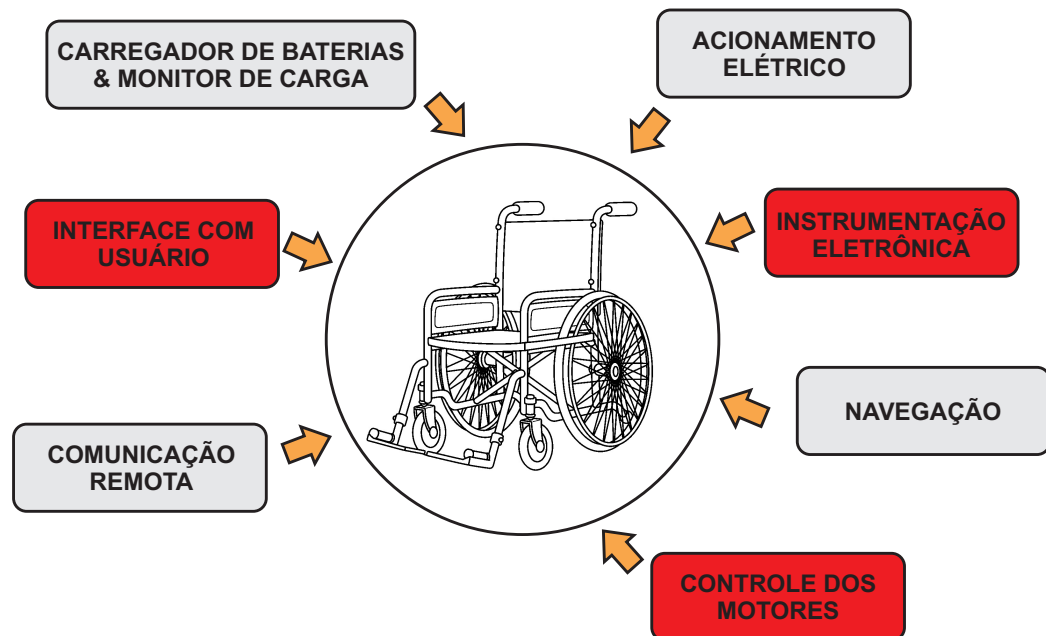
Para o funcionamento do controle, uma instrumentação eletrônica é necessária. Usada na realização da leitura de comandos do usuário para determinar a direção da cadeira, e fazer a leitura das velocidades dos motores para a realimentação do sistema de controle.

Desta forma, para as pessoas com deficiência motora, a cadeira de rodas é um dos principais instrumentos para auxiliar na sua locomoção. Um projeto de controle eficiente é muito importante para garantir movimentos suaves e estáveis na cadeira de rodas, facilitando assim o deslocamento e a melhor inclusão social desses indivíduos.

Em 2000, a Universidade Estadual de Londrina (UEL) iniciou pesquisas sobre controle de cadeiras de rodas automatizadas. Um modelo matemático interessante de uma plataforma retangular com 4 rodas, sendo 2 rodas dianteiras livres e 2 traseiras conectadas a um motor de corrente contínua cada uma foi descrito em (MAZO et al., 1995). Esse modelo foi a referência e inspiração para o início da pesquisa sobre cadeira de rodas elétricas com acionamento por sopro e sucção na UEL (SOBRINHO et al., 2000, 2003).

Foi desenvolvido um tipo de interface de acionamento com sensor de fluxo de ar (SOBRINHO et al., 2003). Outro tipo de interface de acionamento utilizando sensores de pressão de ar foi elaborada em (GERMANOVIX; CERVANTES, 2013). Em (ROSSINI, 2013) e (SANTOS et al., 2013), foram associadas técnicas de controle robusto.

Nesse sentido, no Laboratório Controle Avançado, Robótica e Engenharia Biomédica do Departamento de Engenharia Elétrica da UEL, o projeto em andamento busca modernizar uma cadeira de rodas incluindo, nas mudanças de projeto, os requisitos da figura 1.1:



**Figura 1.1:** Projeto da Cadeira de Rodas na UEL.

Fonte: Próprio autor, adaptado de (BORGES, 2014).

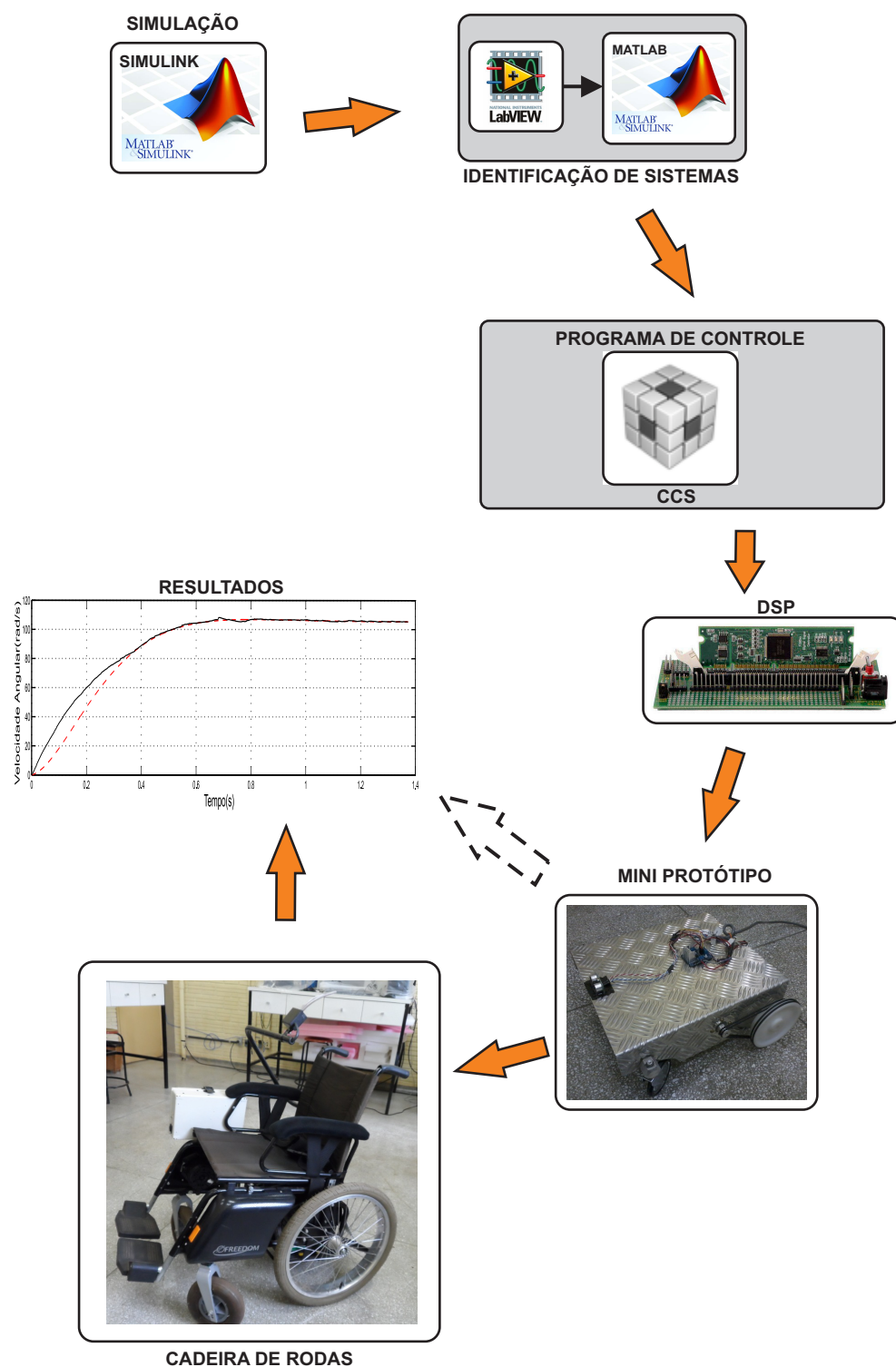
Nesta pesquisa, especificamente, no projeto da cadeira de rodas<sup>2</sup> desenvolvido no Laboratório de Controle Avançado, Robótica e Engenharia Biomédica da UEL, foi realizada a implementação do sistema de controle embarcado em DSP<sup>3</sup>, aplicando um controlador PID (Proporcional, Integral e Derivativo) nos motores. O controlador foi capaz de sincronizar a velocidade das duas rodas de tração, permitindo que a cadeira de rodas se locomovesse corretamente nas direções determinadas pelo usuário. No comando de direção da cadeira, foi utilizada uma instrumentação eletrônica contendo uma interface de sopro e sucção com sensor de fluxo de ar (SOBRINHO et al., 2000, 2003; FERREIRA, 2008); e outra interface com sensor de pressão de ar (GERMANOVIX; CERVANTES, 2013). As etapas do projeto como um todo trabalhadas nessa pesquisa podem ser visualizadas em destaque na figura 1.1.

Dentro dos itens destacados no projeto de modernização da cadeira de rodas, foram desdobradas as seguintes etapas: simulação, identificação dos motores, programação em prototipagem rápida, implementação em DSP em um mini protótipo e na cadeira de rodas, testes e resultados. Estas etapas podem ser visualizadas na figura 1.2.

Assim, essa pesquisa contribui com pessoas com paraplegia, fornecendo uma validação do sistema de controle embarcado dos motores, contribuindo para o desenvolvimento do projeto como um todo. Espera-se, com esta Dissertação,

<sup>2</sup>Cadeira de Rodas com PATENTE REQUERIDA (BR 10 2014 018375 2)

<sup>3</sup>DSP significa Processador Digital de Sinais, proveniente de *Digital Signal Processor*.



**Figura 1.2:** Diagrama do sistema.  
Fonte: Próprio autor.

além de cooperar com o projeto em andamento da cadeira de rodas, abrir novas linhas de pesquisa para trabalhos futuros.

## 1.1 Justificativa

Há uma necessidade cada vez maior de inclusão social e acessibilidade de pessoas com deficiência física na sociedade e no mercado de trabalho. Neste sentido há uma grande motivação para o desenvolvimento de dispositivos de locomoção que facilitem a vida desses indivíduos, que em sua maioria estão em plena atividade intelectual. Em todo o mundo existem pesquisas sobre cadeiras de rodas automatizadas.

Com essa motivação e com a proposta inicial de mestrado em desenvolver controle de motores de corrente contínua aplicados em veículos autônomos e robôs móveis, foi desenvolvida nesse trabalho a implementação do controle PID embarcado para os motores de uma cadeira de rodas com uma interface de acionamento por sopro e sucção.

A pesquisa teve como ponto inicial uma cadeira de rodas elétrica equipada com acionamento por *joystick*. O sistema de acionamento foi substituído por uma interface de sopro e sucção, o que permitiu a utilização por pessoas com limitação de acionamento manual. A implementação do controle dos motores foi primeiramente realizada no mini protótipo de fabricação própria para facilitar a validação inicial do sistema. Na sequência, o sistema foi implementado na cadeira de rodas, o que tornou possível a validação do sistema com a utilização de indivíduos com paraplegia.

## 1.2 Revisão Bibliográfica

O anseio de aumentar a acessibilidade para pessoas portadoras de deficiência motora tem sido interesse de pesquisadores. Nesta seção, serão discutidos artigos publicados em periódicos internacionais referentes a este tema.

A tetraplegia e paraplegia, consistem na paralisia total ou parcial dos membros superiores e inferiores do indivíduo (MCDONALD; SADOWSKY, 2002). Os indivíduos com essas limitações motoras apenas possuem o movimento dos membros superiores ou para os casos mais graves somente movimento da cabeça. Então para proporcionar mobilidade a esses indivíduos, uma das alternativas é uma cadeira de rodas. Cadeiras de rodas mecânicas causam desgastes físicos

nos indivíduos, além não atender todos os tipos de deficiências motoras (CHOW et al., 2009). Como alternativa existem as cadeiras de rodas automatizadas que possuem propulsão elétrica e diversos tipos de interface de acionamento.

Uma das técnicas de acionamento mais recentes encontradas é o acionamento através de movimentos da língua (NAM; ZHAO; CICHOCKI, 2012). Consiste em utilizar um dispositivo colocado na boca composto por sensores capazes de detectar o toque da língua e enviar um sinal de controle para os motores da cadeira permitindo a sua movimentação. Nesse assunto também é importante citar o acionamento por ondas cerebrais (CHO; VINOD; CHENG, 2009), em que o acionamento da cadeira ocorre pelo pensamento do usuário. Outro trabalho relevante e correlacionado com este tema é desenvolvido por (YAMAMOTO; IKEDA; SASAKI, 2008), onde é descrito o estudo de uma interface de acionamento de cadeira de rodas pela pressão da respiração.

Outro assunto importante relacionado a essa pesquisa é o controlador da cadeira de rodas. O controlador utilizado no desenvolvimento desse trabalho foi o PID. Esse controlador é de fácil implementação e bastante aplicado em cadeiras de rodas. Alguns trabalhos que utilizam controle PID de motores de cadeira de rodas podem ser citados (CHEN; CHEN; CHEN, 2000; SHIM et al., 2001; CHU et al., 2004; DING; COOPER, 2005). Outro trabalho importante com PID é descrito em (WANG et al., 2010).

Em contrapartida à simplicidade do PID, ele apresenta baixa robustez à distúrbios externos. Como alternativa para minimizar essa limitação, algumas propostas de modificação foram desenvolvidas. Uma das alternativas é a modificação do controlador PID, tornado-o menos suscetível a distúrbios, como utilizado em (WIDYOTRIATMO; RAUZANFIQR; SUPRIJANTO, 2012). Outra alternativa importante é a associação de uma lógica *Fuzzy* ao controlador PID, ampliando assim seu desempenho (TIAN; XU, 2009; LEISHMAN; HORN; BOURHIS, 2010).

## 1.3 Metodologia

Neste trabalho, utilizou-se o ambiente MATLAB/SIMULINK para a realização das simulações dos sistemas e estimação dos parâmetros do controlador. O ambiente LABVIEW foi utilizado para a aquisição de dados, em conjunto com uma placa de aquisição da *National Instruments* modelo PCI-6024E. O DSP escolhido para a realização deste trabalho foi o *Texas Instruments* TMS320F28335 (SILVA, 2012a, 2012b). O ambiente de programação utilizado foi o *Code Composer Studio*

(CCS).

Realizou-se primeiramente a simulação do sistema, compreendendo nesta etapa o modelo matemático e o controle PID proposto. A seguir as funções de transferências dos motores foram obtidas por meio da aquisição de dados realizada no LABVIEW (SILVA, 2012b) e a estimação de parâmetros por meio do MATLAB (BIAZETO, 2011). A etapa de implementação do sistema no DSP foi então concebida através do ambiente CCS em linguagem de programação C.

## 1.4 Objetivos

O objetivo dessa pesquisa foi realizar a implementação em DSP do controle PID embarcado para os motores de corrente contínua do mini protótipo e da cadeira de rodas, em conjunto com a instrumentação.

O trabalho foi realizado seguindo essas etapas:

1. Simulação do modelo matemático em conjunto com o controle proposto.
2. Aquisição de dados e identificação das funções de transferências dos motores.
3. Ajustes dos ganhos do controlador.
4. Prototipagem rápida em DSP.
5. Testes de funcionamento da cadeira de rodas.
6. Aquisição de dados para análise dos resultados.

## 1.5 Resultados Preliminares

Para se alcançar o resultado final do desenvolvimento do controle da cadeira de rodas que será apresentado nessa dissertação, iniciaram-se os ensaios experimentais em uma plataforma em pequena escala da cadeira de rodas, ilustrada na figura 1.3.

Nesta plataforma foi realizada a identificação das funções de transferência dos motores através da aquisição de tensão e velocidade angular dos motores utilizando o *software LabView* e estimação do modelo matemático utilizando o *software MatLab*, e a implementação do sistema no DSP. Assim, foram gerados



**Figura 1.3:** Mini protótipo da cadeira de rodas  
Fonte: Próprio autor.

alguns resultados preliminares que viabilizaram a implementação do sistema na cadeira de rodas.

No processo de identificação das funções de transferência dos motores do mini protótipo, um modelo matemático foi estimado. As funções de transferência determinadas nesse processo dos motores esquerdo e direito são, respectivamente:

$$H_l(s) = \frac{318,7}{s + 10,09} \quad (1.1)$$

e

$$H_r(s) = \frac{135,4}{s + 9,835}. \quad (1.2)$$

O mini protótipo possui sistema de transmissão por correia, no qual é possível obter a relação:

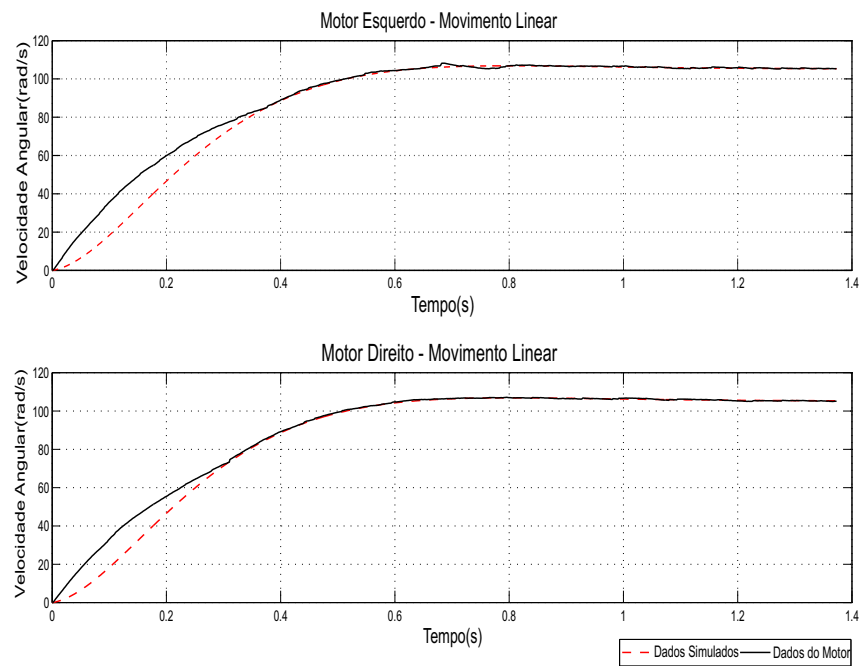
$$Rel_{Transf} = \frac{D_{rodas}}{D_{eixo}} \quad (1.3)$$

onde  $D_{roda}$  é o diâmetro das rodas,  $D_{eixo}$  é o diâmetro dos eixos dos motores e  $Rel_{Transf}$  é a relação de transformação de velocidade entre os eixos dos motores e as rodas. Neste caso,  $D_{roda} = 12cm$  e  $D_{eixo} = 2cm$

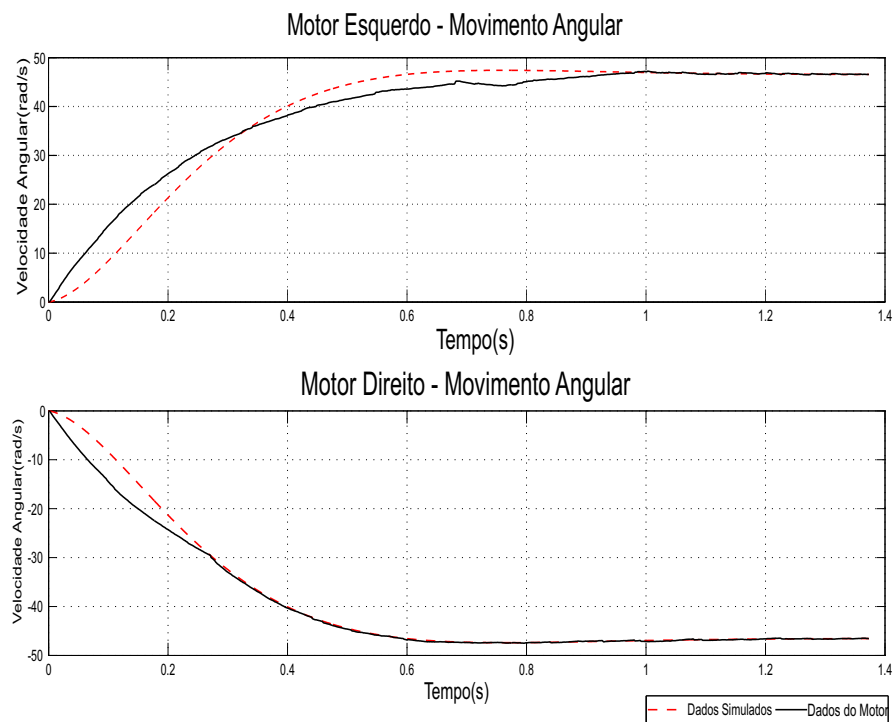
Após a implementação do controlador, os motores do mini protótipo apresentaram as respostas para movimento linear e angular mostradas respectivamente nas figuras 1.4 e 1.5. Consideram-se movimentos lineares os movimentos para frente e trás, e movimentos angulares, os movimentos para direita e esquerda.

## 1.6 Organização da Dissertação

Este trabalho está organizado na seguinte forma: no capítulo 2 é apresentada uma breve introdução teórica a respeito dos temas abordados, no capítulo 3 é apresentada a descrição do projeto e implementação do sistema, no capítulo 4 são apresentados os resultados obtidos e, no capítulo 5 as considerações finais.



**Figura 1.4:** Comparação da simulação e dados captados dos motores em movimento linear  
Fonte: Próprio autor.



**Figura 1.5:** Comparação da simulação e dados captados dos motores em movimento angular  
Fonte: Próprio autor.



## 2 Conceitos Teóricos

Neste capítulo será apresentada uma introdução teórica a respeito dos principais temas abordados na dissertação.

Para o desdobramento deste trabalho é necessário um modelo matemático linear que represente a cinemática do sistema, no caso, a cadeira de rodas. O sistema precisa ser controlado, a fim obter um sistema com comportamento estável. Nesse processo de controle da cadeira de rodas, é necessário determinar um modelo matemático que represente de forma mais aproximada possível o sistema real. Para isso, uma técnica de identificação de sistemas é usada. Desta forma é possível conhecer os parâmetros do sistema, ou seja, seu comportamento de acordo com determinado tipo de entrada no mesmo.

### 2.1 Modelo Matemático Linear

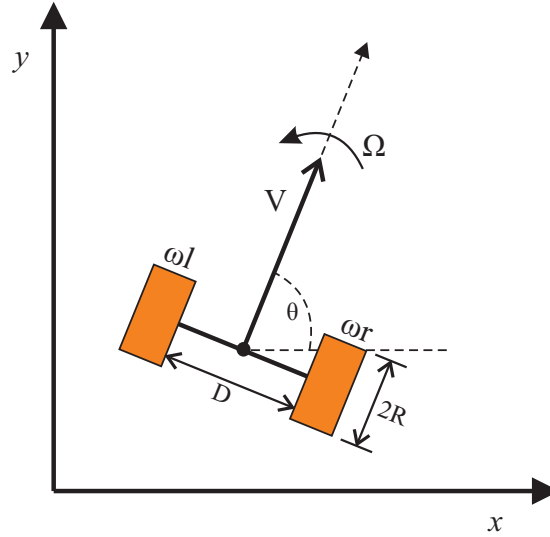
A cadeira consiste de uma plataforma baseada em 4 rodas: duas livres na frente e duas motorizadas independentemente na traseira, caracterizando um robô móvel com tração diferencial. Cada roda usa um motor de corrente contínua.

Um controle eficaz requer o modelo mecânico da cadeira. Para este caso, um modelo cinemático garante bons resultados. Para facilitar a implementação do sistema de controle foi usado um modelo cinemático para robôs móveis com tração diferencial proposto por (MAZO et al., 1995) e adaptado por (SOBRINHO et al., 2000, 2003). Esse modelo simplificado foi adaptado à cadeira de rodas. O modelo é ilustrado na figura 2.1, permite o cálculo das velocidades linear ( $V$ ) e angular ( $\Omega$ ) da cadeira, a partir das velocidades angulares das rodas motorizadas  $\omega_l$  e  $\omega_r$ .

Se  $R$  é o raio das rodas motorizadas,  $D$  a distância entre elas e “ $u$ ” é a posição instantânea da cadeira (SOBRINHO et al., 2003), dada a relação  $\frac{du}{dt} = V$ , então:

$$\theta(t) = \theta_0 + \frac{R}{D} \int_0^t (\omega_l - \omega_r) dt \quad (2.1)$$

$$V = \frac{du}{dt} = \frac{R}{2}(\omega_l + \omega_r) \quad (2.2)$$



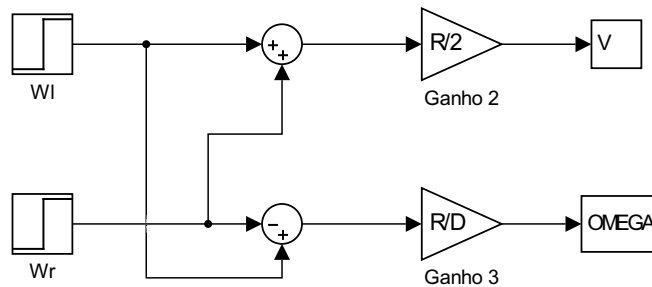
**Figura 2.1:** Modelo linear da cadeira  
Fonte: Próprio autor, adaptado de (MAZO et al., 1995).

As velocidades linear  $V$  e angular  $\Omega$  da cadeira são dadas por:

$$V = \frac{R}{2}(\omega_l + \omega_r) \quad (2.3)$$

$$\Omega = \frac{R}{D}(\omega_l - \omega_r) \quad (2.4)$$

Essas expressões são mostradas graficamente na figura 2.2:



**Figura 2.2:**  $V$  e  $\Omega$  em função de  $\omega_l$  e  $\omega_r$   
Fonte: Próprio autor, adaptado de (MAZO et al., 1995).

## 2.2 Controle de Sistemas

Sistemas de controle automático fazem parte da sociedade moderna e tem desempenhado um papel fundamental no avanço da engenharia e da ciência. Além da extrema importância em sistemas de veículos espaciais, sistemas de direcionamento de mísseis, sistemas robóticos e similares, o controle automático tem se tornado de grande importância e parte integrante dos modernos processos industriais e de produção, além de sistemas de equipamentos de uso doméstico. Por exemplo, o controle automático é essencial no controle numérico na indústria aeroespacial, no projeto de carros e caminhões na indústria automotiva e de dispositivos internos de equipamentos domésticos como computadores, leitores de DVD, etc. É essencial também em operações industriais, como o controle de pressão, de temperatura, de umidade, de viscosidade e de vazão nos processos industriais. Existem também os processos naturais: no próprio corpo humano existem vários sistemas de controle, como o pâncreas que regula o açúcar no sangue e a adrenalina que aumenta os batimentos cardíacos, aumentando o oxigênio nas células (OGATA, 2010; NISE, 2010).

O primeiro trabalho significativo de controle automático foi construído por James Watt para o controle de velocidade de uma máquina a vapor, no século XVIII. Outros trabalhos importantes nos primeiros estágios do desenvolvimento da teoria de controle se devem a Minorsky com controle de pilotagem para embarcações, Hazen com sistemas de controle de posição de servomecanismos e Nyquist com determinação de estabilidade de sistemas, entre muitos outros (OGATA, 2010; NISE, 2010).

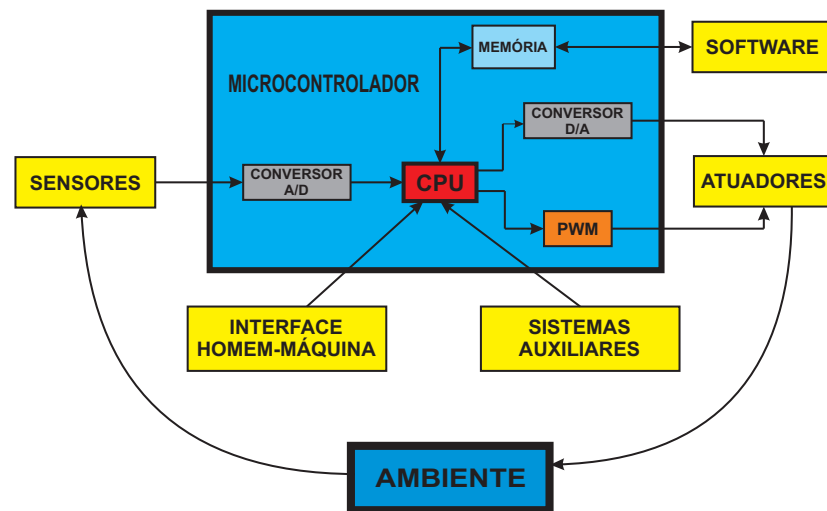
Um item de grande importância de um sistema de controle é saída do sistema. Com base no sinal de entrada e na referência (sinal desejado de saída) podemos controlar as variáveis do sistema para obter a resposta desejada. Para um controlador adequado a cada aplicação é necessário realizar estratégias de projeto de controle para aprimorar o sistema e obter a melhor resposta possível.

### 2.2.1 Sistemas de Controle Embarcado

Um sistema embarcado é um sistema microprocessado no qual o computador é completamente encapsulado ou dedicado ao dispositivo ou sistema que ele controla. É um sistema completo e independente, mas preparado para realizar apenas uma determinada tarefa. O usuário final não tem acesso ao programa que foi embutido no dispositivo, mas em alguns casos poderá interagir com o equipa-

mento através de interfaces homem-máquina como teclados, *displays*, entre outros (MELO, 2007; SILVA, 2012a).

Como principais características de um sistema embarcado, tem-se a sua capacidade computacional e a sua independência de operação. Outros aspectos relevantes são referentes aos tipos de sistemas, modos de funcionamento e itens desejados em aplicações embarcadas (MELO, 2007; SILVA, 2012a).



**Figura 2.3:** Diagrama de blocos básico de um sistema de controle embarcado.  
Fonte: Próprio autor.

A figura 2.3 ilustra o diagrama de blocos básico de um sistema embarcado. A CPU controla e gerencia todas as tarefas do sistema através do software armazenado na memória, o conversor A/D é responsável pela conversão do sinal analógico de entrada em um sinal digital, o conversor D/A e o módulo PWM são responsáveis por gerar os sinais de saída que vão para os atuadores a fim de realizar o trabalho no ambiente, os sensores fazem a conversão das grandezas do ambiente em sinais elétricos, a interface homem-máquina é o parte do sistema em que o usuário pode fazer interações, que podem ser para ajustar o modo de operação do sistema ou fornecer alguma informação e, os sistemas auxiliares são sistemas dedicados à aplicações específicas que podem fazer interação com o sistema.

Sistemas embarcados podem funcionar de duas maneiras e são determinantes para saber como programar o dispositivo e como será seu funcionamento e comportamento na aplicação para o qual foi desenhado.

- *Reativo:* o funcionamento se dá como resposta a eventos externos, que podem ser periódicos ou assíncronos. Há, então, uma necessidade de entrada de dados para que aconteçam as ações de funcionamento. Geralmente não

há limite de tempo para que os sinais de entrada sejam acionados, pois dependem da interação com o usuário ou com o processo ao qual é destinado. Porém, a saída, função do sinal de entrada, deve ser gerada exatamente após os sinais de entrada começarem a atuar.

- *Controle em tempo real:* existem limites de tempo para executar cada tarefa. Por isso mesmo, nem sempre tempo real é igual ao modo mais rápido de executar uma tarefa. Estes modos de operação, por serem cíclicos, não dependem da entrada e sinais para executar as atividades, sendo capaz de tomar decisões referentes à ausência dos mesmos. Os sistemas de tempo real são classificados em:

- **Soft Real Time:** As tarefas podem ser executadas em um intervalo de tempo específico, sem consequências graves se este limite de tempo não for cumprido. Um exemplo é um sistema bancário, onde apenas uma mensagem de erro aparecerá se determinada tarefa não for realizada dentro do tempo pré-determinado. A maior preocupação durante a programação dos dispositivos é relativa a repetição de procedimentos que podem ter seu prazo estourado e sobre as mensagens de alerta ao usuário.
- **Hard Real Time:** As tarefas devem ser executadas em um tempo específico, com consequências graves se qualquer tarefa falhar. Como exemplo pode-se pensar nos sistemas de controle de um avião, em que uma falha pode resultar em queda e perdas de vidas. A maior preocupação durante a programação dos dispositivos é relacionada a ter sempre sistemas redundantes que impeçam ou alertem imediatamente caso algum processo falhe ou não seja executado dentro do tempo pré-determinado.

A aplicação prática de controle de sistemas, na maioria das vezes, necessita de um sistema de controle embarcado para funcionar. A implementação de um sistema de controle embarcado consiste em programar o projeto de controle teórico em algum sistema eletrônico, móvel ou não, para a realização da tarefa.

### 2.2.2 Controladores PID

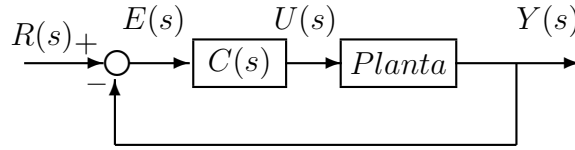
O Controlador Proporcional-Integral-Derivativo (PID) é uma estrutura de controle largamente utilizada em sistemas práticos. Embora este controlador já fosse conhecido desde o período entre as duas guerras mundiais, continua, nos dias

de hoje, a ser o controlador industrial mais utilizado. Este controlador possui como principais características a simplicidade, fácil aplicabilidade no controle de muitas aplicações (GOODWIN; GRAEBE; SALGADO, 2001), é facilmente ajustado em campo, porém não pode proporcionar um controle ótimo. Apesar de seu grande uso comercial, essa estrutura é amplamente estudada desde a sua criação, tendo diversos métodos e ajustes pesquisados. O controlador PID pode ser ajustado para obter um desempenho satisfatório, com base em um nível modesto de informação (AGUIRRE, 2007a). Existem vários métodos de ajuste automático, entretanto o ajuste manual ainda é utilizado.

Um controlador PID é definido por:

$$C_{PID} = \left( K_p + \frac{K_i}{s} + K_D s \right) \quad (2.5)$$

A estrutura PID é composta por três coeficientes: *Proporcional*, *Integral*, *Derivativo*, que podem ser ajustados para obter a resposta ideal. Considere o sistema em malha fechada da figura 2.4, sendo  $C(s) = k_p + k_i/s + k_d s$  a função de transferência do controlador:



**Figura 2.4:** Sistema em malha fechada.  
Fonte: (OGATA, 2010)

As expressões para os controladores Proporcional (P), Proporcional-Integral (PI), Proporcional-Derivativo (PD) e PID são descritas pela relação da saída do controlador  $U(s)$  e o sinal de erro  $E(s) = R(s) - Y(s)$ . As funções de transferências destes controladores são dadas por (OGATA, 2010):

Controle proporcional (P):

$$C(s) = \frac{U(s)}{E(s)} = K_p, \quad (2.6)$$

Controle proporcional-integral (PI):

$$C(s) = \frac{U(s)}{E(s)} = K_p \left( 1 + \frac{1}{T_i s} \right), \quad (2.7)$$

Controle proporcional-derivativo (PD):

$$C(s) = \frac{U(s)}{E(s)} = K_p (1 + T_d s), \quad (2.8)$$

Controle proporcional-integral-derivativo (PID):

$$C(s) = \frac{U(s)}{E(s)} = K_p \left( 1 + \frac{1}{T_i s} + T_d s \right), \quad (2.9)$$

onde  $T_i$  é chamado de constante de tempo integrativo,  $T_d$  é chamado de constante de tempo derivativo e  $K_p$  é denominado ganho proporcional. Os efeitos das ações de controle podem ser combinadas. Essa combinação tem a vantagem de usar as características individuais de cada uma. Analisando esses efeitos individuais tem-se:

- **Ação proporcional:** Essa ação é baseada no sinal de erro do sistema,  $E(s) = R(s) - Y(s)$ . Então um sinal de controle maior deve ser aplicado ao processo quando o sinal de saída atual se encontra distante do valor de referência, ou, aplicar um sinal de controle menor quando o sinal de saída atual se encontra próximo do valor de referência. Com o aumento do ganho proporcional tem-se um aumento da velocidade de resposta do sistema de controle (AGUIRRE, 2007a).
- **Ação Integral:** A ação integral fornece uma saída no controle proporcional ao erro acumulado (KIM; SCHAEFER, 2005). Desta forma, essa ação irá acumular o erro ao longo do tempo. A sua principal característica prática é reduzir o erro estacionário.
- **Ação Derivativa:** Essa ação fornece a aplicação de um sinal de controle proporcional à derivada do sinal de erro (AGUIRRE, 2007a). Aplicar como controle um sinal proporcional à derivada do sinal de erro é equivalente a aplicar uma ação baseada na tendência de evolução do erro. Esta ação é capaz de antecipar uma resposta e tende a fazer com que o sistema reaja mais rapidamente.

### 2.2.3 Estabilidade de Polinômios com Dependência Polinomial através do Critério de Routh

Em Teixeira, Assunção e Covacic (2007), foi descrito um programa desenvolvido em Matlab para determinar a faixa de valores de  $k$  que torna estável um sistema com um controlador proporcional  $k$  e realimentação negativa. Neste caso, o polinômio característico  $d(s, k)$  é o denominador da função de transferência de malha fechada. Este programa também determina a região de estabilidade de um sistema com um controlador PID, proporcional-integral (PI), proporcional-derivativo (PD), entre outros.

A estabilidade é um item muito importante em um projeto de sistema de controle. Considere um sistema linear invariante no tempo, cujo polinômio característico  $d(s, k)$  é dado por:

$$d(s, k) = d_n(k)s^n + \dots + d_1(k)s + d_0(k), \quad (2.10)$$

sendo  $d_n(k)$ ,  $d_{(n-1)}(k)$ ,  $\dots$ ,  $d_1(k)$  e  $d_0(k)$  polinômios em  $k$ , como descrito abaixo:

$$d_i(k) = d_{ib_i}k^{b_i} + \dots + d_{i1}k + d_{i0}, \quad (2.11)$$

para  $i = 0, \dots, n$ , sendo  $b_i$  o grau do coeficiente  $d_i(k)$ .

O sistema é estável se e somente se todas as raízes do polinômio característico  $d(s, k)$ , apresentado em (2.10), tiverem parte real negativa. O Critério de Estabilidade de Routh-Hurwitz é uma ferramenta útil para determinar se o sistema é estável (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007). Considere a tabela 2.1.

**Tabela 2.1:** Tabela de Routh.

Fonte: (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

|               |           |                                                      |                                                      |                                                      |
|---------------|-----------|------------------------------------------------------|------------------------------------------------------|------------------------------------------------------|
| linha $n$     | $s^n$     | $a_{n1}(k) = \frac{p_{n1}(k)}{q_n(k)}$               | $a_{n2}(k) = \frac{p_{n2}(k)}{q_n(k)}$               | $a_{n3}(k) = \frac{p_{n3}(k)}{q_n(k)}$               |
| linha $(n-1)$ | $s^{n-1}$ | $a_{(n-1)1}(k) = \frac{p_{(n-1)1}(k)}{q_{(n-1)}(k)}$ | $a_{(n-1)2}(k) = \frac{p_{(n-1)2}(k)}{q_{(n-1)}(k)}$ | $a_{(n-1)3}(k) = \frac{p_{(n-1)3}(k)}{q_{(n-1)}(k)}$ |
| $\vdots$      | $\vdots$  | $\vdots$                                             | $\vdots$                                             | $\vdots$                                             |
| linha 2       | $s^2$     | $a_{21}(k) = \frac{p_{21}(k)}{q_2(k)}$               | $a_{22}(k) = \frac{p_{22}(k)}{q_2(k)}$               |                                                      |
| linha 1       | $s^1$     | $a_{11}(k) = \frac{p_{11}(k)}{q_1(k)}$               |                                                      |                                                      |
| linha 0       | $s^0$     | $a_{01}(k) = \frac{p_{01}(k)}{q_0(k)}$               |                                                      |                                                      |

Os termos da linha  $n$  são dados por  $a_{n1}(k)$ ,  $a_{n2}(k)$ ,  $\dots$ , sendo:

$$p_{n1}(k) = d_n(k), \quad p_{n2}(k) = d_{(n-2)}(k), \quad \dots$$

$$q_n(k) = 1. \quad (2.12)$$

Os termos da linha  $(n-1)$  são dados por  $a_{(n-1)1}(k)$ ,  $a_{(n-1)2}(k)$ ,  $\dots$ , sendo:

$$p_{(n-1)1}(k) = d_{(n-1)}(k), \quad p_{(n-1)2}(k) = d_{(n-3)}(k), \quad \dots$$

$$q_{(n-1)}(k) = 1. \quad (2.13)$$

Os termos da linha  $j$ ,  $j = n-2, \dots, 1, 0$ , são dados por  $a_{ji}(k) = p_{ji}(k)/q_j(k)$ , sendo:

$$p_{ji}(k) = p_{(j+1)1}(k)p_{(j+2)(i+1)}(k) - p_{(j+2)1}(k)p_{(j+1)(i+1)}(k),$$

$$q_j(k) = q_{(j+2)}(k)p_{(j+1)1}(k), \quad (2.14)$$



$$j = n - 2, n - 1, \dots, 1, 0, \quad i = 1, 2, \dots$$

Os elementos não-nulos  $p_{ji}(k)$  e  $q_j(k)$ ,  $j = n, n - 1, \dots, 1, 0$ ,  $i = 1, 2, \dots$ , são polinômios em  $k$ . De acordo com o Critério de Estabilidade de Routh-Hurwitz, o polinômio  $d(s, k)$  é Hurwitz para um dado valor de  $k = k_o$ , isto é, todas as raízes de  $d(s, k_o)$  têm parte real negativa, se e somente se todos os elementos da primeira coluna da tabela de Routh apresentam o mesmo sinal (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

### 2.2.4 Estabilidade de Sistemas com Realimentação Estática da Saída

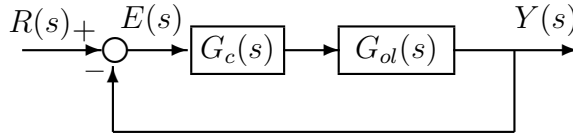
Considere uma planta controlável e observável,  $G_{ol}(s) = n(s)/d(s)$ , com:

$$n(s) = n_n s^n + n_{(n-1)} s^{n-1} + \dots + n_1 s + n_0, \quad (2.15)$$

$$d(s) = d_n s^n + d_{(n-1)} s^{n-1} + \dots + d_1 s + d_0 \quad (2.16)$$

e  $d_n \neq 0$ .

É proposto o seguinte problema: determine a faixa de valores de  $k$ , se existirem, tais que o sistema realimentado da figura 2.5, com  $G_c(s) = k$ , é estável, utilizando o Critério de Estabilidade de Routh.



**Figura 2.5:** Sistema realimentado com controlador  $G_c(s)$ .

Fonte: (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007)

A função de transferência do sistema de malha fechada  $G_d(s)$  é dada por:

$$G_d(s) = \frac{Y(s)}{R(s)} = \frac{kG_{ol}(s)}{1 + kG_{ol}(s)} = \frac{kn(s)}{d(s) + kn(s)}. \quad (2.17)$$

Os polos do sistema são as raízes do polinômio característico  $d(s, k) = d(s) + kn(s)$ . Substituindo (2.15) e (2.16) em (2.17), obtém-se o polinômio característico do sistema de malha fechada. Assim, por meio do Critério de Estabilidade de Routh, podem-se obter os valores de  $k$  tais que todas as raízes do polinômio característico  $d(s, k)$  tenham parte real negativa, caso existam (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

### 2.2.5 Estabilidade de Sistemas com Controladores PID e PI

A ação de controle de um controlador Proporcional-Integral-Derivativo (PID) é baseada na combinação dos três controladores: controle proporcional, controle integral e controle derivativo, mantendo sempre as vantagens individuais de cada uma das três ações de controle, como descrito na seção 2.2.2. Conforme (OGATA, 2010), o controle PID é definido por:

$$u(t) = K_p \cdot e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}. \quad (2.18)$$

A função de transferência de um controlador PID é dada por:

$$G_c(s) = K_p + \frac{K_i}{s} + K_d s.$$

Na prática, a função de transferência do termo derivativo  $K_d s$  é:

$$G_d(s) = \frac{K_d s}{\tau_d s + 1},$$

mas, usualmente,  $\tau_d$  é muito menor que as constantes de tempo do processo e, conseqüentemente, pode ser ignorado.

Um controlador PID possui um polo em  $s = 0$  e dois zeros, cujas posições dependem de  $K_p$ ,  $K_i$  e  $K_d$ , que são ajustáveis. Para a implementação do controlador PID, é necessário determinar, para um dado processo, os ganhos proporcional, integral e derivativo.

Considere o sistema realimentado na figura 2.5, sendo  $G_c(s)$  um controlador PID. A função de transferência do sistema de malha fechada  $G_{cl}(s) = Y(s)/R(s)$  é dada por:

$$G_{cl}(s) = \frac{(K_p s + K_i + K_d s^2)n(s)}{sd(s) + (K_p s + K_i + K_d s^2)n(s)}.$$

Assim os polos do sistema são as raízes do polinômio característico  $r(s, K_p, K_i, K_d) = sd(s) + (K_p s + K_i + K_d s^2)n(s)$ .

Um caso particular de controlador PID é o controlador PI, que é obtido a partir do controlador PID, com  $K_d = 0$ . O controlador PI é descrito por:

$$G_c(s) = K_p + \frac{K_i}{s}$$

e a função de transferência do sistema de malha fechada  $G_{cl}(s)$  é dada por:

$$G_{cl}(s) = \frac{(K_p s + K_i)n(s)}{sd(s) + (K_p s + K_i)n(s)}.$$

Assim os polos do sistema são as raízes do polinômio característico  $r(s, K_p, K_i) = sd(s) + (K_p s + K_i)n(s)$ .

Para  $K_I = 0$ , tem-se um controlador PD, descrito por:

$$G_c(s) = K_P + K_D s$$

e a função de transferência do sistema de malha fechada  $G_{cl}(s)$  é dada por:

$$G_{cl}(s) = \frac{(K_P + K_D s)n(s)}{d(s) + (K_P + K_D s)n(s)}.$$

Os polos do sistema são as raízes do polinômio característico  $r(s, K_P, K_D) = d(s) + (K_P + K_D s)n(s)$ . Para o controlador PID, fixando-se os valores de dois parâmetros (por exemplo,  $K_i$  e  $K_d$ ), é possível determinar a faixa de estabilidade do terceiro parâmetro (no caso,  $K_p$ ), utilizando-se o Critério de Estabilidade de Routh-Hurwitz. Atribuindo-se, então, um conjunto de valores, definido por um valor inicial, um valor final e um valor de incremento, para cada um dos dois parâmetros fixos, pode-se obter um esboço da região de estabilidade do sistema da figura 2.5 realimentado com um controlador PID.

Para o controlador PI, a análise da estabilidade é obtida fixando-se um dos dois parâmetros e determinando-se a faixa de estabilidade do segundo parâmetro, através do Critério de Estabilidade de Routh-Hurwitz (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

## 2.3 Identificação de Sistemas

Modelagem matemática é a área do conhecimento que estuda maneiras de desenvolver e implementar modelos matemáticos de sistemas reais. Há várias formas e técnicas de para obtenção de modelos matemáticos, sendo a *modelagem caixa branca* uma das mais usadas. Nesse caso, faz-se necessário conhecer a fundo o sistema a ser modelado. Além de estar bem familiarizado com o sistema, para esse tipo de modelagem é necessário conhecer as relações matemáticas que descrevem os fenômenos envolvidos. Infelizmente, devido ao conhecimento e tempo necessário para modelar um sistema partindo do equacionamento dos fenômenos envolvidos, nem sempre é viável seguir esse procedimento de modelagem (AGUIRRE,

2004).

A identificação de sistemas é uma área do conhecimento que estuda técnicas alternativas de modelagem matemática. Uma das características dessas técnicas é que pouco ou nenhum conhecimento prévio do sistema é necessário, e consequentemente, tais métodos são também referidos como *modelagem (ou identificação) caixa preta* (AGUIRRE, 2004).

Para o projeto do controlador da cadeira de rodas, é necessário o conhecimento do comportamento dos motores elétricos. Como suas características físicas são, na maioria, desconhecidas, para esse caso a modelagem caixa branca torna-se muito complexa. Então, a modelagem caixa preta ou identificação de sistemas foi escolhida para obter a representação matemática dos motores.

### 2.3.1 Conceitos de Identificação de Sistemas

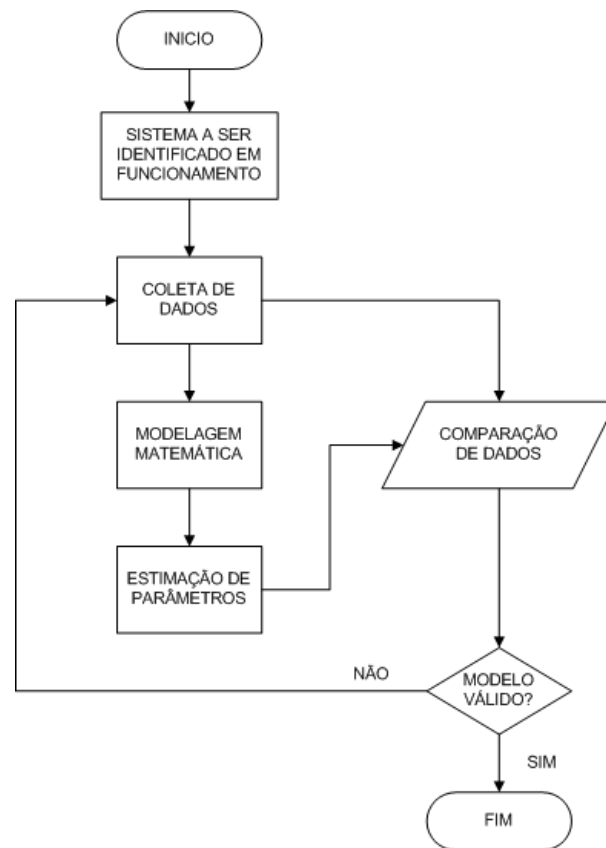
Para a realização de uma identificação modelo caixa preta, deve-se seguir algumas regras como mostrado no fluxograma da figura 2.6.

O primeiro passo é aplicar algum tipo de sinal na entrada do sistema e, após isso, observar sua saída por um determinado tempo, armazenando as informações de entrada, saída e tempo. As informações são coletadas em forma de amostras durante o intervalo de tempo de coleta. E essas amostras são utilizadas para determinar a estrutura do modelo a ser estimado. Depois de armazenados os dados é adotada alguma técnica de estimação do modelo, levando-se em consideração as características do sistema original para determinar qual o modelo mais adequado, e de maior qualidade para o sistema, então é feita a estimação dos parâmetros (LJUNG, 1987; AGUIRRE, 2004; SILVA, 2012b; SOUZA, 2014).

Para validar o modelo é preciso saber se ele representa matematicamente o sistema real de forma verdadeira. Para isso, é preciso fazer uma comparação entre os dados coletados e os parâmetros estimados, ou seja, essa comparação vai demonstrar quão próximo do sistema real o modelo estimado está. Sendo uma aproximação aceitável, o modelo é considerado válido, caso não seja uma nova coleta de dados e estimação dos parâmetros precisam ser feitas.

### 2.3.2 Modelos

A grande inspiração para os estudos dos cientistas é a natureza. Observando a natureza, desde muito tempo atrás, o homem busca entender e reproduzir seus fenômenos. Com base nisso, começou-se a desenvolver modelos de sistemas, com



**Figura 2.6:** Fluxograma de Identificação de Sistemas.  
 Fonte: Próprio autor, adaptado de (LJUNG, 1987; SOUZA, 2014).

os quais tentavam representar um fenômeno na natureza através de regras ou conjunto de regras e procedimentos. A modelagem de sistemas permitiu o estudo de muitos sistemas complexos como, por exemplo, o funcionamento do sistema do corpo humano (AGUIRRE, 2004) e o comportamento dos átomos (RODRIGUES, 1996).

Segundo Aguirre (2007b) modelos podem ser regras ou um conjunto delas, que descrevem o comportamento do sistema fornecendo informações temporais das variáveis envolvidas. Existem variados tipos de modelos: gráficos, matemáticos, computacionais, mentais, etc.

Normalmente utilizamos os modelos mentais no dia a dia; esses modelos indicam uma sequência de ações para atingir um objetivo. A habilidade de dirigir um carro para deslocamento é um exemplo de modelo mental. O modelo gráfico é descrito por tabelas que relacionam as diferentes variáveis do sistema. Como exemplo de modelo gráfico tem-se a curva característica tensão-corrente de um dispositivo eletrônico (RODRIGUES, 1996).

Modelos matemáticos são estruturas utilizadas para representar o comportamento de sistemas de forma matemática, ou seja, uma ou um conjunto de relações matemáticas representam o comportamento do sistema. Modelos ma-

temáticos são constituídos de equações diferenciais (tempo contínuo) ou equações de diferenças (tempo discreto). Para Aguirre (2004) os modelos matemáticos podem ser classificados como:

- **Modelos estáticos e dinâmicos:** Modelos estáticos são aqueles que a saída só depende do valor da entrada. Nos modelos dinâmicos a saída do sistema é dependente do valor de entrada e do valor da saída de um instante de tempo anterior, ou seja, possui uma realimentação, sendo representado pelo sistema em malha fechada. Modelos estáticos são descritos por equações algébricas e os modelos dinâmicos são descritos por equações diferenciais.
- **Modelos discretos e contínuos:** São modelos contínuos quando a saída possui valores conhecidos por todo o tempo, sem interrupção e são descritos por equações diferenciais. Quando se tem a saída com valores apenas em instantes de tempo, tem-se um modelo discreto no tempo, os valores são amostrados em pequenos intervalos de tempo. Esse modelo é descrito por equação de diferenças.
- **Modelos monovariáveis e multivariáveis:** Basicamente, modelos multivariáveis são os que têm múltiplas entradas ou múltiplas saídas (MIMO, MISO e SIMO). Os modelos monovariáveis (SISO) têm apenas um par de variáveis, ou seja, uma entrada e uma saída.
- **Modelos determinísticos e estocásticos:** Modelos determinísticos são modelos que descartam a presença de ruídos ou incertezas presentes no sistema, mesmo que sejam permitidos. Já os modelos estocásticos, além de levar em conta a presença de ruídos e incertezas em seus dados, também utilizam recursos para eliminá-los ou reduzi-los.

### 2.3.3 Modelos Paramétricos

Para a identificação de sistemas existem dois métodos básicos: métodos paramétricos e não paramétricos. Nessa parte será estudado o método paramétrico por ser o método de maior relevância para esse trabalho.

Segundo Coelho e Coelho (2004), a identificação paramétrica é a observação de variáveis do sistema com algum critério predefinido, gerando assim modelos matemáticos.

Métodos paramétricos utilizam relações matemáticas parametrizadas para descrever o comportamento dinâmico do sistema. Eles são caracterizados por números ou coeficientes e ajustados por técnicas de estimação a partir de dados coletados do sistema.

Métodos não paramétricos são caracterizados por gráficos, como por exemplo: resposta ao impulso e resposta em frequência (AGUIRRE, 2007b).

Os métodos paramétricos são utilizados quando há alguma informação, mesmo que mínima, do sistema, como por exemplo algumas relações que permitam a escolha de famílias de modelos mais adequada para o sistema (AGUIRRE, 2007b). Os modelos paramétricos são usados para estimar vetores de parâmetros, aqui designados como  $\alpha$ , para um certo modelo. Como exemplo tem-se a seguinte equação:

$$y(t) + ay(t-1) = bu(t-1) + e(t). \quad (2.19)$$

Com base na equação (2.19), tem-se uma equação clássica de um modelo discreto de primeira ordem, então seu vetor pode ser definido como (SODERSTROM; STOICA, 1989) :

$$\alpha = \frac{a}{b} \quad (2.20)$$

Com base em (AGUIRRE, 2004), tem-se diversos tipos de modelos que podem ser utilizados como por exemplo: Box-Jenkins (BJ), output error (OE), auto regressivo com entrada externa (ARX), auto regressivo com média móvel e entradas exógenas (ARMAX).

### 2.3.4 Modelo ARX

A auto-regressão é uma representação matemática do comportamento de um processo através de um modelo que pode ser utilizado para determinar o seu comportamento futuro da saída  $y(k)$  em relação à entrada  $u(k)$  (SOUZA, 2014). Para obtenção do modelo, é necessária a aplicação de um estímulo no sistema e a observação das variáveis de saída.

Com base nisso, o conjunto de variáveis observadas pode ser representado por um sinal  $y(t) = [y_t, y_{t-1}, \dots, y_{t-k}]$ , onde  $t$  é o instante referente a um dado valor de  $y$  e  $k$  é o valor do atraso considerado para o sinal (SOUZA, 2014).

O modelo geral segundo (AGUIRRE, 2004) é :

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}e(k), \quad (2.21)$$

$$y(k) = \frac{B(q)}{F(q)A(q)}u(k) + \frac{C(q)}{D(q)A(q)}e(k),$$

$$y(k) = H(q)u(k) + G(q)e(k),$$

em que  $q^{-1}$  representa o operador de atraso, isto é,  $y(k)q^{-1} = y(k-1)$ , e  $e(k)$  um ruído branco com  $A, B, C, D, F$  polinômios definidos a seguir:

$$A(q) = 1 + a_1q^{-1} + \dots + a_{n_a}q^{-n_a}, \quad (2.22)$$

$$B(q) = b_1q^{-1} + \dots + b_{n_b}q^{-n_b}, \quad (2.23)$$

$$C(q) = 1 + c_1q^{-1} + \dots + c_{n_c}q^{-n_c}, \quad (2.24)$$

$$D(q) = 1 + d_1q^{-1} + \dots + d_{n_d}q^{-n_d}, \quad (2.25)$$

$$F(q) = 1 + f_1q^{-1} + \dots + f_{n_f}q^{-n_f}. \quad (2.26)$$

As funções  $H(q)$  e  $G(q)$  são referidas como funções de transferência do processo e ruído, respectivamente.

O modelo AR (auto-regressivo) é considerado o mais comum regressor paramétrico e é dado por (AGUIRRE, 2004):

$$A(q)y(t) = e(t). \quad (2.27)$$

O operador  $q^{-n_a}$  indica o valor da função  $y(t)$  em um instante anterior  $y(t - n_a)$ . Desta forma, selecionando a ordem  $n_a$  do operador, determina-se o número de valores atrasados de  $y(t)$  que serão utilizados para determinar o valor no instante atual (CARVALHO, 2008). O erro cometido ao se tentar modelar  $y(t)$  em função dos seus valores atrasados é  $e(t)$ . Ampliando a equação (2.27) temos:

$$y(t) + a_1y(t-1) + \dots + a_{n_a}y(t-n_a) = e(t). \quad (2.28)$$

Ao acrescentar ao modelo AR uma entrada externa  $u(t)$ , que também é uma série temporal de uma variável que é admitida como parte explicativa do comportamento de  $y(t)$ , teremos modelo ARX conforme a equação (2.29).

$$A(q)y(k) = B(q)u(k) + e(k). \quad (2.29)$$



Logo, o modelo ARX é obtido a partir da equação (2.21) realizando  $C(q) = D(q) = F(q) = 1$ , sendo  $A(q)$  e  $B(q)$  polinômios arbitrários e, considerando que  $A(q)y(k)$  é a parte regressiva e  $B(q)u(k)$  a entrada externa. Desta forma é obtida a equação (2.29).

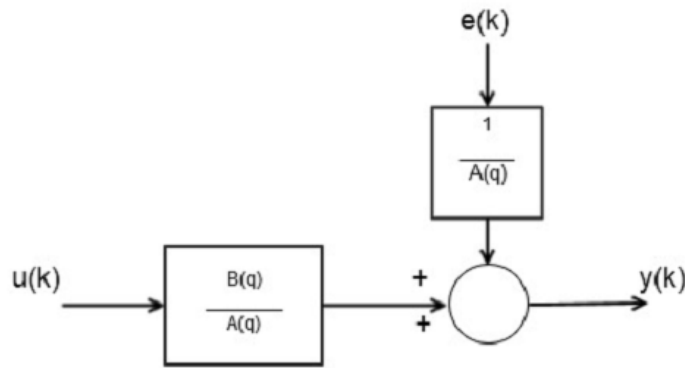
Surge o erro  $e(k)$  diretamente na equação, então o modelo ARX pode ser classificado como um modelo de erro da equação (AGUIRRE, 2004).

A equação (2.29) pode ser reescrita, resultando em:

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{1}{A(q)}e(k), \quad (2.30)$$

o que coloca em evidência as funções de transferência do sistema  $H(q) = B(q)/A(q)$  e de ruído  $C(q)/[D(q)A(q)] = 1/A(q)$  (LJUNG, 1987).

A representação ARX pode ser representada pela figura 2.7.



**Figura 2.7:** Representação do modelo ARX  
Fonte: (AGUIRRE, 2004).

A equação (2.29) pode ser reescrita em termos dos regressores e parâmetros dos polinômios  $A(q)$  e  $B(q)$  da seguinte forma:

$$y(k) = \psi^T \theta + e(k), \quad (2.31)$$

sendo  $\psi^T = [y(k-1) \ y(k-2) \cdots y(k-n_a) \ u(k-1) \ u(k-2) \cdots u(k-n_b)]$  e  $\theta^T = [a_1 \ a_2 \ \cdots \ a_{n_a} \ b_1 \ b_2 \ \cdots \ b_{n_b}]$ . Portanto,  $y(k)$  é a saída,  $\psi$  e  $\theta$  são denominados vetores de medidas e de parâmetros respectivamente,  $e(k)$  os ruídos (AGUIRRE, 2007b). Na seção 2.3.5 será visto o estimador de mínimos quadrados.

### 2.3.5 O Estimador dos Mínimos Quadrados

Após a determinação da estrutura de um modelo, deve-se estimar seus parâmetros para que se aproxime do comportamento real do sistema. Um dos métodos mais

conhecidos para obtenção de parâmetros a partir de dados experimentais é o método dos mínimos quadrados (AGUIRRE, 2004).

Levando-se em conta um conjunto de dados, o método de mínimos quadrados procura um melhor ajustamento para esses dados a fim de minimizar a soma dos quadrados das diferenças entre o valor estimado e os dados observados. O objetivo principal desse método é determinar o melhor ajuste do modelo aos dados experimentais a partir da minimização do erro. Com base em (AGUIRRE, 2004), é apresentado um sistema de equação com solução única e posteriormente o caso sobredeterminado, em que há mais equações do que incógnitas.

Considera-se uma função escalar  $y = f(x)$  aplicada a  $N$  valores de  $x$ , de forma que:

$$\begin{aligned} y_1 &= f(x_1) \\ y_2 &= f(x_2) \\ &\vdots \\ y_N &= f(x_N) \end{aligned} \tag{2.32}$$

No caso vetorial  $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$  depende de um vetor  $\theta$  de  $n$  parâmetros. Dessa forma, a função  $f(x)$  é parametrizada por  $\theta \in \mathbb{R}^n$  e pode ser representada como:

$$y = f(x, \theta). \tag{2.33}$$

Assim, tem-se um conjunto de equações a partir de várias observações do escalar  $y$  (variável dependente) e do vetor de variáveis independentes, da seguinte forma:

$$\begin{aligned} y_1 &= f(x_1, \theta) \\ y_2 &= f(x_2, \theta) \\ &\vdots \\ y_N &= f(x_N, \theta), \end{aligned} \tag{2.34}$$

sendo que  $y_i$  é a  $i$ -ésima observação de  $y$ , e  $x_i = [x_{1i}, x_{2i}, \dots, x_{ni}]^T$  são as  $i$ -ésimas observações dos  $n$  elementos de vetor  $x$ . A função definida na equação (2.33) define uma família de equações, sendo que  $N$  membros dessa família estão representados na equação (2.34). Assim, cada membro será denominado *restrição*, ou seja, a equação (2.34) é um conjunto de  $N$  restrições da função descrita na equação (2.33).

Caso sejam conhecidos  $x_i$  e  $y_i$   $i \in \{1, 2, \dots, N\}$ , deseja-se determinar  $f$  e  $\theta$ . Para isso, serão feitas as seguintes considerações.

1. A função  $f$  e o vetor  $\theta$  não variam de uma restrição para outra, ou seja, todas as restrições são, de fato, da mesma equação.
2. A equação (2.33) pode ser escrita como

$$y = x^T \theta. \quad (2.35)$$

3. São consideradas  $n$  restrições, a fim de se ter  $n$  equações para determinar os  $n$  de  $\theta$ , de forma que  $N = n$ .

Da consideração 1 fica claro que em problemas de identificação de sistemas dinâmicos, normalmente supõe-se que o sistema seja invariante no tempo e que os sinais medidos sejam estacionários. A consideração 2 implica que  $f$  seja linear nos parâmetros. A partir das considerações acima, pode-se escrever a equação (2.33) da seguinte forma:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}, \quad (2.36)$$

$$y = X\theta,$$

sendo  $X \in \mathbb{R}^{n \times n}$  e  $x_i$  a  $i$ -ésima coluna de  $X$  (deve-se notar que  $x_i$  é um vetor coluna de  $n$  linhas, ou seja,  $x_i \in \mathbb{R}^n$  que é diferente de  $x_i \in \mathbb{R}$ ).

Na equação (2.36),  $y$  é a variável dependente, pois depende dos regressores  $x_1, x_2, \dots, x_n$ , que são também chamados de variáveis independentes.  $\theta$  é o vetor de parâmetros a determinar. Pode-se determinar o vetor de parâmetros invertendo  $X$  (desde que  $X$  seja não singular), ou seja:

$$\theta = X^{-1}y \quad (2.37)$$

Se houver  $N > n$  restrições da equação (2.33), tem-se um sistema sobre-determinado, de forma que  $X \in \mathbb{R}^{N \times n}$ ,  $y \in \mathbb{R}^{N \times 1}$  e  $\theta \in \mathbb{R}^{n \times 1}$ . Como a matriz  $X$  não é quadrada, ela não pode ser invertida. Entretanto, pré-multiplicando a equação (2.36) por  $X^T$  em ambos os lados tem-se:

$$X^T y = X^T X \theta. \quad (2.38)$$

Dessa forma,  $X^T y \in \mathbb{R}^{n \times 1}$  e  $X^T X \in \mathbb{R}^{n \times n}$ . Se  $X^T X$  for não singular, pode

ser invertida chegando-se a:

$$\theta = [X^T X]^{-1} X^T y, \quad (2.39)$$

sendo  $[X^T X]^{-1} X^T$  chamada de matriz pseudo inversa. A equação (2.39) resolve o problema de determinação de  $\theta$  quando se tem maior número de restrições do que parâmetros.

No método dos mínimos quadrados, supondo que se conhece o valor estimado do vetor de parâmetros  $\hat{\theta}$  e que é cometido um erro  $\xi$  ao se tentar explicar o valor observado  $y$  a partir dos regressores de  $x$  e de  $\hat{\theta}$ , ou seja:

$$y = x^T \hat{\theta} + \xi \quad (2.40)$$

Escrevendo de forma matricial, quando se tem  $N > n$  medições da equação (2.40):

$$y = X \hat{\theta} + \xi. \quad (2.41)$$

Deseja-se encontrar  $\hat{\theta}$  que minimize o valor do vetor de erros  $\xi$ . Este valor será dado pelo somatório do quadrado dos erros  $\xi$ , ou seja:

$$J = \sum_{i=1}^N \xi(i)^2 = \xi^T \xi = \|\xi\|^2. \quad (2.42)$$

O custo  $J$  é uma quantia que mostra o quanto o vetor  $\hat{\theta}$  se ajusta às medidas de  $y$  e  $X$ . Quanto menor for  $J$ , melhor será esse ajuste. Isolando  $\xi$  na equação (2.41) e substituindo na equação (2.42):

$$J = (y - X \hat{\theta})^T (y - X \hat{\theta}) = y^T y - y^T X \hat{\theta} - \hat{\theta}^T X^T y + \hat{\theta}^T X^T X \hat{\theta}. \quad (2.43)$$

Para encontrar o vetor de parâmetros  $\hat{\theta}$  que minimiza o valor de  $J$ , deve-se resolver  $\frac{\partial J}{\partial \hat{\theta}} = 0$ . Fazendo isso, tem-se

$$\begin{aligned} \frac{\partial J}{\partial \hat{\theta}} &= -(y^T X)^T - X^T y + (X^T X + X^T X) \hat{\theta}, \\ \frac{\partial J}{\partial \hat{\theta}} &= -(X^T y) - X^T y + 2X^T X \hat{\theta}, \\ &= -2(X^T y) + 2X^T X \hat{\theta}. \end{aligned} \quad (2.44)$$

Igualando a equação (2.44) a 0 tem-se:

$$\hat{\theta} = [X^T X]^{-1} X^T y. \quad (2.45)$$

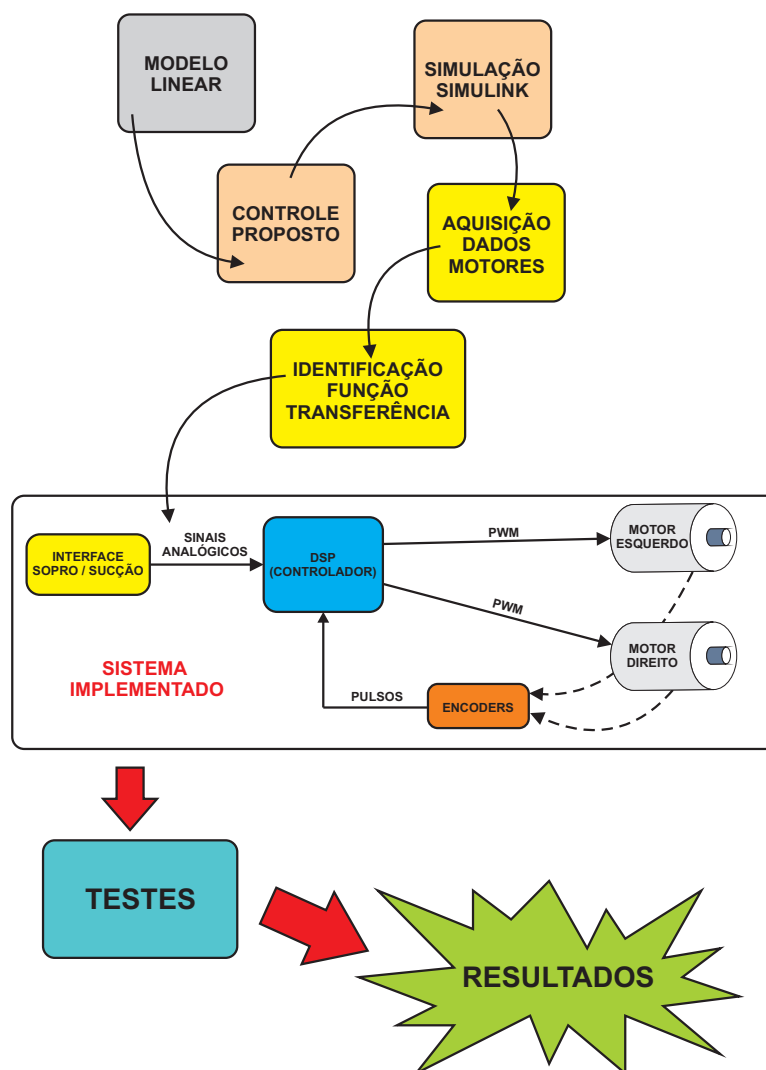
Para que  $\hat{\theta}$  corresponda ao mínimo de  $J$ , é necessário verificar que

$$\frac{\partial J^2}{\partial \hat{\theta}^2} = 2X^T X > 0,$$

o que é verdadeiro, pois  $2X^T X$  é positiva definida por construção (AGUIRRE, 2004). Então conclui-se que as equações (2.39) e (2.45) são iguais.

### 3 Descrição do Projeto Proposto

A proposta de projeto deste trabalho é a realização da implementação prática do controle da cadeira de rodas acionada por sopro e sucção. Para se obter êxito na implementação, várias etapas precisam ser realizadas. A figura 3.1 ilustra um esquema com as principais etapas necessárias no projeto.



**Figura 3.1:** Diagrama de blocos do desenvolvimento do projeto.  
Fonte: Próprio autor.

As etapas iniciais (modelo linear, controle proposto, simulação no Simulink,

aquisição de dados dos motores e identificação da função de transferência dos motores) compreendem o planejamento, dimensionamento e validação teórica do projeto. As demais etapas (implementação do sistema e testes) compreendem a parte prática, onde foram realizadas a programação, montagem e testes do sistema na cadeira de rodas. Os resultados obtidos na realização do projeto podem ser visualizados no Capítulo 4.

## 3.1 Dispositivos Eletrônicos

Nesta seção serão apresentados os dispositivos eletrônicos que foram utilizados para o desenvolvimento do projeto. No desenvolvimento do controle, utiliza-se o DSP para a programação de toda a funcionalidade do sistema. Os motores elétricos de corrente contínua são equipados com sensores *encoder* para leitura de rotação e ainda necessitam de uma interface de potência para acionamento. A condução da cadeira por sopro e sucção é realizada por uma interface que utiliza sensores de pressão de ar ou um sensor de fluxo de ar.

### 3.1.1 DSP *Texas Instruments* TMS320F28335

Para o desenvolvimento desse trabalho foi escolhido o DSP TMS320F28335 Del-fino, da Família C2000 da *Texas Instruments*. Dentre os diversos modelos de DSPs disponíveis no mercado, esse modelo se destaca pela facilidade de manuseio, além de possuir periféricos internos dedicados ao controle de motores, o que o torna muito interessante para esse trabalho (SILVA, 2012a).

A *Texas Instruments* disponibiliza esse DSP separadamente, ou montado em uma placa em formato de pente que pode ser encaixada em uma estação de trabalho, também disponibilizada pela empresa. Essa topologia de montagem favorece o desenvolvimento de projetos pela facilidade de troca de placa ou do sistema experimental envolvido. A placa e a estação de trabalho escolhidas podem ser vistas na figura 3.4. A plataforma de software de desenvolvimento IDE é o *Code Composer Studio (CCS)*.

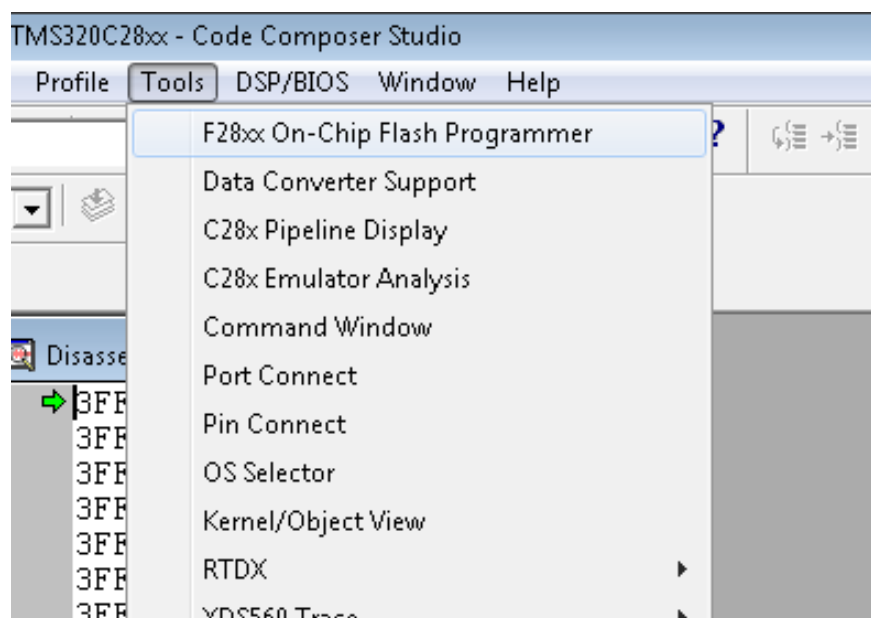
Para execução dos códigos nesse DSP, há duas possibilidades:

- **Online:** após a compilação do código programado, o CCS embarca (via conexão USB) as instruções de execução temporariamente na memória RAM (volátil) do DSP, e ele as executa enquanto tiver ligado à fonte, porém quando desligado esse código é perdido. Esse modo de execução tem a van-

tagem de serem mais rápidas a gravação e execução, agilizando o processo de desenvolvimento e ajustes de código e a possibilidade de usar o modo de depuração de código.

- **Offline:** diferentemente do modo online, no modo offline o CCS embarca (via conexão USB) as instruções de execução em um banco de memória Flash (não volátil) do DSP, e quando é ligado, uma instrução do código primeiramente lê a memória Flash, armazena as instruções na memória RAM e as executam. Esse modo tem como desvantagem a maior demora para embarcar o código e não possui a possibilidade de usar o modo *Debug*. Entretanto, para aplicações móveis e para versão final do projeto, se faz necessário esse modo de execução.

Para fazer a gravação do código fonte na memória Flash do DSP no CCS, deve-se acessar o menu *Tools / F28xx On-Chip Flash Programmer* como mostrado na figura 3.2. Uma nova janela será aberta, onde realiza-se a gravação do código. A figura 3.3 mostra esta janela, onde no espaço indicado pela seta 1 deve-se endereçar o código fonte já compilado pelo CCS, a seta 2 indica a opção de gravação (*Erase, Program and Verify*) e por fim a seta 3 indica o botão *Execute Operation*, que inicia o processo de gravação.



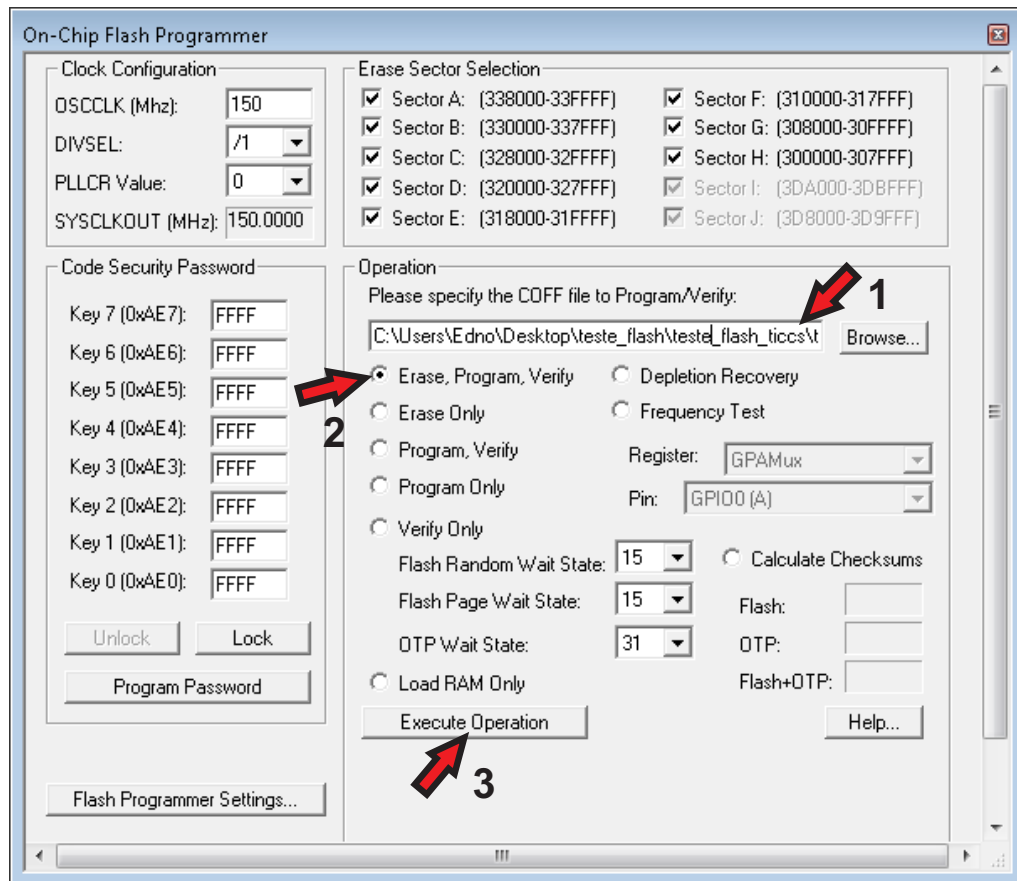
**Figura 3.2:** Menu do CCS.

Fonte: Próprio autor.

As principais especificações do DSP com estação de trabalho são:

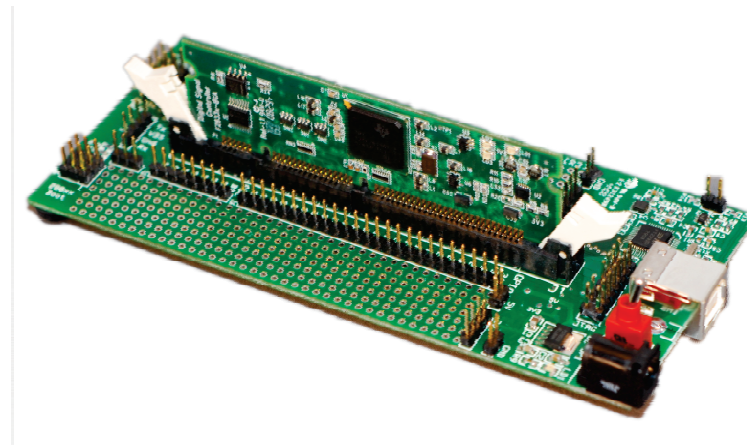
- Processador Digital de Sinais.





**Figura 3.3:** Menu de gravação do CCS.

Fonte: Próprio autor.



**Figura 3.4:** Estação de trabalho e DSP TMS320F28335.

Fonte: (INSTRUMENTS, 2011).

- Velocidade de Operação de 150 MHz.
- Unidades de ponto flutuante de 32 bits.
- 68Kbytes de memória RAM.
- 512Kbytes de memória Flash.
- 256Kbytes de memória SRAM.

- 16 Entradas analógicas com Conversor A/D de 12 bits com variação de tensão de entrada entre 0 e 3V e máxima taxa de conversão de 12.5 MSPS ou 80ns.
- 6 Módulos de PWM com saídas normal e complementar.
- Clock de Entrada de 30 MHz.
- Conexão USB.
- Interface CAN 2.0.
- Opera com 5V e pode ser alimentado pela USB ou por fonte DC.

O diagrama funcional interno do DSP TMS320F28335 é mostrado na figura 3.5.

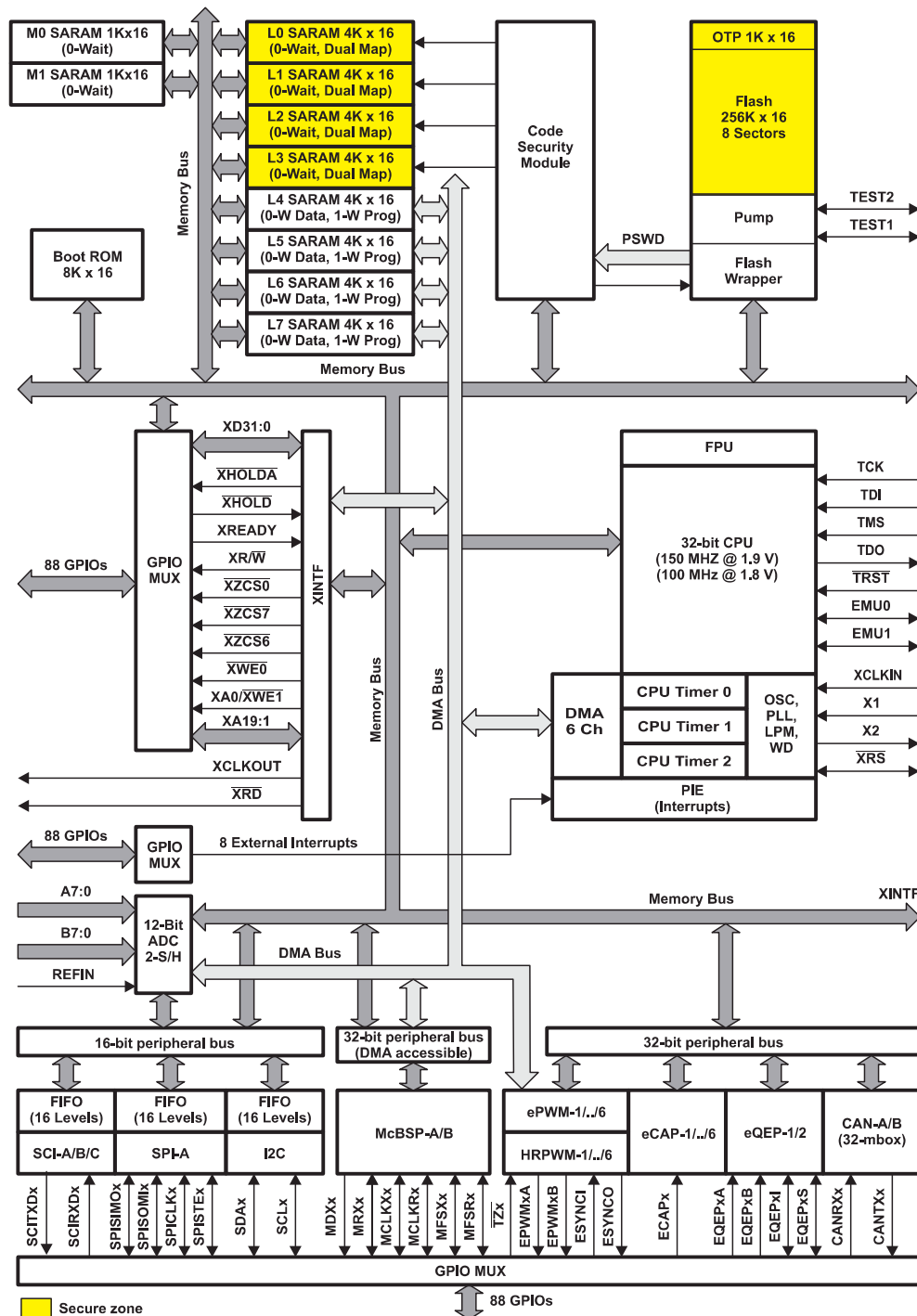
### 3.1.2 Motores Elétricos e Acionamento de Potência

A cadeira de rodas possui 2 motores elétricos de corrente contínua, sendo estes fabricados pela *Bosch* (ilustrado na figura 3.6) e suas especificações principais são:

- **Tensão nominal:** 12V.
- **Potência nominal:** 240W.
- **Torque nominal:** 75Ncm.
- **Velocidade nominal:** 3025rpm.
- **Corrente nominal:** 25,3A.

A figura 3.7 mostra graficamente as curvas de corrente (I), torque (M), velocidade (n) e potência (P) dos motores.

O acionamento dos motores é feito por sinais PWM provenientes do sistema de controle embarcado no DSP. Entretanto, estes sinais PWM não possuem potência suficiente para acionar os motores. Para que seja possível o acionamento é necessária uma interface capaz de fornecer potência aos motores. Uma das técnicas mais conhecidas para realizar a interface de potência de motores de corrente contínua é a Ponte-H. Esse circuito foi utilizado pois é bastante prático seu funcionamento, possibilitando a inversão de rotação dos motores e o chaveamento por PWM pelo controlador com simplicidade.



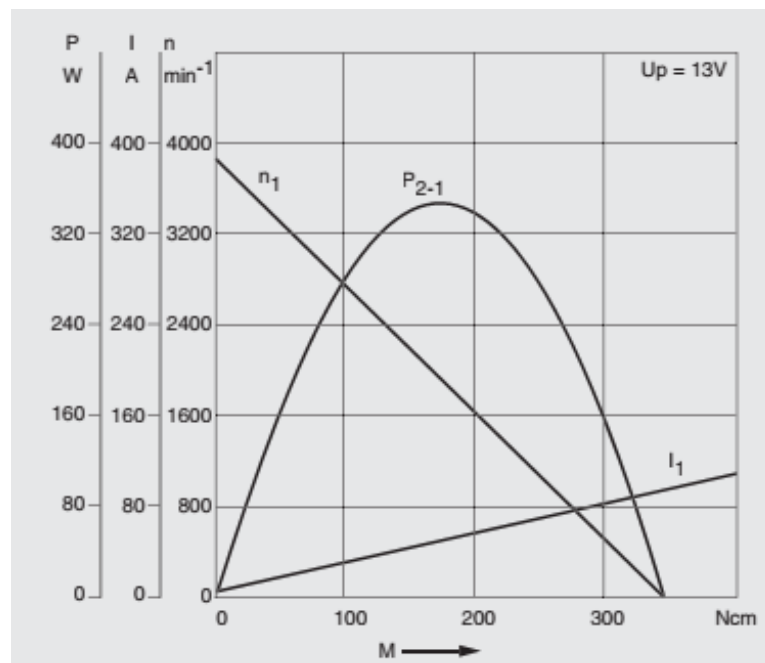
**Figura 3.5:** Diagrama funcional do DSP TMS320F28335.

Fonte: (INSTRUMENTS, 2011).

A ponte-H utilizada para acionamento dos motores é composta com 4 transistores de potência e 2 transistores para habilitar o seu funcionamento. O esquema lógico de construção é mostrado na figura 3.8. Quando aplica-se sinal lógico na entrada de direção A o motor gira em sentido horário e na entrada de direção B, gira em sentido anti-horário. Para determinar a intensidade de acionamento aplica-se um sinal de pulsos na entrada PWM.



**Figura 3.6:** Motor da cadeira de rodas.  
Fonte: Manual *Bosch* .



**Figura 3.7:** Gráfico da curva do motor.  
Fonte: Manual *Bosch*.

A figura 3.9 mostra a foto do sistema eletrônico de controle desenvolvido para a cadeira de rodas, onde tem-se destacados a ponte-H e o DSP.

### 3.1.3 Sensor *Encoder*

A leitura de velocidade dos motores elétricos da cadeira de rodas é muito importante, pois é através dessa informação que é feita a realimentação da malha fechada do controle. Uma das maneiras mais utilizadas para leitura de velocidade de motores é por meio de sensores tipo *encoder*.

*Encoders* são dispositivos que transformam movimentos em pulsos elétricos.

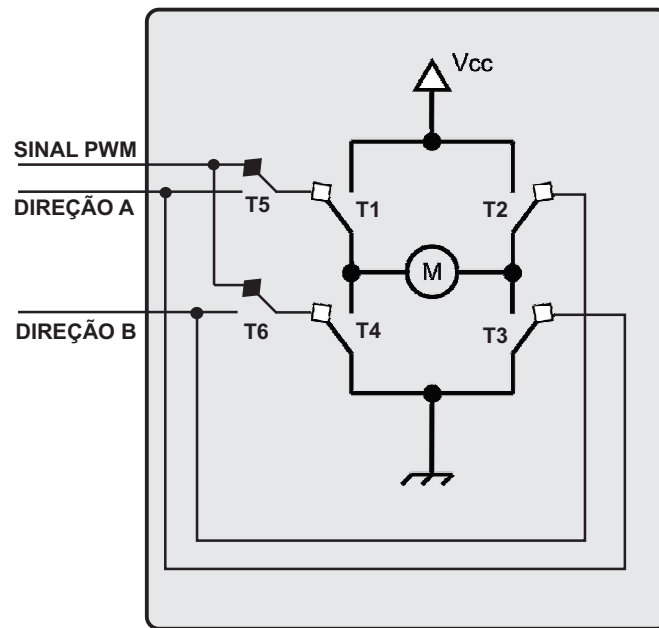


Figura 3.8: Esquema da ponte-H.  
Fonte: Próprio autor.

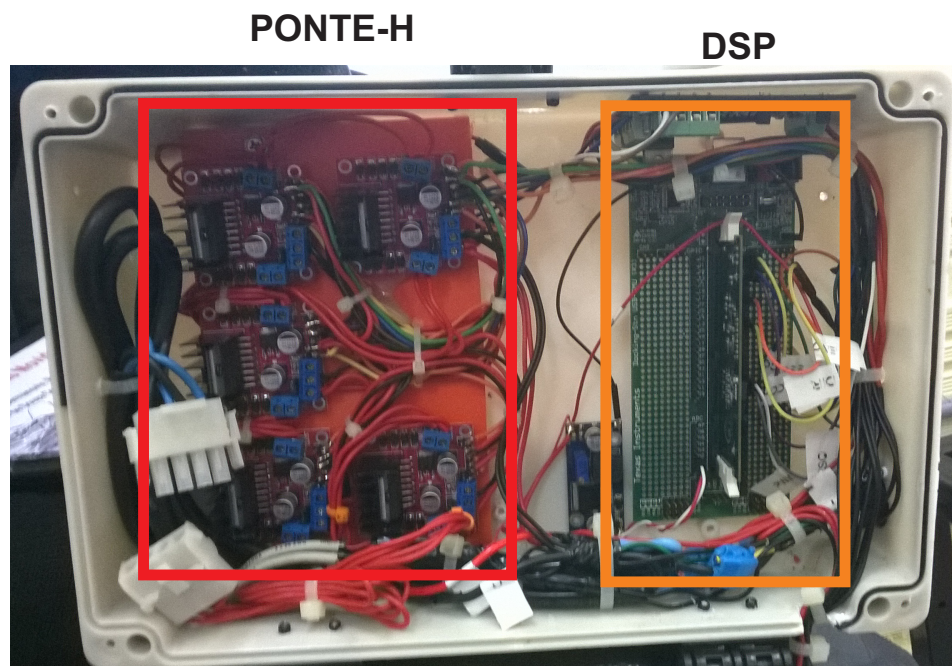
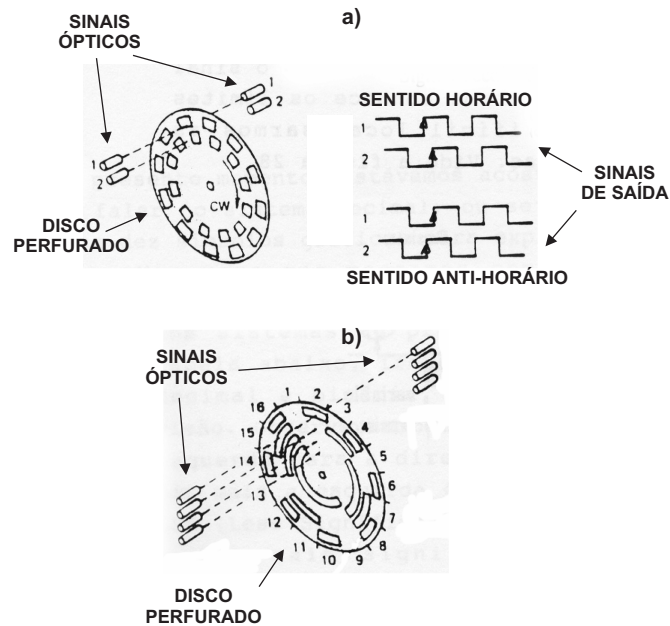


Figura 3.9: Foto do circuito eletrônico de controle da cadeira de rodas.  
Fonte: Próprio autor.

Existem dois tipos deste sensor:

- **Encoder incremental:** com esse tipo de *encoder* é possível ler a velocidade do motor através de pulsos elétricos e o sentido de rotação através do ângulo de fase entre os sinais de saída.
- **Encoder absoluto:** nesse tipo de sensor além das informações de velocidade e sentido de rotação, também é possível obter a posição em que o eixo do motor se encontra.

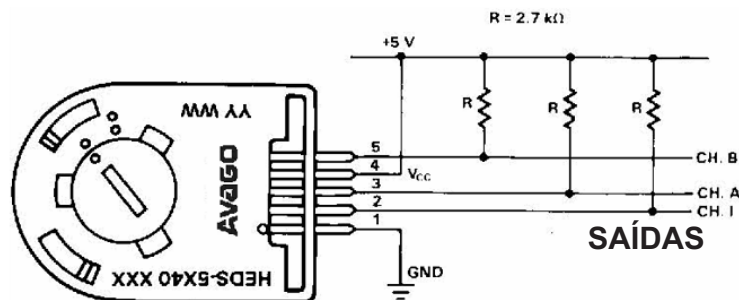
A figura 3.10 ilustra um esquema básico de um *encoder* incremental e um *encoder* absoluto, respectivamente.



**Figura 3.10:** a) *Encoder* incremental, b) *Encoder* absoluto  
 Fonte: Próprio autor.

Devido ao fato da aplicação desse trabalho não necessitar da informação da posição do eixo do motor, foi utilizado um *encoder* óptico incremental para a medição de rotação de motores. O modelo escolhido foi de fabricação *Avago Technologies* HEDS-5645 ilustrado na figura 3.11, suas principais características são:

- Tensão de alimentação: 5V.
- Velocidade máxima: 30000rpm.
- Resolução: 360passos/volta.
- Saídas: 3 saídas com nível TTL 5V.



**Figura 3.11:** *Encoder* Avago HEDS-5645  
 Fonte: Manual Avago Technologies.

### 3.1.4 Interface de Sopro e Sucção

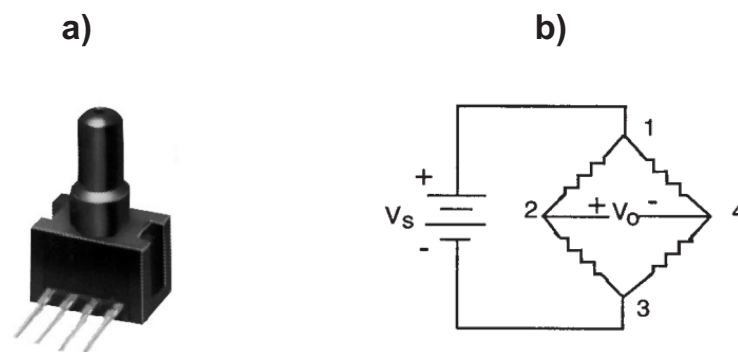
A interface de sopro e sucção é a parte responsável pela condução da cadeira de rodas pelo usuário com deficiência física.

A utilização do sopro e sucção para a realização do acionamento da cadeira se faz útil pela facilidade de acionamento por pessoas que possuem paralisação total ou parcial das mãos. Nenhuma funcionalidade adicional ou especial foi realizada através dessas interfaces neste trabalho. Duas interfaces foram utilizadas, sendo as opções: uma com sensores de pressão de ar e outra com sensor de fluxo de ar.

A interface de sopro e sucção que utiliza sensores de pressão de ar é equipada com dois sensores de fabricação *Honeywell* e modelo 24PCEFA6G (GERMANOVIX; CERVANTES, 2013). As principais características deste sensor são:

- **Tensão de alimentação:** 10V.
- **Pressão nominal de trabalho:** 0,5psi.
- **Sensibilidade de pressão:** 70mV/psi.
- **Tensão nominal de saída:** -35 a 35mV.
- **Tempo de resposta:** 1ms.

A figura 3.12 ilustra o sensor e o circuito esquemático interno, em que os pinos 1 e 3 são responsáveis pela alimentação, o pino 2 é a saída de sinal para pressão positiva e o pino 4 é a saída de sinal para pressão negativa.

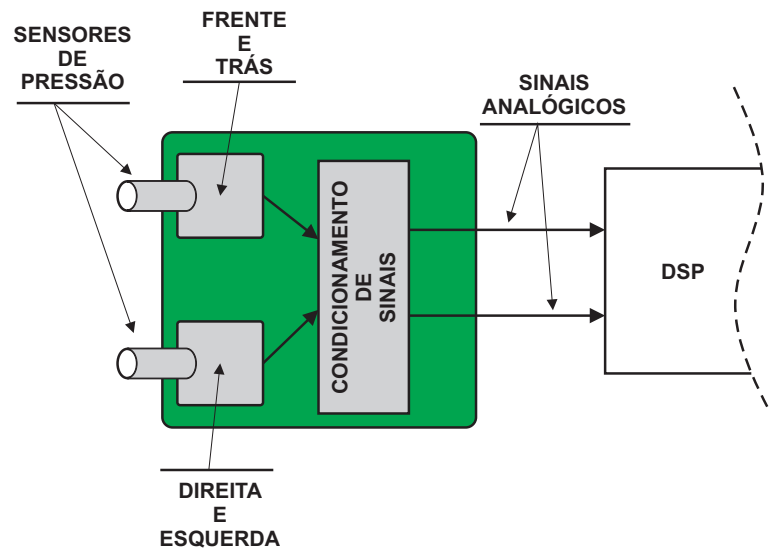


**Figura 3.12:** a) Sensor de pressão *Honeywell* 24PCEFA6G, b) Circuito esquemático interno do sensor.

Fonte: Manual *Honeywell* .

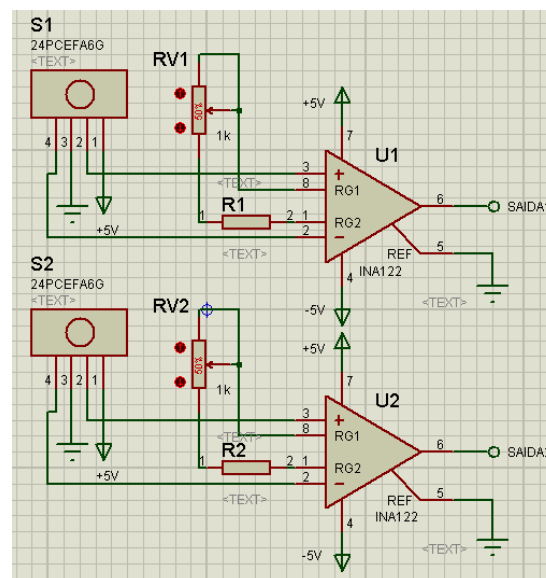
A figura 3.13 mostra um esquema desta interface:

Os sensores de pressão de ar possuem capacidade de leitura do sentido da pressão (negativa para sucção e positiva para sopro) e da intensidade. Entretanto



**Figura 3.13:** Diagrama da interface de sopro e sucção com sensor de pressão.  
Fonte: Próprio autor.

para a aplicação na cadeira de rodas, usuários com deficiência física possuem dificuldade em realizar acionamentos sensíveis com a boca. Pensando nisso, um circuito de condicionamento de sinais foi adicionado aos sensores. Desta forma os sensores funcionam como chaves (ligado ou desligado), ou seja, com o uso deste circuito a leitura de intensidade de pressão é descartada e também os níveis de tensão nas saídas analógicas são adequados. A figura 3.14 ilustra o circuito de condicionamento dos sensores.



**Figura 3.14:** Circuito de condicionamento de sinais dos sensores de pressão  
Fonte: Próprio autor, adaptado de (LOPES, 2012).

O esquema de funcionamento desta interface é demonstrado na tabela 3.1, na qual pode ser observado que há um sensor para cada tipo de movimento, ou seja, de acordo com o movimento desejado deve-se aplicar sopro ou sucção no sensor correto. Nas saídas 1 e 2 estão presentes os sinais elétricos que são ligados ao



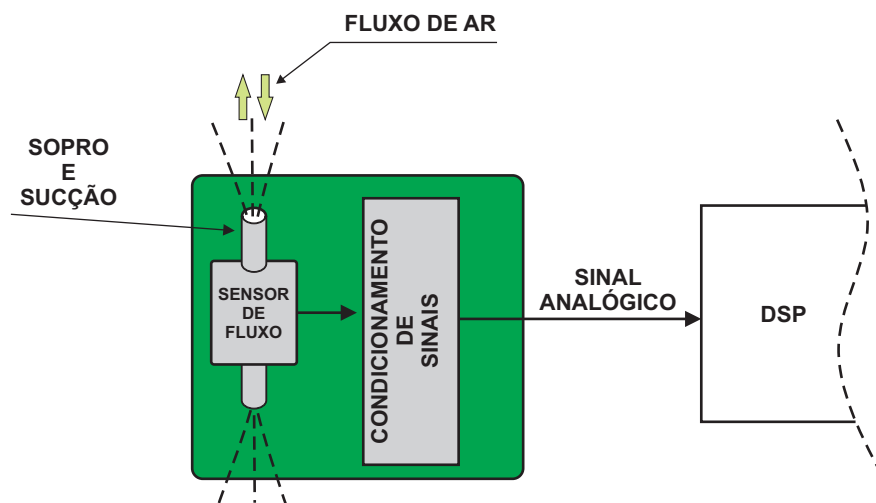
DSP.

| Movimento | Sensor 1 | Sensor 2 | Saída 1 | Saída 2 |
|-----------|----------|----------|---------|---------|
| PARADO    | –        | –        | 1,4V    | 1,4V    |
| FRENTE    | SOPRO    | –        | 2,5V    | 1,4V    |
| TRÁS      | SUCÇÃO   | –        | 0V      | 1,4V    |
| DIREITA   | –        | SOPRO    | 1,4V    | 2,5V    |
| ESQUERDA  | –        | SUCÇÃO   | 1,4V    | 0V      |

**Tabela 3.1:** Dados de funcionamento da interface de sopro e sucção (sensor de pressão de ar).

Fonte: Próprio autor.

A outra interface utilizada é equipada com um sensor de fluxo de ar. Sua função é a mesma, ou seja, realizar o acionamento da cadeira de rodas através de sopro e sucções. Entretanto sua construção é diferente, utiliza apenas um sensor e possui apenas uma saída de sinal analógico (SOBRINHO et al., 2000; FERREIRA, 2008). A figura 3.15 mostra o diagrama desta interface.



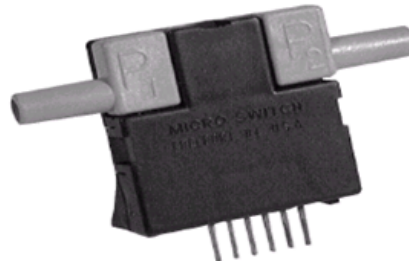
**Figura 3.15:** Diagrama da interface de sopro e sucção com sensor de fluxo.

Fonte: Próprio autor.

O sensor de fluxo de ar utilizado é de fabricação *Honeywell* modelo AWM2100V (ilustrado na figura 3.16) cujas principais características são:

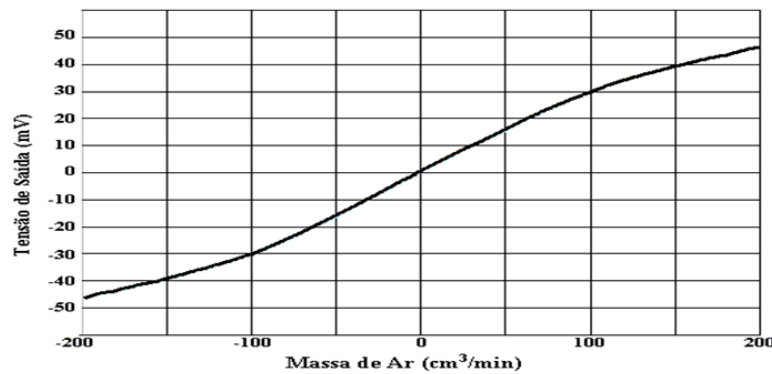
- **Tensão de alimentação:** 10V.
- **Direção do fluxo de ar:** bidirecional.
- **Tensão de saída:** -60 a +60mV.
- **Tempo de resposta:** 1ms.

Esse sensor utiliza o princípio da variação de temperatura para a medição do fluxo de ar, ou seja, internamente possui um sistema gerador de calor e de



**Figura 3.16:** Sensor de fluxo de ar - AWM2100V.  
Fonte: Próprio autor.

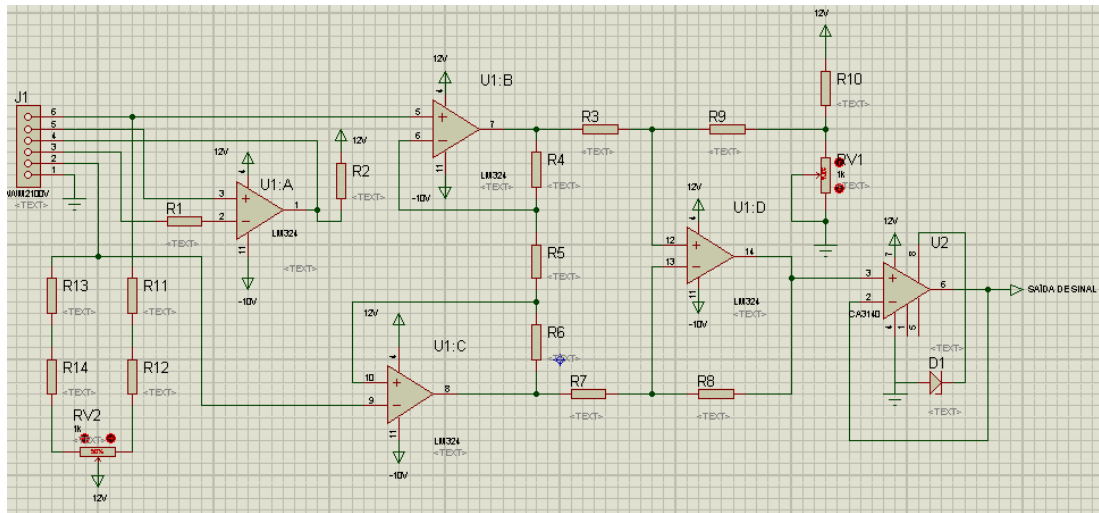
acordo com a intensidade do fluxo de ar que passa através dele a temperatura varia proporcionalmente com a quantidade de ar. Basicamente é construído com dois circuitos de ponte de Wheatstone, uma para controle do circuito gerador de temperatura e outra para medição da variação de temperatura de acordo com o fluxo de ar. A curva de tensão característica do sensor é mostrada na figura 3.17.



**Figura 3.17:** Curva característica do sensor de fluxo de ar.  
Fonte: Manual *Honeywell*.

Como observado na figura 3.17 o sensor de fluxo ar também apresenta uma capacidade de leitura de intensidade. Entretanto, de forma análoga, essa funcionalidade foi descartada e sensor é aplicado como uma chave (ligado ou desligado). Um circuito de condicionamento de sinais foi utilizado para adequação dos níveis de tensão e calibração do sensor. A figura 3.18 ilustra do circuito de condicionamento.

Esta interface possui apenas uma saída de sinal, o que a torna mais complexa para a realização do acionamento de todas as direções da cadeira. A alternativa utilizada para solucionar essa dificuldade foi temporizar cada acionamento. Desta forma, para um mesmo tipo de acionamento é possível determinar direções distintas, sendo que ao selecionar uma determinada direção o sistema fica acionado até que haja uma nova seleção. O esquema de acionamento está demonstrado na tabela 3.2.



**Figura 3.18:** Circuito de condicionamento e ajuste de sinal do sensor de fluxo.  
Fonte: Próprio autor, adaptado de (HONEYWHEEL, 2005).

| Movimento | Sensor          | Tempo | Saída |
|-----------|-----------------|-------|-------|
| PARADO    | SOPRO ou SUCÇÃO | 0,5s  | 2,5V  |
| FRENTE    | SOPRO           | 1s    | 0V    |
| TRÁS      | SOPRO           | 1,5s  | 0V    |
| DIREITA   | SUCÇÃO          | 1,5s  | 5V    |
| ESQUERDA  | SUCÇÃO          | 1s    | 5V    |

**Tabela 3.2:** Dados de funcionamento da interface de sopro e sucção (sensor de fluxo de ar).

Fonte: Próprio autor.

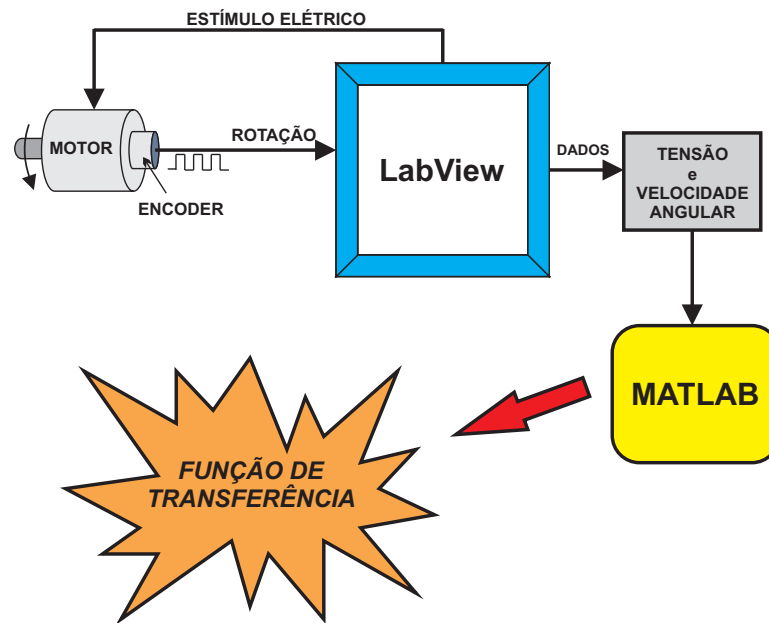
## 3.2 Identificação da Função de Transferência dos Motores

Para o desenvolvimento de um sistema de controle é necessário que o sistema possua um modelo matemático que represente, de forma mais próxima possível, o sistema real.

Neste trabalho os objetos controlados são os motores elétricos. Então foi realizada a identificação de sistemas para obtenção das funções de transferências dos motores. O esquema principal da identificação dos motores é ilustrado na figura 3.19. Nesse processo, utilizou-se uma placa de aquisição de dados da fabricante *National Instruments* modelo PCI6024E e os softwares *LabView* e *MatLab*.

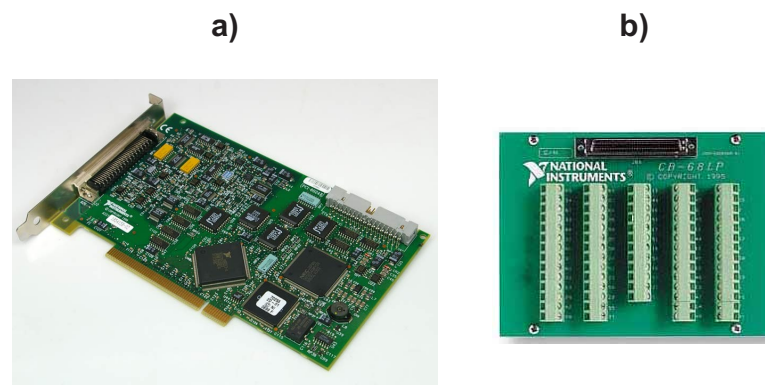
### 3.2.1 Aquisição de Dados

Uma das formas de fazer aquisição de dados em sistemas elétricos é através de plataformas computacionais (*Hardware/Software*), onde o *hardware* é responsável por coletar as grandezas elétricas, transformá-las em dados e enviar ao *software*.



**Figura 3.19:** Esquema de identificação dos motores.  
Fonte: Próprio autor.

No processo de identificação das funções de transferências dos motores elétricos foi realizada a aquisição de dados utilizando uma placa (DAQ) modelo PCI6024E e uma placa de conexões da *National Instruments*, como pode ser visto na figura 3.20. A placas **Série E** são placas multifunções de baixo custo e com as seguintes características e tecnologias: circuitos de proteção contra flutuações de temperaturas, tecnologia para melhorar a resolução, auto calibração *Onboard*, possui um controle projetado para prover maior flexibilidade, baixa potência de consumo e alta imunidade a ruído.



**Figura 3.20:** a) Placa de aquisição *National Instruments* PCI6024E, b) Placa de conexões.

Fonte: Próprio autor.

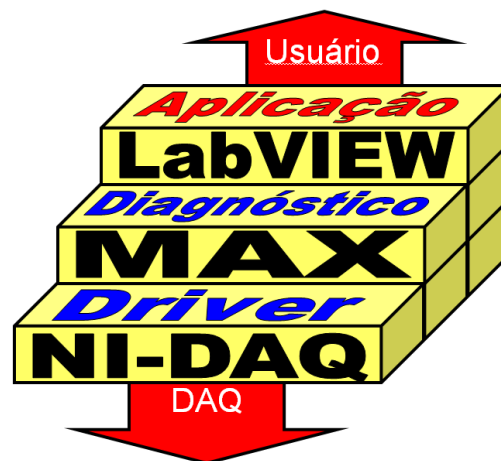
Essa placa DAQ possui as especificações:

- **Conexão com computador:** Barramento PCI.
- **Entradas analógicas:** 16 portas, resolução de 12 bits, taxa de amostragem

máxima de 200kS/s.

- **Saídas analógicas:** 2 portas, resolução de 12 bits, taxa de amostragem máxima de 10kS/s.
- **Portas digitais I/O:** 8 portas (TTL/CMOS).
- **Entrada/Saída de pulso:** 2 portas com contadores de 24 bits.
- **Buffer de entrada:** FIFO de 2048 amostras.
- **Tensão de trabalho:**  $\pm 10V$ .
- **Impedância de entrada:** 100G $\Omega$ .

O Software *LabView* possui diversas ferramentas poderosas principalmente para aquisição de dados. Primeiramente é importante salientar que essa plataforma computacional é dividida em 3 camadas, como mostra a figura 3.21.



**Figura 3.21:** Diagrama de blocos das camadas do software.  
Fonte: Próprio autor.

Cada camada possui uma função específica para a aquisição de dados:

- **NI-DAQ:** essa camada representa o *Driver*, que possui a função de conectar o *hardware* ao *software*, ou seja, para que haja uma conexão e troca de dados entre o *software* e a placa de aquisição (*hardware*) é necessário o uso do NI-DAQ.
- **NI-MAX:** essa camada possui uma função intermediária entre o *driver* e a aplicação. Ela é responsável por gerenciar as atividades e configurar a placa de aquisição de acordo com as necessidades especificadas pela aplicação. Adicionalmente também realiza a função de diagnóstico, por exemplo, quando um sinal que está sendo medido está em uma frequência

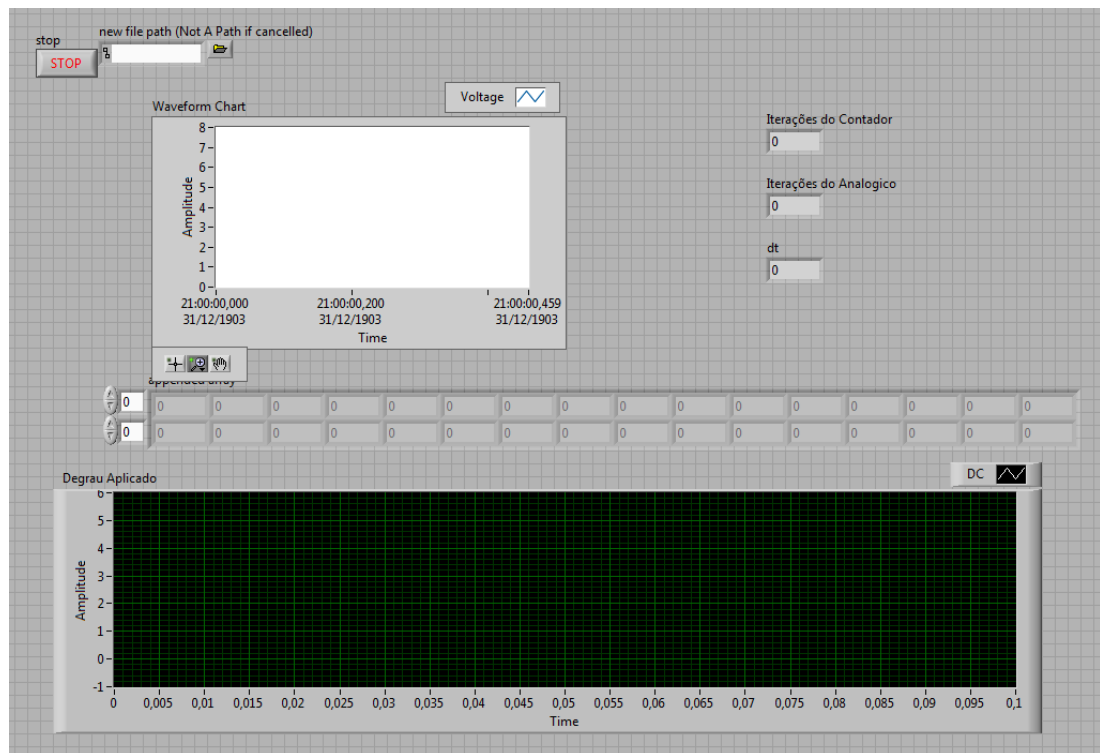
maior do que a capacidade de leitura da placa DAQ, o NI-MAX detecta um erro de leitura e reporta à aplicação.

- **LabView:** o *LabView* compreende a camada mais superior do *software*, considerado a aplicação. É nessa camada que o usuário pode manipular as funções do sistema de aquisição de dados, ou seja, a aplicação é a interface de comunicação com o usuário. O *LabView* possui um grande número de ferramentas integradas no mesmo ambiente, em sua maioria para aquisição de dados, entretanto não se limita somente a isso, possui também ferramentas de controle, identificação de sistemas, armazenamento de dados, comunicação em rede, interface visual homem-máquina, entre muitas outras (GAINO et al., 2002). O usuário pode fazer a programação da maneira mais conveniente para sua necessidade, utilizando uma linguagem de blocos particular da *National Instruments* ou algumas outras linguagens como *C*, por exemplo. Para melhor visualização, o *LabView* é dividido em duas partes: painel frontal, responsável pela visualização dos dados e aplicações, e painel traseiro, responsável pela programação da aplicação.

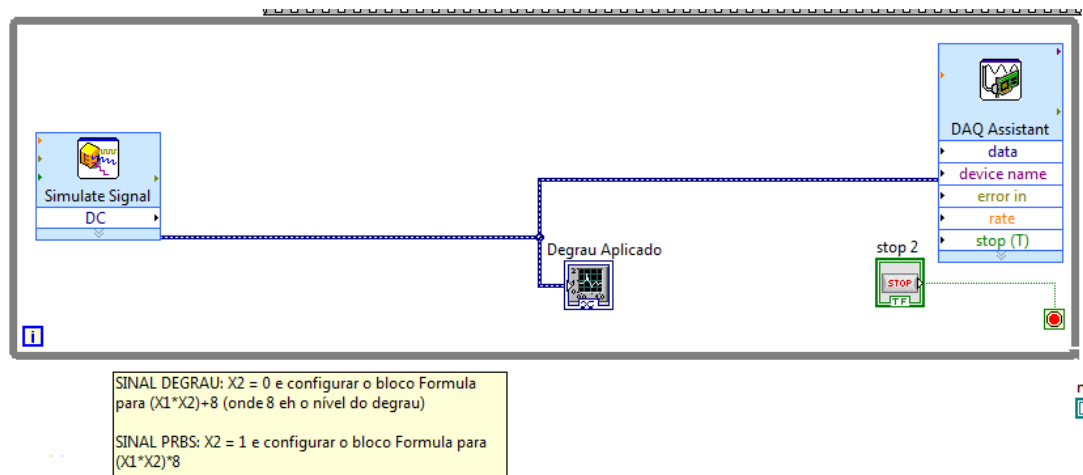
O programa de aquisição de dados utilizado neste trabalho foi desenvolvido em LabView. Resumidamente, o funcionamento deste programa pode ser descrito da seguinte forma:

Primeiramente o programa gera um sinal degrau com uma tensão contínua de 10V e aplica em uma saída analógica da placa DAQ, que tem a função de acionar o motor. Então o programa começa a fazer a leitura dos dados. Um contador é responsável por sincronizar as leituras de maneira que todas aconteçam simultaneamente.

- **Interface gráfica:** essa é a parte do programa em que o usuário pode interagir, tendo a opção de ligar ou desligar o programa e selecionar onde o programa irá salvar os dados. A interface é ilustrada na figura 3.22, e fornece as informações gráficas e numéricas de tensão e velocidade angular do motor.
- **Geração do estímulo elétrico:** nessa parte o programa gera o sinal degrau de tensão contínua igual a 10V na saída analógica da placa DAQ, que realiza a função de acionar o motor para a coleta de dados. Vale salientar que a placa DAQ não possui capacidade para fornecer corrente a circuitos externos, então para o acionamento do motor foi utilizada uma interface de potência ponte-H. A figura 3.23 ilustra essa parte do programa.



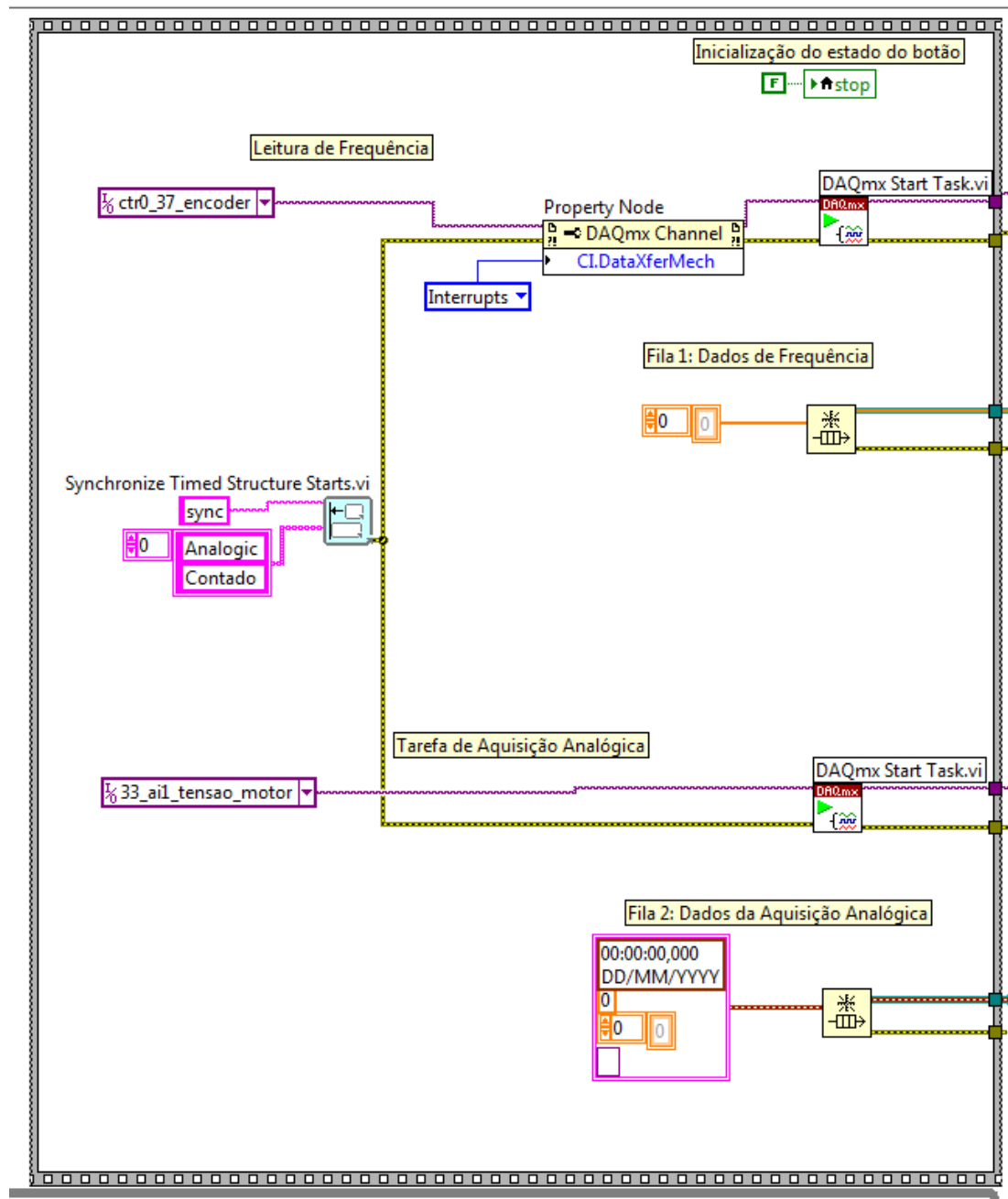
**Figura 3.22:** Interface Gráfica do programa de aquisição de dados (*LabView*).  
Fonte: Próprio autor, adaptado de (SILVA, 2012b).



**Figura 3.23:** Parte do programa responsável pela geração de estímulo elétrico para o motor.

Fonte: Próprio autor, adaptado de (SILVA, 2012b).

- **Leitura dos dados:** essa parte, ilustrada na figura 3.24, é responsável por ler a tensão do motor e a frequência de pulsos provenientes do *encoder*. Para isso utiliza uma entrada analógica e uma entrada de pulsos (contador) da placa DAQ. Na leitura dos pulsos, as amostras são coletadas no instante de tempo que chegam, pois são eventos aleatórios que dependem do sistema externo. Já na leitura da tensão, o programa faz a coleta em amostras com intervalo de tempo de  $100\mu s$ .



**Figura 3.24:** Parte do programa responsável pela leitura de tensão do motor e frequência do *encoder*.

Fonte: Próprio autor, adaptado de (SILVA, 2012b).

- **Sincronização dos dados:** nessa parte os dados coletados são colocados em filas (uma para a frequência e uma para a tensão). Então, esses elementos são armazenados em *buffers*. Os dados desses *buffers* são lidos e colocados em novas filas para serem enviados ao arquivo de texto. Tanto no *loop* que controla o *buffer* de frequência quanto no *loop* que controla o *buffer* de tensão existem contadores para sincronizar essa atividade, garantindo que os dados de frequência e tensão sejam enviados juntos, em um intervalo de tempo de  $1ms$  (SILVA, 2012b).

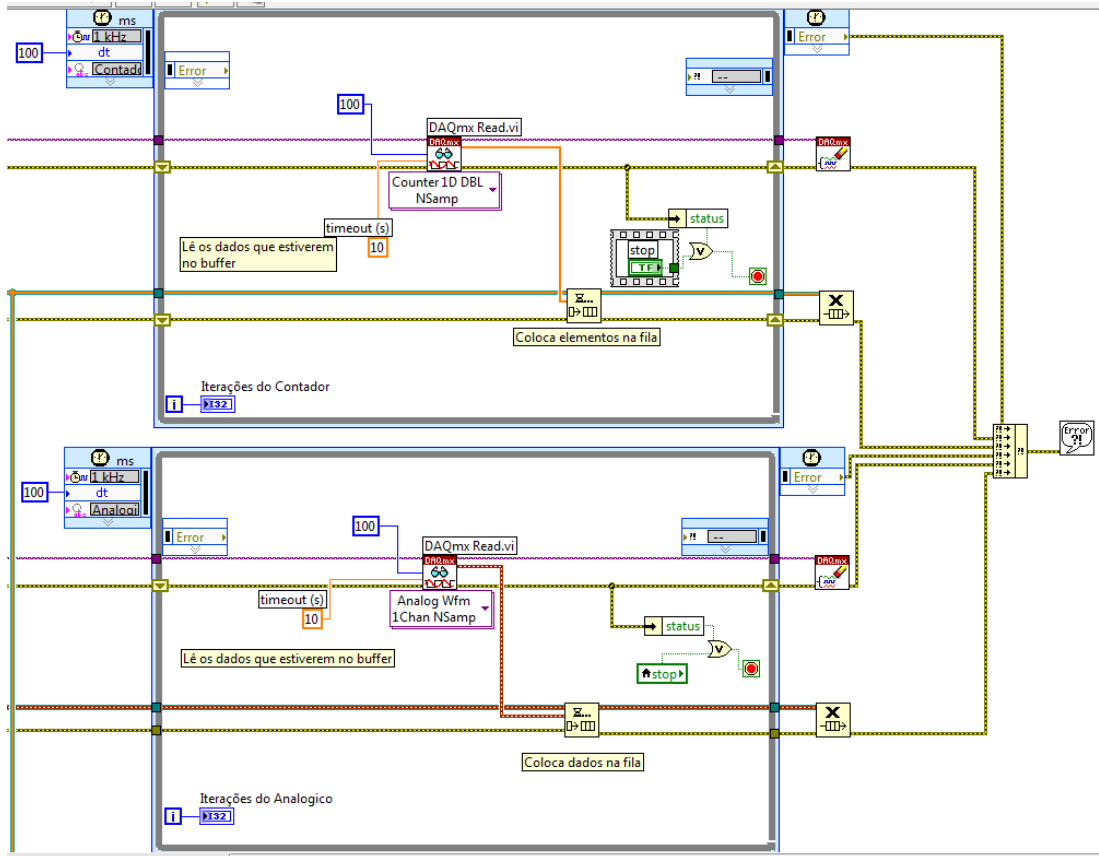
Daí então, as filas são lidas e separadas para serem gravadas no arquivo



de texto. Simultaneamente, os valores são mostrados de forma gráfica e indicativa no painel frontal (figura 3.22). A partir daí, é gerado um arquivo de texto com os valores de tensão em volts e os valores de frequência em radianos por segundo. É importante salientar que os valores de frequência são captados pelo programa em *Hertz* e na última etapa, antes de gerar o arquivo de texto, é utilizado um bloco “Formula”, que multiplica os valores de frequência por  $2\pi$  para obter valores em *rad/s*. No mesmo bloco, o resultado é dividido por 360. Esta divisão é feita para se obter a frequência de giro do motor. Esse cálculo é definido por:

$$X_1 = \frac{f \cdot 2\pi}{n}, \quad (3.1)$$

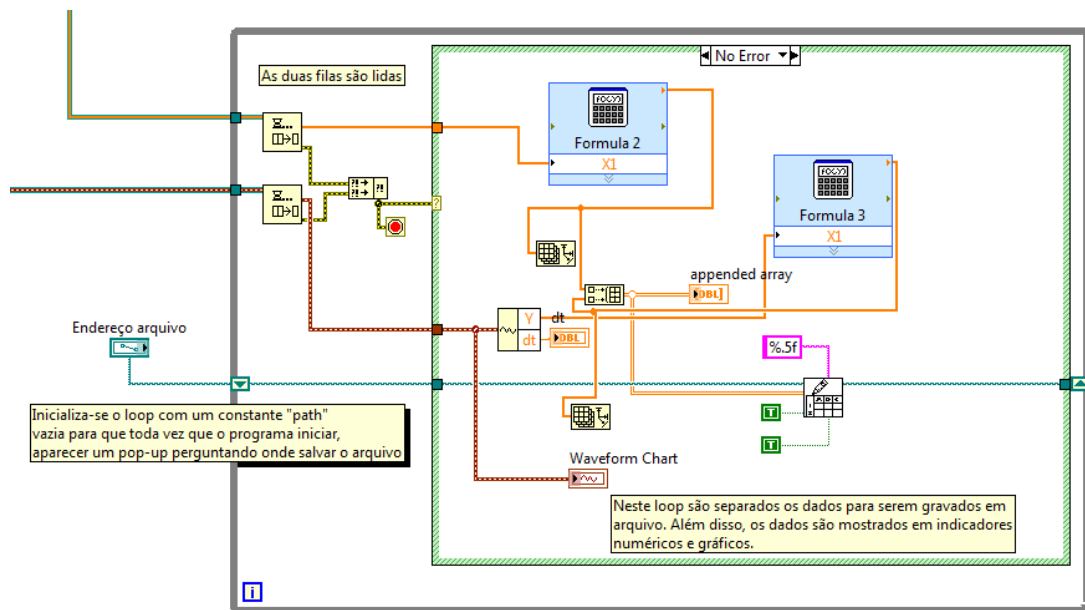
onde  $X_1$  é velocidade angular em *rad/s*,  $f$  a frequência em *Hertz* e  $n$  a quantidade de furos do disco do *encoder*.



**Figura 3.25:** Parte do programa responsável pela sincronização dos dados.  
Fonte: Próprio autor, adaptado de (SILVA, 2012b).

- **Armazenamento dos dados:** nessa etapa, os dados coletados, sincronizados e organizados em filas, são enviados à parte responsável pela gravação dos dados em um arquivo de texto. Entretanto, como já mencionado anteriormente, antes dos dados serem gravados no arquivo de texto, uma conversão da frequência é realizada. Os dados de frequência são coletados pelo

programa em unidade *Hertz*, e por conveniência para os cálculos futuros, são convertidos em velocidade angular em *rad/s* por meio de (3.1) presente dentro do bloco “Formula”. A figura 3.26 ilustra esta fração do programa.



**Figura 3.26:** Parte do programa responsável pelo armazenamento dos dados.  
Fonte: Próprio autor, adaptado de (SILVA, 2012b).

### 3.2.2 Obtenção das Funções de Transferências dos Motores com Matlab

A Identificação de Sistemas constrói modelos matemáticos a partir de dados medidos, através do procedimento de aquisição de dados descrito na subseção 3.2.1. Para esse projeto os dados medidos foram: tensão (V) e velocidade angular (rad/s) dos motores, sendo esses dados entrada e saída do sistema, respectivamente.

Um problema relevante no processo de identificação é a coleta de dados com ruídos. No mundo real, praticamente todas as medidas apresentam algum tipo de ruído, por esse motivo é necessária uma filtragem nos dados medidos antes de fazer o processo matemático para estimar o modelo do sistema.

Um filtro *Butterworth* passa-baixa de primeira ordem foi projetado para a filtragem dos dados. Esse filtro foi desenvolvido no Matlab, através da *toolbox* **FDAtool**, e possui uma frequência de amostragem de  $300\text{kHz}$  e uma frequência de corte de  $50\text{Hz}$ . O código do filtro segue abaixo:

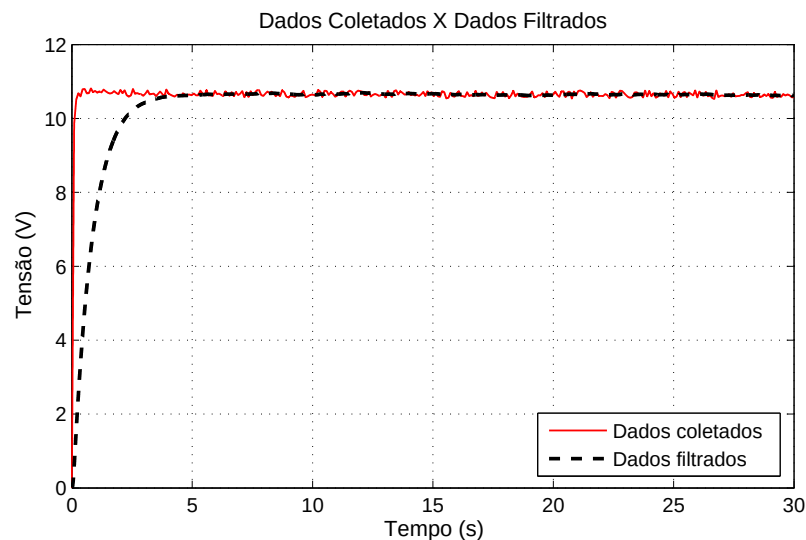
```
function Hd = filtro50_1
```

```

%FILTRO50_1 Returns a discrete-time filter object.
% MATLAB Code
% Generated by MATLAB(R) 7.12 and the Signal Processing Toolbox 6.15.
% Generated on: 05-May-2013 09:28:37
% Butterworth Lowpass filter designed using FDESIGN.LOWPASS.
% All frequency values are in Hz.
Fs = 300000; % Sampling Frequency
N = 1; % Order
Fc = 50; % Cutoff Frequency
% Construct an FDESIGN object and call its BUTTER method.
h = fdesign.lowpass('N,F3dB', N, Fc, Fs);
Hd = design(h, 'butter');

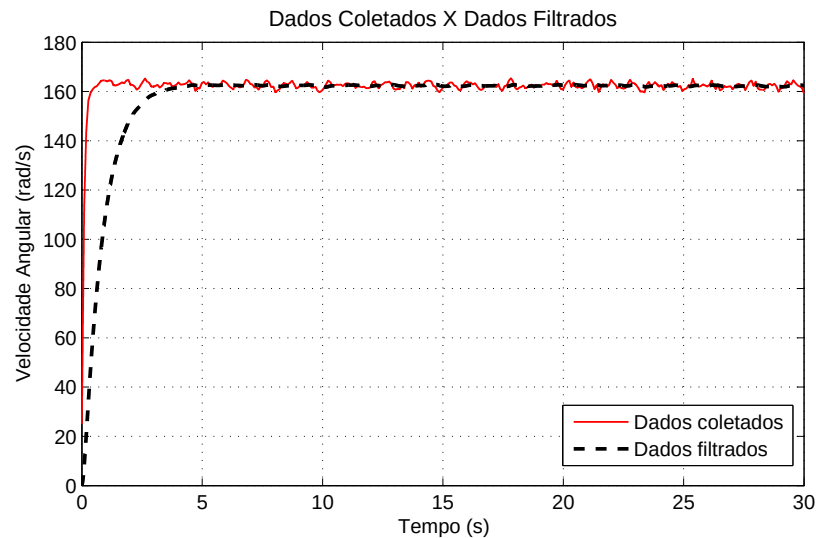
```

Após a execução do código do filtro no Matlab, dados filtrados foram gerados, possibilitando assim a estimação do modelo matemático. As figuras 3.27 e 3.28 mostram os dados coletados e filtrados do motor direito (como exemplo), referente à tensão e velocidade angular respectivamente.



**Figura 3.27:** Dados de tensão do motor (coletados e filtrados).  
Fonte: Próprio autor.

Na identificação, a escolha de uma estrutura que representa o comportamento do sistema dinâmico desejado é importante. Primeiramente, deve-se definir se a representação é linear ou não linear e também escolher os métodos paramétricos e não paramétricos que serão utilizados para estimar os parâmetros do modelo (COELHO; COELHO, 2004). Em sistemas de controle, tem-se a preferência por modelos lineares do grande número de técnicas de controle baseadas nessas estruturas (AGUIRRE, 2004). Definida a representação que será usada, o próximo passo é escolher a estrutura do modelo. Para (AGUIRRE, 2004), a escolha da estrutura



**Figura 3.28:** Dados de velocidade angular do motor (coletados e filtrados).

Fonte: Próprio autor.

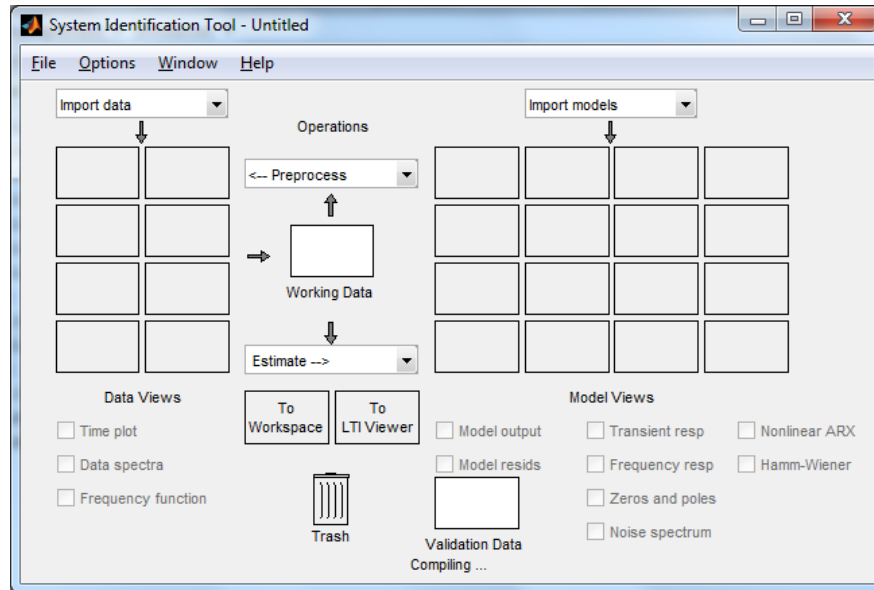
do modelo é subjetiva, por depender de vários aspectos: do uso pretendido, tipo de problema, da disponibilidade e qualidade dos dados utilizados, experiência do usuário e a complexidade do sistema. Além disto, que a escolha de estrutura do modelo é uma tarefa normalmente feita por tentativa e erro (SOUZA, 2014).

Ljung (1987) sugere que primeiramente a escolha de uma estrutura deve ser iniciada por modelos mais simples, como o ARX (*Autoregressive with exogenous inputs*), que normalmente é a estrutura mais facilmente estimada. Desta forma, a utilização de uma estrutura mais complexa apenas se justifica se a mais simples não for suficiente. Por esses motivos o modelo ARX foi escolhido para este trabalho.

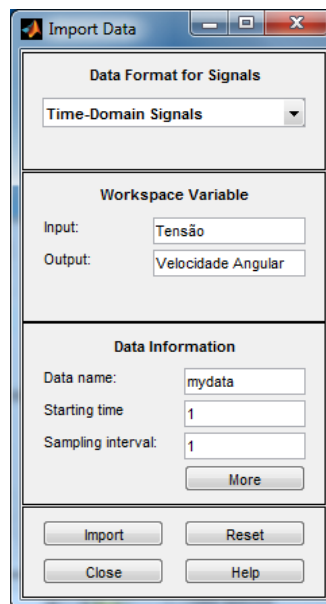
Existem várias maneiras para a estimação do modelo, neste trabalho foi utilizada a *Toolbox* de identificação de sistemas do Matlab, que possui uma interface gráfica de fácil manipulação pelo usuário (figura 3.29). Essa ferramenta constrói modelos matemáticos de sistemas dinâmicos utilizando os dados obtidos da entrada e saída do sistema (LJUNG, 2007).

Para a utilização dessa *Toolbox* primeiramente deve-se carregar os dados do sistema na aba *Import Data*, no caso desse projeto os dados são tensão como entrada e velocidade angular como saída. A janela do programa usada para essa etapa é mostrada na figura 3.30.

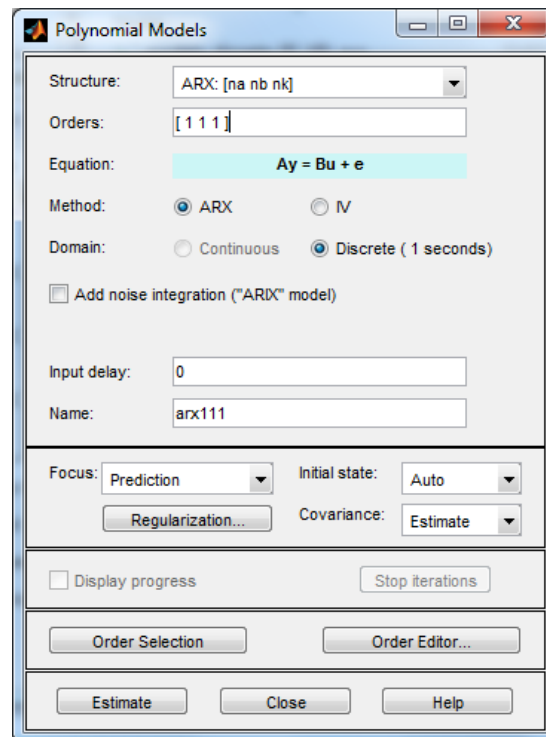
Utilizando a opção *Estimate > Linear parametric models* desta ferramenta, ilustrada na figura 3.31, pode-se estimar modelos paramétricos regressivos ARX. Na figura 3.32, observam-se os principais modelos ARX ( $na$  = números de polos,  $nb$  = número de zeros,  $nk$  = atraso de tempo) obtidos.



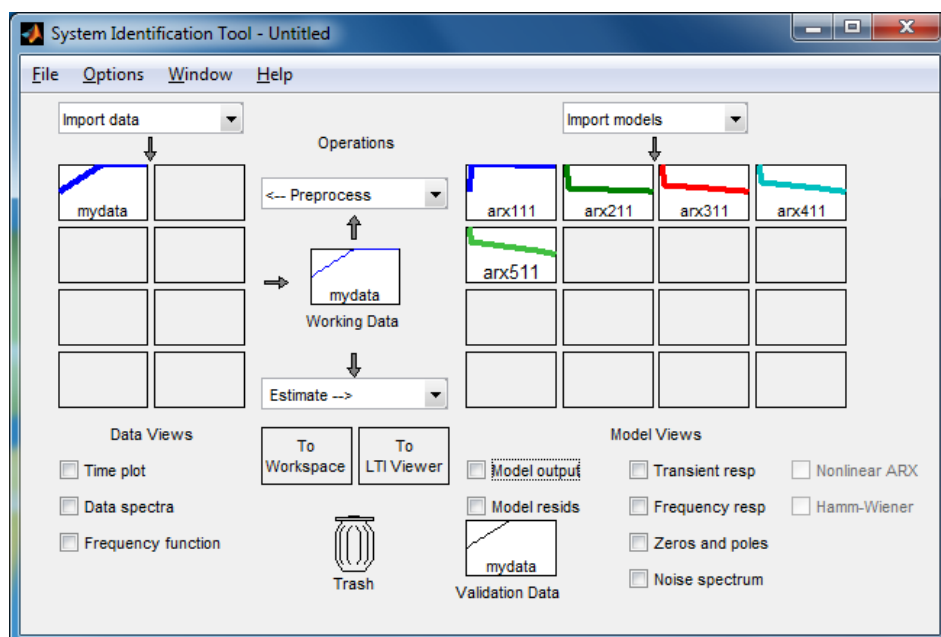
**Figura 3.29:** *Toolbox* de identificação de sistemas do Matlab  
Fonte: Próprio autor.



**Figura 3.30:** Janela para carregar os dados na *Toolbox* do Matlab  
Fonte: Próprio autor.



**Figura 3.31:** Janela para determinação da ordem do modelo  
Fonte: Próprio autor.



**Figura 3.32:** Modelos estimados  
Fonte: Próprio autor.

A estimação dos modelos foi feita para ambos os motores (esquerdo e direito) com a cadeira sem e com carga<sup>1</sup> e para movimentos linear (frente/trás) e giro (com carga).

Foram estimados modelos ARX de primeira ordem. A *toolbox* de identificação calcula a porcentagem de aproximação do sistema real. A tabela 3.3 contém a porcentagem de aproximação dos modelos estimados em relação ao sistema real

| Motor            | Aproximação |
|------------------|-------------|
| Direito (Vazio)  | 98,4%       |
| Direito (Carga)  | 97,1%       |
| Direito (Giro)   | 94,9%       |
| Esquerdo (Vazio) | 98,8%       |
| Esquerdo (Carga) | 97,1%       |
| Esquerdo (Giro)  | 95,4%       |

**Tabela 3.3:** Percentual de aproximação dos modelos estimados e do sistema real.

Fonte: Próprio autor.

O que se procura em um modelo para aplicação em controle é um modelo de menor ordem possível, que consiga se adequar satisfatoriamente aos dados (AGUIRRE, 2007b). Sabe-se pela literatura (AGUIRRE, 2004; KOSOW, 1985; FITZGERALD; KINGSLEY; KUSKO, 1975) que o motor de corrente contínua possui um modelo de primeira ordem. As estruturas obtidas pela *Toolbox* de Identificação são selecionadas com um índice que qualifica, em termos percentuais, o modelo estimado, que utiliza para estimação o estimador de Mínimos Quadrados, se for escolhida a estrutura estatística ARX.

Então é preciso escolher qual modelo utilizar. Soderstrom e Stoica (1989) e Aguirre (2004) sugerem a utilização do princípio da parcimônia. Esse princípio é uma regra útil para a determinação da ordem de um modelo apropriado, ou seja, diante de dois ou mais modelos com ajustes igualmente bons, escolhe-se aquele com o menor número de parâmetros. As justificativas para isso são: *a)* simplicidade na estrutura identificada e *b)* não usar parâmetros extras para descrever um sistema dinâmico, se eles não são necessários. Desta forma, a estrutura escolhida para representar os motores é a  $ARX(na = 1, nb = 1, nk = 1)$  por ter uma pequena complexidade e uma satisfatória aproximação com o modelo real, facilitando assim a implementação do controle.

Para a determinação da função de transferência é preciso exportar o modelo desejado, no caso deste projeto o ARX111, para o *workspace* do Matlab. Desta forma, é possível aplicar a função *Transfer Function* e obter a função de trans-

<sup>1</sup>A coleta de dados dos motores foi realizada com uma carga de 75Kg.

ferência. O código utilizado para essa função é descrito abaixo:

```
sys=tf(arx441); % obtenção da função transferência  
funcao=d2c(sys,'zoh'); % conversão para função contínua no tempo
```

Desta forma, foram encontradas as funções de transferências  $H_1(s)$  para o motor direito vazio,  $H_2(s)$  para o motor direito com carga,  $H_3(s)$  para o motor esquerdo vazio,  $H_4(s)$  para o motor esquerdo com carga e por fim,  $H_5(s)$  e  $H_6(s)$  para giro com carga dos motores direito e esquerdo respectivamente. As funções são mostradas abaixo:

$$H_1(s) = \frac{397,3}{s + 28,38}, \quad (3.2)$$

$$H_2(s) = \frac{0,03165}{s + 0,002328}, \quad (3.3)$$

$$H_3(s) = \frac{0,535}{s + 0,03509}, \quad (3.4)$$

$$H_4(s) = \frac{0,02673}{s + 0,001843}, \quad (3.5)$$

$$H_5(s) = \frac{0,02875}{s + 0,002215}, \quad (3.6)$$

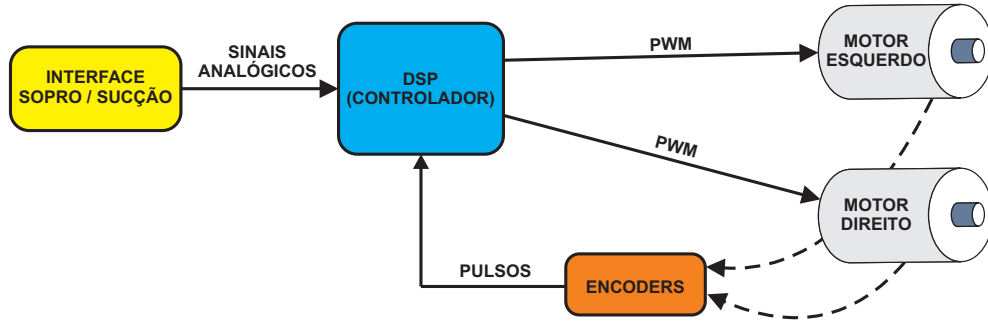
$$H_6(s) = \frac{0,01818}{s + 0,001309}. \quad (3.7)$$

### 3.3 Implementação do Controle

Nesta seção serão descritos o sistema de controle da cadeira de rodas e os procedimentos realizados na implementação.

O sistema de controle compreende os controladores PID dos motores e o programa de gerenciamento de tarefas, que realiza o monitoramento da instrumentação, aplica o controle PID e faz a atuação nos motores. A figura 3.33 ilustra o diagrama do sistema implementado na cadeira de rodas:

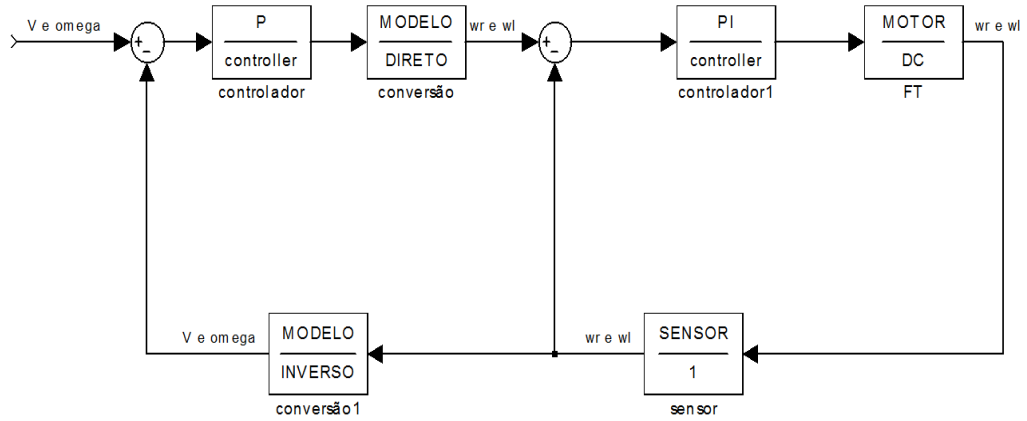




**Figura 3.33:** Diagrama principal do sistema implementado.  
Fonte: Próprio autor.

### 3.3.1 Controlador Proposto

Para a proposta do trabalho, um controle simples e eficaz é necessário. Adotou-se o sistema de controle idealizado primeiramente por (MAZO et al., 1995) e posteriormente adaptado por (SOBRINHO et al., 2003), pois nesses trabalhos mostrou-se um controle simplificado e eficaz para a proposta da cadeira de rodas. Com isso, reunindo o controle já trabalhado anteriormente e as novas tecnologias de identificação de sistemas e implementação em DSPs, deu-se a realização desse trabalho. A figura 3.34 mostra as variáveis do sistema de controle sendo  $V$  e  $\Omega$  as velocidades linear e angular, respectivamente, da cadeira.



**Figura 3.34:** Diagrama de blocos do sistema de controle  
Fonte: Próprio autor, adaptado de (MAZO et al., 1995; SOBRINHO et al., 2003).

O bloco “Modelo Direto” converte  $V$  e  $\Omega$  em  $\omega_l$  e  $\omega_r$ , conforme (3.8). O bloco “Modelo Inverso” converte  $\omega_l$  e  $\omega_r$  em  $V$  e  $\Omega$ , conforme (3.9).

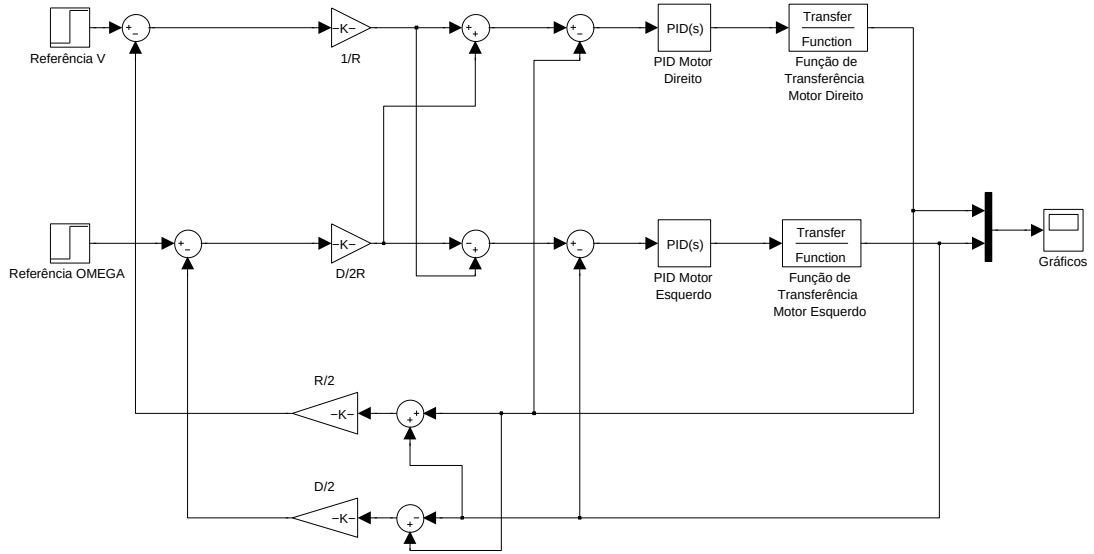
$$\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = \begin{bmatrix} \frac{1}{R} & \frac{D}{2R} \\ \frac{1}{R} & -\frac{D}{2R} \end{bmatrix} \cdot \begin{bmatrix} V \\ \Omega \end{bmatrix} \quad (3.8)$$

$$\begin{bmatrix} V \\ \Omega \end{bmatrix} = \begin{bmatrix} \frac{R}{2} & \frac{R}{2} \\ \frac{R}{D} & \frac{-R}{D} \end{bmatrix} \cdot \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \quad (3.9)$$

Para o motor DC, assumiu-se uma função de transferência simplificada, onde  $\omega$  é a velocidade angular,  $V_a$  é a tensão de armadura,  $K$  é o ganho,  $T$  é a constante de tempo do motor e  $m$  a relação de polias entre o eixo do motor e as rodas da cadeira, tem-se então (MAZO et al., 1995):

$$H(s) = \frac{\omega(s)}{V_a(s)} = \frac{K.m}{T.s + 1} \quad (3.10)$$

A figura 3.35, mostra o diagrama de blocos completo do controlador:



**Figura 3.35:** Diagrama de blocos do controlador.

Fonte: Próprio autor, adaptado de (SOBRINHO et al., 2003).

Os ganhos  $(1/R)$ ,  $(D/2R)$ ,  $(R/2)$  e  $(D/2)$  são calculados pelas medidas físicas da cadeira de rodas, sendo  $R$  o raio das rodas traseiras e  $D$  a distância entre elas. Para o modelo utilizado neste trabalho:  $D = 55cm$  e  $R = 25cm$ .

Assim como no mini protótipo, a cadeira de rodas também apresenta uma relação de transformação de velocidades das rodasXmotores. Os motores são ligados às rodas por meio de correias. A relação de velocidade pode ser calculada por:

$$Rel_{Transf} = \frac{D_{rodas}}{D_{eixo}} \quad (3.11)$$

onde  $D_{roda}$  é o diâmetro das rodas,  $D_{eixo}$  é o diâmetro dos eixos dos motores e  $Rel_{Transf}$  é a relação de transformação de velocidade entre os eixos dos motores e as rodas. no sistema da cadeira de rodas,  $D_{roda} = 50cm$  e  $D_{eixo} = 2,5cm$

### 3.3.2 Obtenção dos Ganhos do Controlador PID

O processo em que os parâmetros do controlador são calculados para um melhor índice de desempenho é conhecido como sintonia do controlador. A função dessa

sintonia é ajustar os valores para  $K_p$ ,  $K_i$  e  $K_d$  tomando como base a resposta experimental ao degrau ou ao valor crítico de  $K_p$  (OGATA, 2010). Esse processo de sintonia pode também ser desenvolvido de forma empírica, entretanto pode ser um trabalho difícil.

A cadeira de rodas necessita que sejam inseridos esses parâmetros em seu controlador, afim de proporcionar a estabilidade dos motores. A dificuldade em encontrar ganhos para o controlador usando o método de *Routh-Hurwitz* aumenta conforme a ordem da planta. Sendo essas plantas de ordem elevada, fica difícil a investigação das mudanças de sinal na primeira coluna de tabela de Routh (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

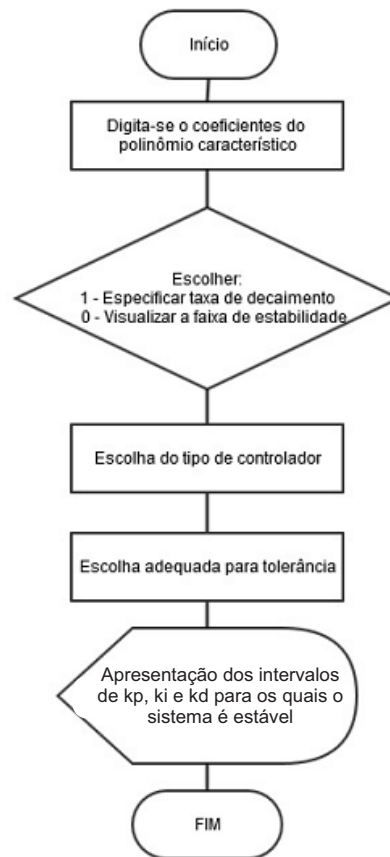
Os parâmetros do controlador PID utilizado neste trabalho foram encontrados pela aplicação de um algoritmo desenvolvido no Matlab por Teixeira, Assunção e Covacic (2007) (detalhes apresentados nas seções 2.2.3, 2.2.4 e 2.2.5), que foi de grande importância para a aplicação do controlador.

O algoritmo baseia-se no polinômio característico e determina um conjunto de ganhos do controlador que garante a estabilidade do sistema. Para a obtenção dos parâmetros, digitam-se os coeficientes do polinômio característico e obtêm-se valores para os parâmetros  $K_p$ ,  $K_i$ ,  $K_d$  respectivamente.

A execução do algoritmo começa digitando-se os coeficientes do numerador e denominador da função de transferência. Após isso, escolhem-se mais duas especificações: taxa de decaimento ou apenas estabilidade. A taxa de decaimento (BOYD et al., 1994) está relacionada com a velocidade de resposta do sistema: quanto maior for essa taxa, mais rápido será o transitório do sistema, e pode ser levado em consideração pelo algoritmo ou não, dependendo da escolha do usuário. Por fim, escolhe-se o tipo de controlador usado. Esse algoritmo contém um pequeno erro de aproximação quando se encontra as raízes reais dos termos da primeira coluna, assim, algumas raízes repetidas podem ser vistas como raízes diferentes. Dessa forma, para resolver esse problema, duas raízes reais adjacentes  $z_i$  e  $z_{i+1}$ , são consideradas diferentes se somente se  $|z_i - z_{i+1}| > \varepsilon$ , onde  $\varepsilon = 10^{-6}$  é a margem de tolerância (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007).

Para o caso da escolha do controlador PID, o programa solicita valores finais e iniciais que devem ser digitados para  $K_d$  e  $K_i$  respectivamente. Assim, obtém-se para cada par  $(K_d, K_i)$  a faixa de estabilidade de  $K_p$ . O fluxograma de funcionamento pode ser observado na figura 3.36.

Segundo o programa, realizado por (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007),



**Figura 3.36:** Fluxograma do Algoritmo de Ganho PID

Fonte: (TEIXEIRA; ASSUNÇÃO; COVACIC, 2007), adaptado por (SOUZA, 2014).

a tabela 3.4 apresenta um exemplo de faixa de valores obtidas pelo algoritmo para o motor esquerdo da cadeira de rodas (para exemplificação). Desta forma, pode-se observar a faixa de valores para aplicação dos parâmetros do controlador que garantem sua estabilidade do sistema.

Dentro da faixa de valores que garantem estabilidade obtidas pelo algoritmo, foram escolhidas as faixas de valores apresentadas na tabela 3.5 para serem utilizadas neste trabalho.

### 3.3.3 Implementação do Controlador no DSP

O controle proposto inicialmente para a cadeira de rodas sofreu alguns ajustes e modificações para um melhor desempenho prático. Em testes de laboratório, notou-se uma diferença de comportamento dos motores da cadeira em movimentos lineares e de giro. Essa diferença existe porque há diferenças de velocidade para os movimentos angular e linear, caracterizando pontos de operação diferentes. Então foi realizada a inclusão de um *Chaveamento de Controladores*, ou seja, para cada tipo de movimento é usado um controlador com parâmetros diferentes para atuar nos motores, proporcionando assim um melhor aproveitamento do

---

|                                       |                             |
|---------------------------------------|-----------------------------|
| For $k_d = -0.25$ and $k_i = -1.00$ : | The problem has no solution |
| For $k_d = -0.25$ and $k_i = -0.75$ : | The problem has no solution |
| For $k_d = -0.25$ and $k_i = -0.50$ : | The problem has no solution |
| For $k_d = -0.25$ and $k_i = -0.25$ : | The problem has no solution |
| For $k_d = -0.25$ and $k_i = 0.00$ :  | Solution: $k_p > -0.066586$ |
| For $k_d = -0.25$ and $k_i = 0.25$ :  | Solution: $k_p > -0.066586$ |
| For $k_d = -0.25$ and $k_i = 0.50$ :  | Solution: $k_p > -0.066586$ |
| For $k_d = -0.25$ and $k_i = 0.75$ :  | Solution: $k_p > -0.066586$ |
| For $k_d = -0.25$ and $k_i = 1.00$ :  | Solution: $k_p > -0.066586$ |
| $\vdots$                              | $\vdots$                    |

---

**Tabela 3.4:** Exemplo de faixas de valores para o PID dos motores.

Fonte: Próprio autor.

| Motor            | $K_d$ | $K_i$ | $K_p$         | $K_p$ Escolhido |
|------------------|-------|-------|---------------|-----------------|
| Direito (Vazio)  | -0.25 | -0.5  | $> -0.071432$ | 3.45            |
| Direito (Carga)  | -0.25 | 0.03  | $> -0.07355$  | 5.75            |
| Direito (Giro)   | -0.25 | 0.03  | $> -0.077043$ | 5.3             |
| Esquerdo (Vazio) | -0.25 | 0.03  | $> -0.065589$ | 6.75            |
| Esquerdo (Carga) | -0.25 | 0.03  | $> -0.068949$ | 7               |
| Esquerdo (Giro)  | -0.25 | 0.03  | $> -0.072002$ | 8.5             |

**Tabela 3.5:** Faixa de valores obtidas pelo algoritmo escolhidas para o projeto.

Fonte: Próprio autor.

controlador, pois em ponto de operação há uma função de transferência para cada motor. Esse modelo modificado é ilustrado na figura 3.37, e os resultados de simulações realizadas serão mostrados no capítulo 4.

O programa de controle da cadeira de rodas foi simulado no Simulink e posteriormente foi construído o código em linguagem C com o *software Code Composer Studio* (CCS) para a implementação no DSP *Texas Instruments* modelo TMS320F28335. O código em linguagem C está demonstrado no Apêndice C.

O fluxograma com funcionamento geral do programa da cadeira de rodas é ilustrado na figura 3.38.

Para a implementação do controlador, o programa ilustrado na figura 3.37 foi adaptado, pois para funcionamento no DSP mudanças precisaram ser feitas, tendo em vista que é uma plataforma computacional diferente do Matlab. Esse programa é ilustrado na figura 3.39 e a figura 3.40 ilustra um diagrama básico de blocos mostrando os módulos internos do DSP que foram utilizados para a implementação do programa.

Basicamente, o programa funciona da seguinte maneira: primeiramente as

portas ADs (provenientes dos sinais da interface de sopro e sucção) são lidas, então o bloco “ACIONAMENTO”<sup>2</sup> determina qual a direção de movimento que o usuário determinou. A partir daí, esse bloco é responsável por acionar a variável “sel” para selecionar o controlador correto (1 para movimento linear ou 2 para movimento angular), acionar as variáveis “*Digital Outputs*” para selecionar o sentido de giro de cada motor na interface de acionamento dos motores (ponte-H) e por fim aplicar os valores de referência para os controladores PID na variáveis  $V$  e  $W$ .

Depois de configurados os valores de referência, o sentido de rotação de cada motor e selecionado o controlador PID correto, a rotina do controlador entra em funcionamento. Essa rotina faz a leitura dos valores de referência, aplica o controle PID, que determina o valor de saída do mesmo e aplica o sinal nos módulos de PWM. Então, esse módulo converte o valor numérico da saída em pulsos modulados por largura, que é aplicado na interface de acionamento dos motores elétricos. A figura 3.41 ilustra o sinal de saída do controlador PID e o sinal PWM gerado para acionar o motor.

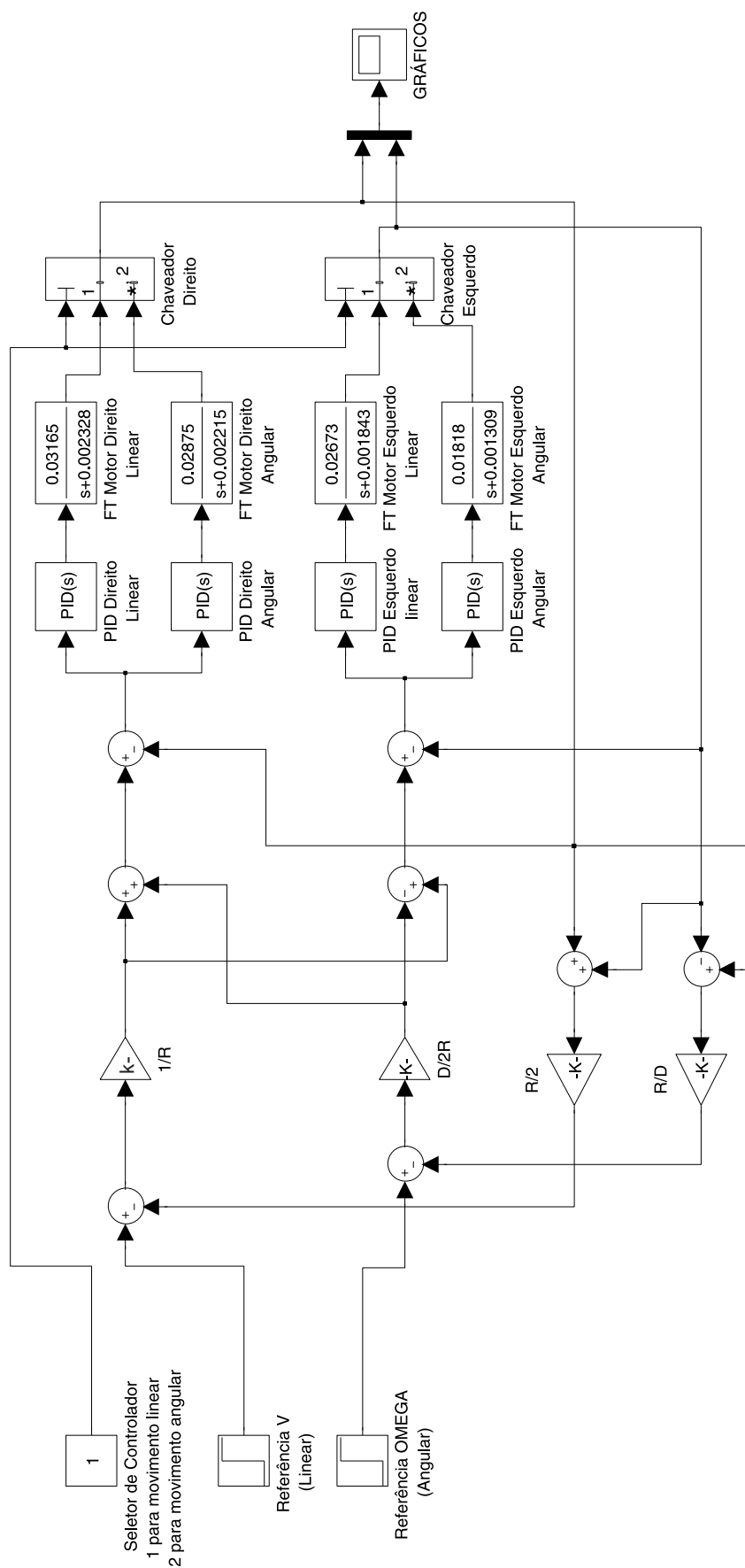
A realimentação dos controladores é realizada através da leitura de pulsos dos sensores *encoder*. Essa leitura é feita por meio do módulo *eQEP*, que é um módulo exclusivo para leituras de *encoder*, disponível no DSP utilizado. O módulo eQEP possibilita o cálculo de velocidade de rotação através da fórmula:

$$V(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta x}{T}, \quad (3.12)$$

onde  $V$  é a velocidade em rotações por segundo,  $\Delta x$  é a quantidade de pulsos medidos no intervalo de tempo  $T$ . O programa faz a contagem de pulso a cada intervalo de tempo  $T$ , que neste caso, é aproximadamente  $500\mu s$ . Após o cálculo de velocidade, é realizada uma filtragem para a retirada de ruídos. Essa filtragem é muito importante para o correto funcionamento dos controladores PID, pois no processo de identificação e determinação dos parâmetros, os dados utilizados também foram filtrados. Desta forma o dimensionamento dos filtros seguiu as mesmas especificações dos filtros usados anteriormente, ou seja, filtro passa-baixa *Butterworth* de 1º ordem, com frequência de amostragem de  $300KHz$  e frequência de corte de  $50Hz$ . O DSP foi programado para trabalhar em uma frequência de *Clock* de  $150MHz$  e a frequência de geração dos sinais PWM foi de  $30KHz$ .

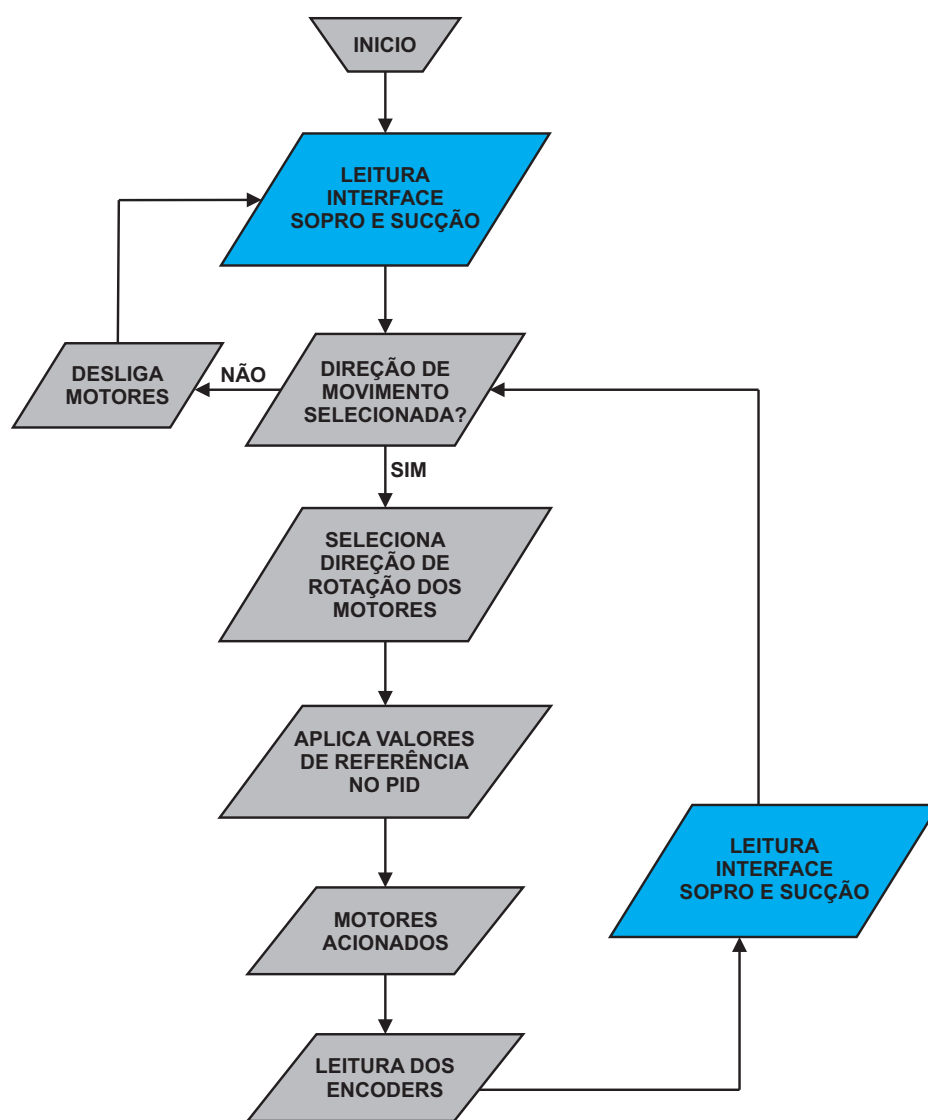
---

<sup>2</sup>Lógica descrita em código de linha de texto no Apêndice B.



**Figura 3.37:** Diagrama de blocos do controlador.

Fonte: Próprio autor.



**Figura 3.38:** Fluxograma de funcionamento geral do programa da cadeira de rodas.

Fonte: Próprio autor.



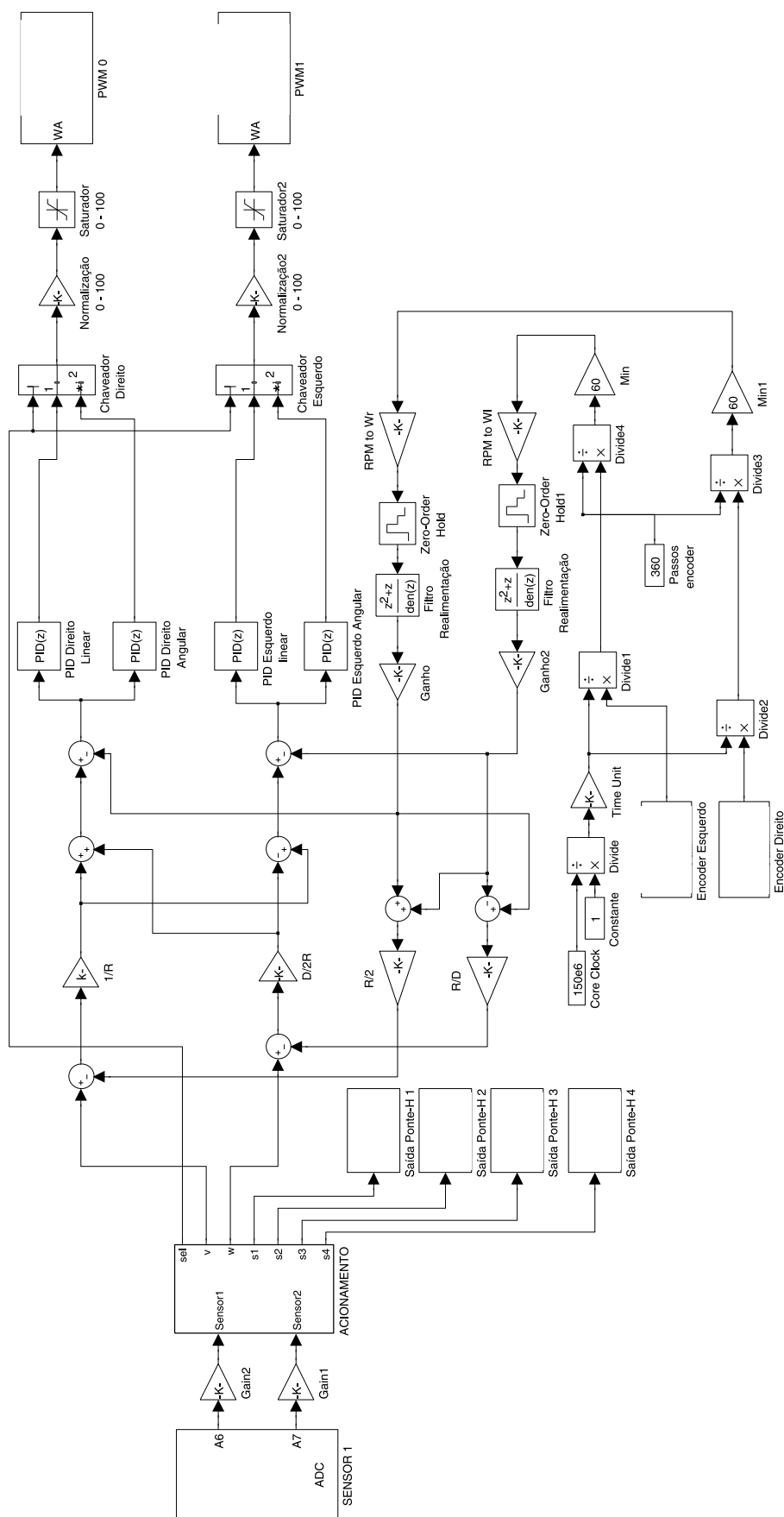
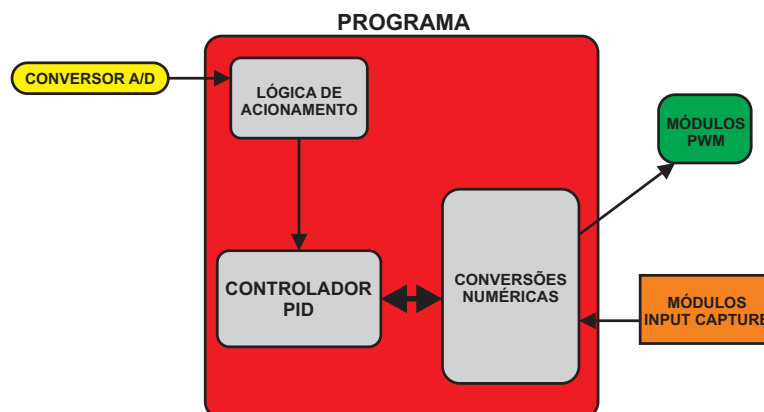
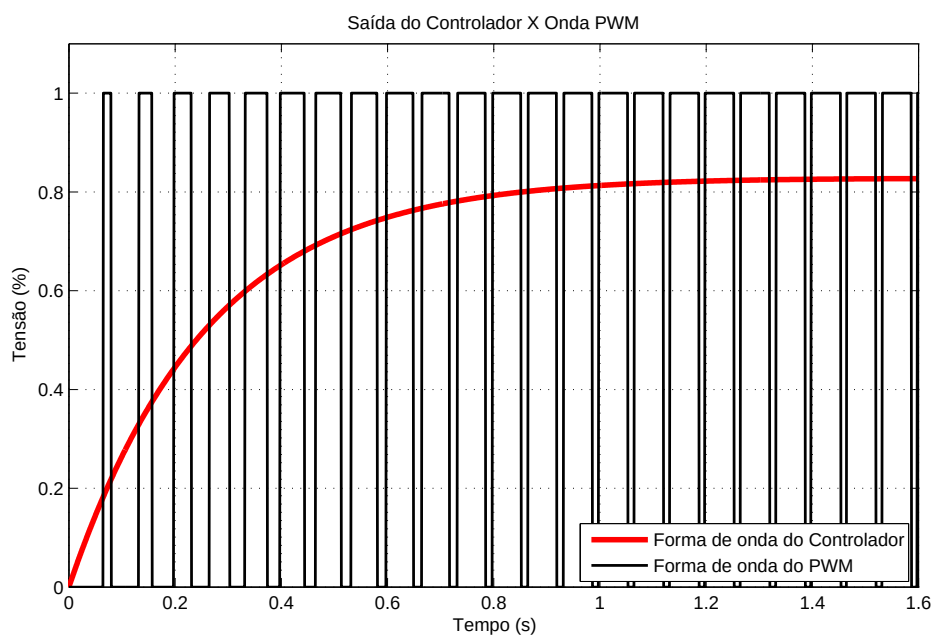


Figura 3.39: Diagrama de blocos do programa implementado.

Fonte: Próprio autor



**Figura 3.40:** Diagrama interno do programa de controle.  
Fonte: Próprio autor.



**Figura 3.41:** Sinal PWM gerado pelo controlador.  
Fonte: Próprio autor.

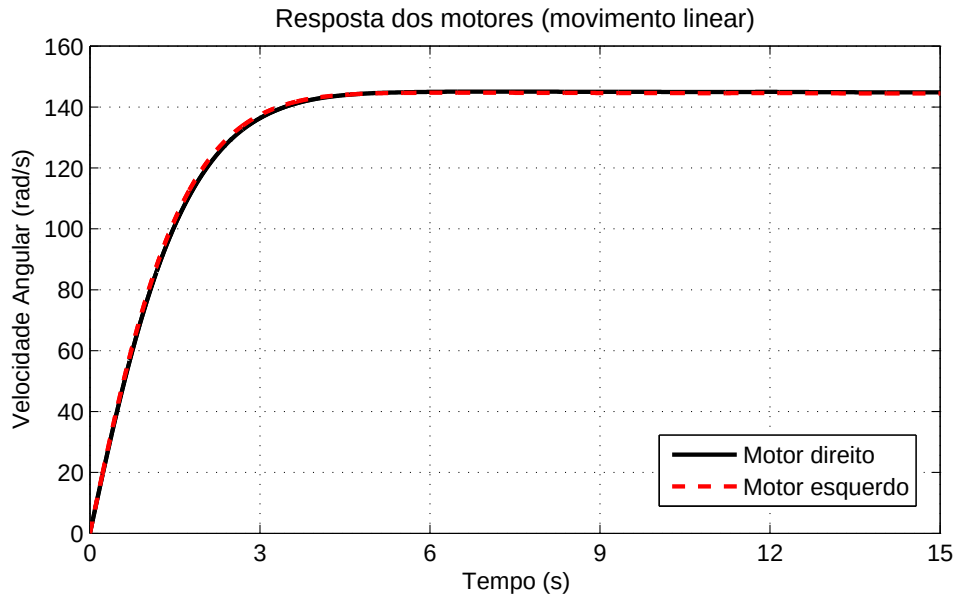
## 4 Resultados

Neste capítulo serão apresentados os resultados obtidos nas simulações do controlador PID e na implementação do sistema. As simulações foram realizadas no Simulink e a coleta de dados dos testes práticos foram realizadas com o LabView.

### 4.1 Simulações do Controlador PID

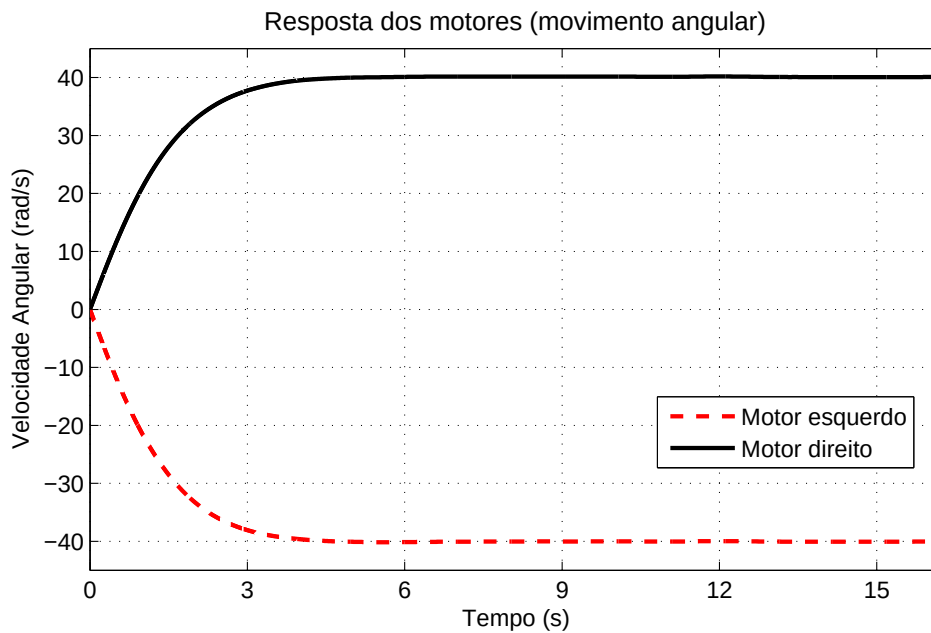
As simulações dos controladores foram realizadas no *software* Simulink conforme a figura 3.37. Primeiramente foi feita a identificação das funções de transferência dos motores, depois a determinação dos parâmetros dos controladores PID com o algoritmo de Teixeira, Assunção e Covacic (2007). Tendo como base esses dados, então foram realizadas as simulações. O programa de simulação foi feito utilizando as funções de transferência  $H_2(s) = \frac{0,03165}{s+0,002328}$  e  $H_4(s) = \frac{0,02673}{s+0,001843}$  encontradas para os motores direito e esquerdo, respectivamente, com carga. Também foi implementado outro programa utilizando  $H_5(s) = \frac{0,02875}{s+0,002215}$  e  $H_6(s) = \frac{0,01818}{s+0,001309}$  para os motores direito e esquerdo respectivamente, em giro com carga.

A figura 4.1 ilustra a simulação dos controladores PID para movimentos lineares, ou seja, para frente e para trás. É importante salientar que o dispositivo chaveador de controladores foi utilizado, sendo assim, para esse movimento a chaveador ficou na posição 1 e, quando foram necessários movimento angulares, o chaveador foi alterado para a posição 2.



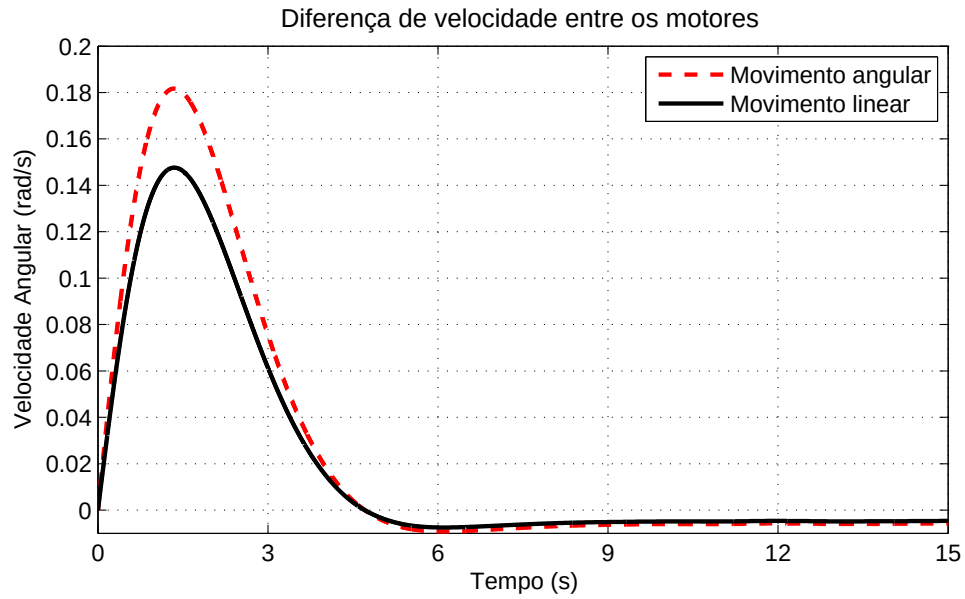
**Figura 4.1:** Resposta dos motores em movimento linear (simulação).  
Fonte: Próprio autor.

Em movimentos angulares, ou seja, giro para direita ou para a esquerda, a simulação foi a ilustrada na figura 4.2.



**Figura 4.2:** Resposta dos motores em movimento angular (simulação).  
Fonte: Próprio autor.

Tendo em vista que estas simulações são base para a implementação do sistema, as funções de transferência usadas foram as encontradas dos motores reais da cadeira de rodas. Então podem-se encontrar uma diferença de velocidade nos motores, que foi apresentada também na simulação. Essa diferença de velocidade também pode ser chamada de erro. A figura 4.3 mostra o erro de velocidade entre os motores direito e esquerdo em movimentos linear e angular.



**Figura 4.3:** Erro de velocidade entre os motores em movimento linear e angular (simulação).

Fonte: Próprio autor.

Analisando a figura 4.3, pode-se observar que o erro maior ocorre nos primeiros instantes de funcionamento e para movimentos angulares, ficando cada vez menor no decorrer do tempo. A diferença de velocidade entre as rodas causa alteração na trajetória da cadeira de rodas. Desta forma, com essa informação é possível concluir que na partida de movimento a cadeira terá um leve desvio de trajetória.

## 4.2 Resposta do Sistema Implemento no DSP

O controlador foi implementado com um período de amostragem de  $1ms$  e com os ganhos mostrados na tabela 4.1.

| Motor            | $K_d$ | $K_i$ | $K_p$ |
|------------------|-------|-------|-------|
| Direito (Carga)  | -0.25 | 0.03  | 5.75  |
| Direito (Giro)   | -0.25 | 0.03  | 5.3   |
| Esquerdo (Carga) | -0.25 | 0.03  | 7     |
| Esquerdo (Giro)  | -0.25 | 0.03  | 8.5   |

**Tabela 4.1:** Valores dos ganhos utilizados no controlador PID discreto.

Fonte: Próprio autor.

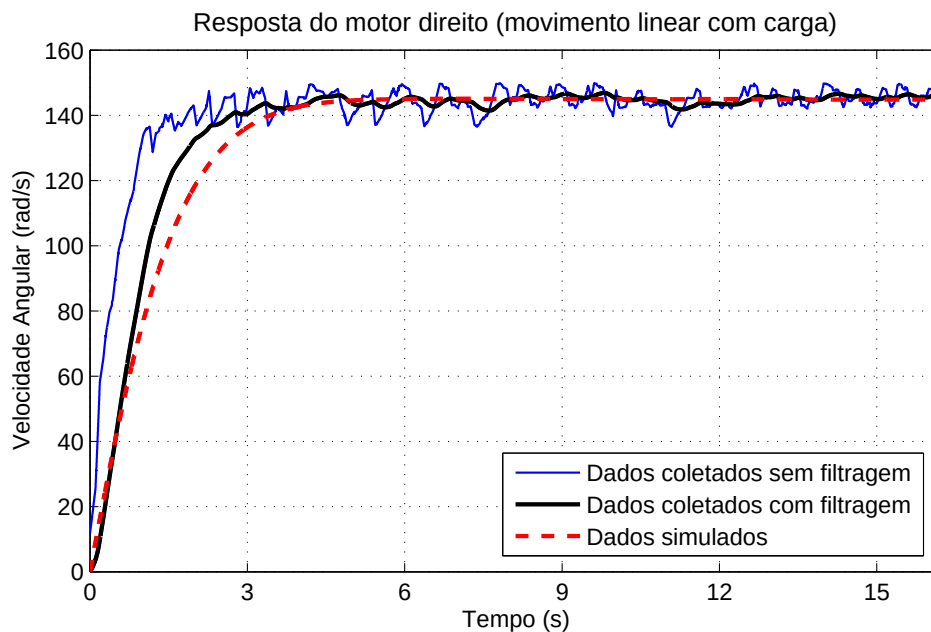
Depois do projeto ser implementado no DSP, o sistema foi colocado em funcionamento. Testes de condução da cadeira foram realizados com uma carga de 75Kg.

Foram realizados todos os movimentos com a cadeira de rodas, frente, trás,

direita e esquerda. Durante os movimentos, uma coleta de dados da velocidade angular dos motores foi realizada. Para essa coleta, utilizou-se uma placa DAQ portátil com conexão USB, juntamente com um computador portátil com o *software* LabView.

Desta forma, pode-se então visualizar os gráficos referentes a essa coleta de dados, onde são apresentados os sinais gerados pela simulação, dados coletados sem a filtragem e por fim os dados coletados com filtragem, podendo assim visualizar a resposta do sistema real implementado em comparação a simulação do sistema.

As respostas dos motores direito e esquerdo para movimento linear são mostradas nas figuras 4.4 e 4.5 respectivamente.



**Figura 4.4:** Resposta do motor direito para movimento linear.

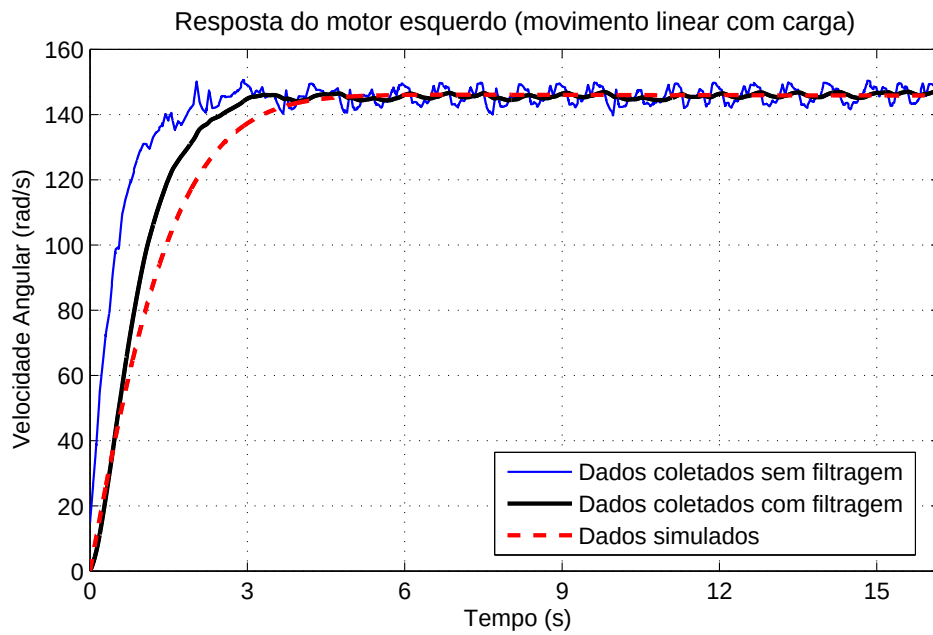
Fonte: Próprio autor.

Assim como na simulação, na implementação prática os motores também apresentaram uma leve diferença, ou erro, de velocidade entre os dois motores. Para movimento linear da cadeira, o sistema apresentou o erro mostrado na figura 4.6.

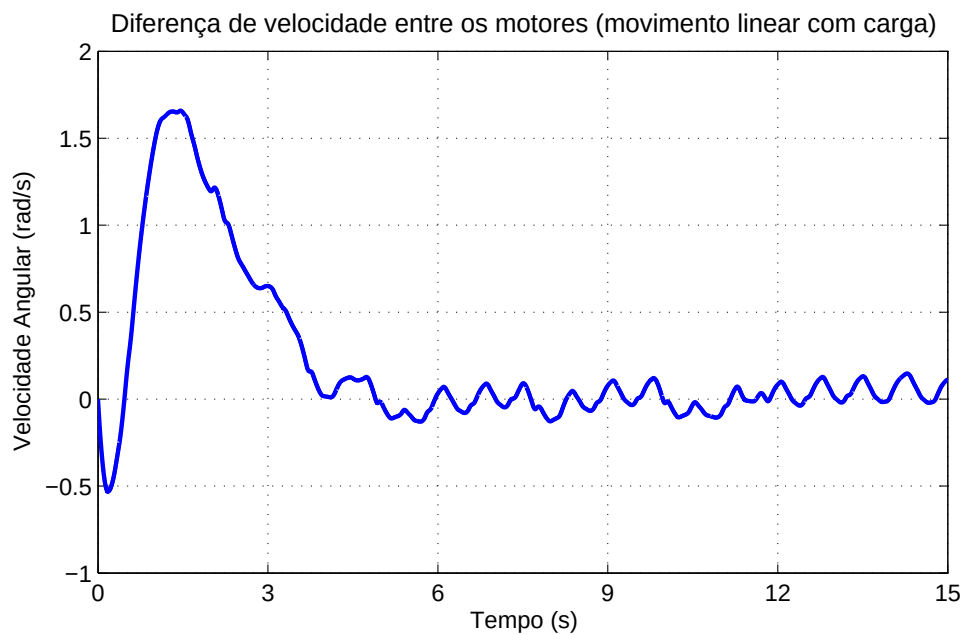
As respostas dos motores direito e esquerdo para movimento angular são mostradas nas figuras 4.7 e 4.8, respectivamente.

Da mesma forma do movimento linear, o erro apresentado no movimento angular é mostrado na figura 4.9.

O motivo pelo qual não se anulam as diferenças de velocidades entre as rodas (observadas nas figuras 4.6 e 4.9) , deve-se ao motivo de distúrbios elétricos,



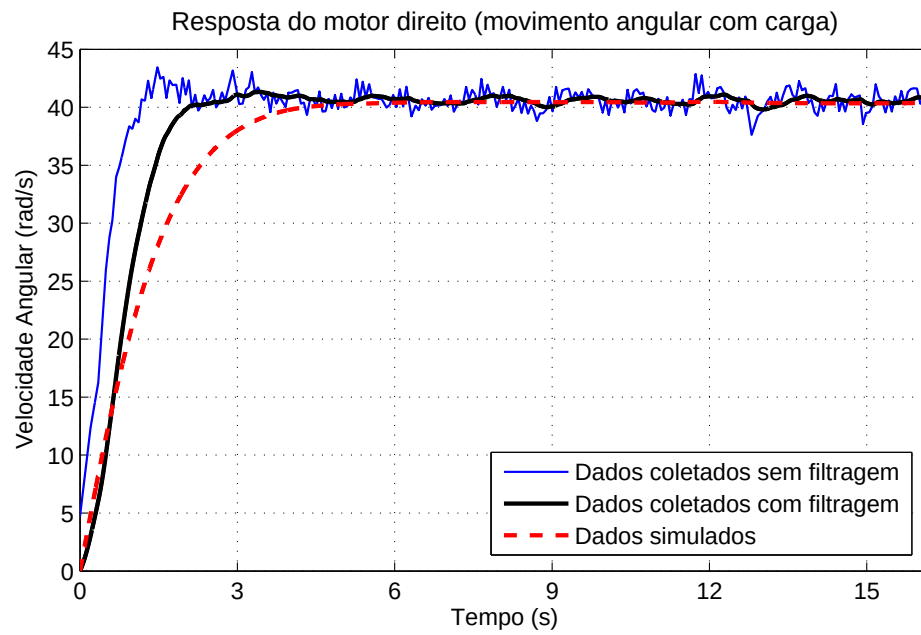
**Figura 4.5:** Resposta do motor esquerdo para movimento linear com carga.  
Fonte: Próprio autor.



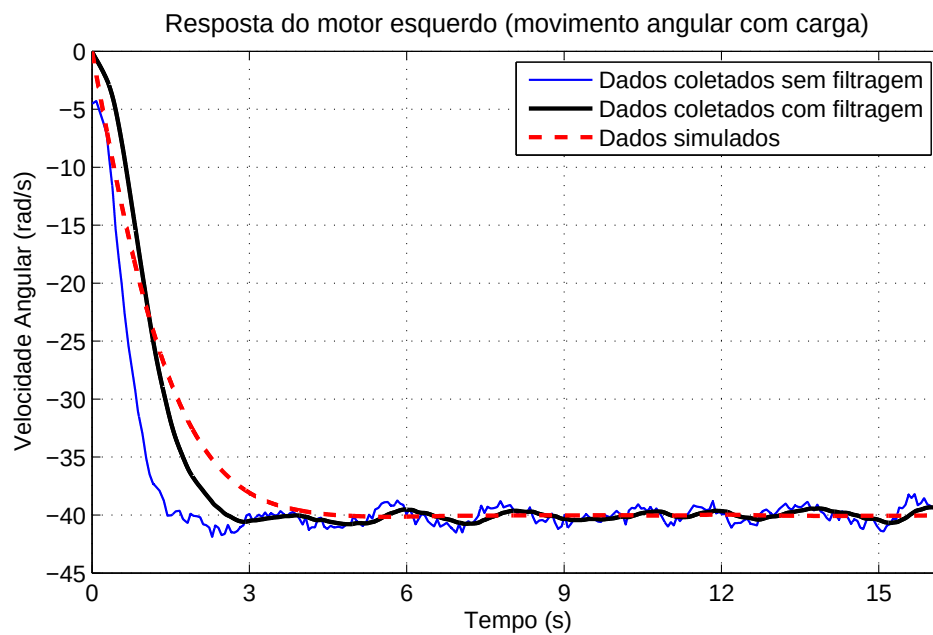
**Figura 4.6:** Erro de velocidade entre os motores em movimento linear com carga.  
Fonte: Próprio autor.

distúrbios mecânicos e efeito de escorregamento das rodas sobre o piso.

Analisando os gráficos de resposta do sistema pode-se observar, para todos os movimentos na situação prática, que as velocidades dos motores se mantiveram próximas às velocidades provenientes da simulação, demonstrando a aplicabilidade prática do projeto. A estabilização do sistema ocorreu como esperada, próximo à 3s. Então, como observado nos gráficos de erro, é nesse intervalo (até aproximadamente 3s) que ocorre a maior taxa de erro de velocidade entre os



**Figura 4.7:** Resposta do motor direito para movimento angular com carga.  
Fonte: Próprio autor.

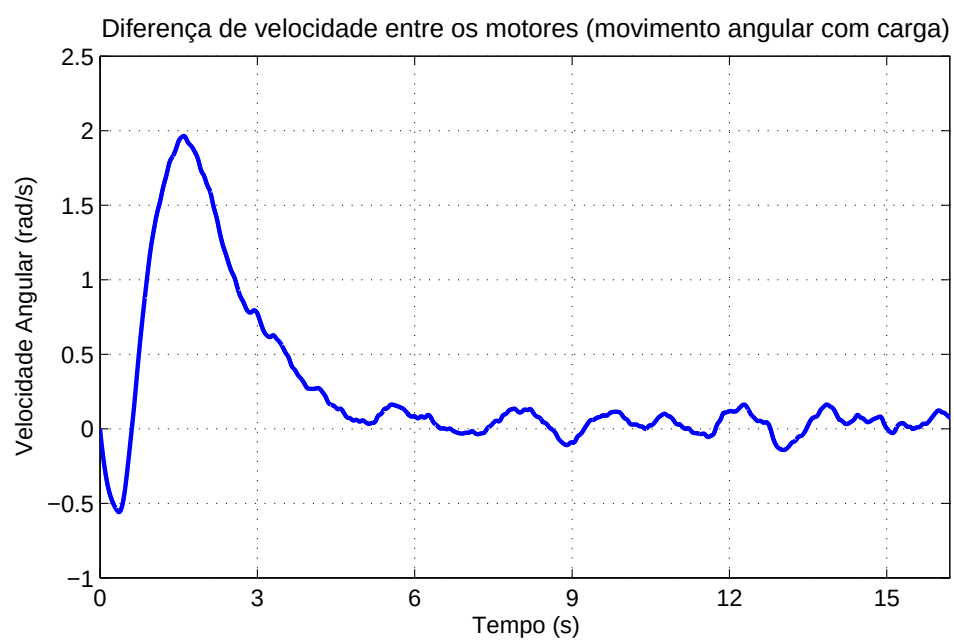


**Figura 4.8:** Resposta do motor esquerdo para movimento angular com carga.  
Fonte: Próprio autor.

motores. Após a estabilização do sistema (após 3s) a taxa de erro diminui e se mantém muito próxima a zero.

Desta forma, o sistema se mostrou eficiente em seu propósito, que é garantir a estabilidade de velocidade dos motores em todos os movimentos da cadeira de rodas.





**Figura 4.9:** Erro de velocidade entre os motores em movimento angular com carga.

Fonte: Próprio autor.

## 5 Conclusão

Trabalhos com temas relacionados à acessibilidade e inclusão de pessoas com deficiência física na sociedade estão em fase crescente no Brasil e no mundo. Pensando nisso, já no ano 2000, iniciaram-se os estudos sobre cadeira de rodas acionada por sopro e sucção na Universidade Estadual de Londrina. Vários trabalhos foram realizados por alunos e professores do Departamento de Engenharia Elétrica, resultando em uma embasamento técnico e científico que fez possível a realização desse trabalho.

Como vantagens desse trabalho, tem-se a simplicidade de seu sistema e os resultados satisfatórios, como por exemplo, a estabilidade na rotação dos motores e suavidade nos movimentos da cadeira de rodas. Dessa forma é possível incentivar a disseminação desse projeto na sociedade, tendo em vista a necessidade das pessoas, principalmente de classes sociais menos favorecidas, de locomover-se e ter mais liberdade para realizar as tarefas diárias e também serem incluídas na sociedade e mercado de trabalho.

O acionamento por sopros e sucções é uma interface de fácil uso para pessoas com deficiência motora, o que torna a cadeira de rodas muito prática para o uso. Entretanto, ainda é de grande necessidade que a cadeira de rodas possua um sistema eletrônico de controle. Então o controlador PID foi escolhido, pois embora não possa apresentar resultados ótimos, apresenta grande facilidade de implementação, o que o torna o controlador mais prático de se aplicar. Para gerenciar todas as tarefas do sistema, um DSP foi escolhido, pois é um sistema computacional rápido, robusto e portátil. O DSP da *Texas Instruments* foi utilizado por sua facilidade de trabalho e ótimas ferramentas de desenvolvimento, além de atender todas as necessidades do projeto.

Desta forma os objetivos do trabalho, sendo o projeto e implementação do controlador para a cadeira de rodas acionada por sopro e sucção, foram alcançados. Observando os resultados, nota-se que o sistema apresentou um funcionamento satisfatório com uma margem pequena de erro de velocidade, o que torna o sistema viável para a aplicação prática.

---

Entretanto, o projeto ainda pode ser melhorado. Outras funcionalidades podem ainda ser acrescentadas como por exemplo: sensores detectores de obstáculos, controle de posição por GPS, controle de trajetória com algoritmos de navegação autônoma, etc. Estes são temas abertos para trabalhos futuros.

# Referências

- AGUIRRE, L. *Enciclopédia de automática: Controle e automação volume II*. São Paulo: Blucher, 2007a. ISBN 9788521204091.
- AGUIRRE, L. *Enciclopédia de automática: Controle e automação volume III*. São Paulo: Blucher, 2007b. ISBN 9788521204091.
- AGUIRRE, L. A. *Introdução à identificação de sistemas - Técnicas lineares e não-lineares aplicadas a sistemas reais*. Uberlândia - MG: editora UFMG, 2004.
- BIAZETO, A. R. Contribuições de identificação de sistemas lineares e não lineares para o caso de paciente paraplégico. *TCC - Engenharia Elétrica - Universidade Estadual de Londrina*, 2011.
- BORGES, F. A. A. *Implementação de sistema de acionamento para plataformas robóticas móveis com ênfase em estimador de estado de carga de bateria*. Dissertação (Mestrado) — Engenharia Elétrica - Universidade Estadual de Londrina, Londrina - PR, 2014.
- BOYD, S.; GHAOUI, L. E.; FERON, E.; BALAKRISHNAN, V. *Linear Matrix Inequalities in System and Control Theory*. Philadelphia: SIAM, 1994.
- CARVALHO, A. S. *Modelagem de Colunas de Destilação Através de Modelos Auto-Regressivos*. Dissertação (Mestrado) — Universidade Estadual do Norte Fluminense, Campos dos Goytacazes - RJ, 2008.
- CHEN, R. X.; CHEN, L. G.; CHEN, L. System design consideration for digital wheelchair controller. *IEEE Transactions on Industrial Electronics*, v. 47, p. 898–908, 2000.
- CHO, S. Y.; VINOD, A. P.; CHENG, K. W. E. Towards a brain-computer interface based control for next generation electric wheelchairs. In: *3<sup>o</sup> Internacional Conference on Power Electronics Systems and Applications*. Hong Kong: , 2009. p. 1–5.
- CHONG, L.; HONG w. The design of wheelchair based on emg control. *Bioinformatics and Biomedical Engineering*, v. 25, p. 1721–1724, 2008.
- CHOW, J. W.; MILLIKAN, T. A.; CARLTON, L. g; CHAE, W. sik; LIM, Y. tae; MORSE marty i. Kinematic and electromyographic analysis of wheelchair propulsion on ramps of different slopes for young men with paraplegia. *Arch Phys Med Rehabil*, v. 90, p. 271–278, 2009.
- CHU, J. U.; MOON, I. H.; CHOI, G. W.; RYU, J. C.; MUN, M. S. Design of BLDC motor controller for electric power wheelchair. In: *Korea Orthopedics and Rehabilitation Engineering*. Korea: , 2004. p. 92–97.

- COELHO, A. A. R.; COELHO, L. dos S. *Identificação de sistemas dinâmicos lineares*. Florianópolis - SC: Didáctica, 2004. ISBN 9788532802804.
- DING, D.; COOPER, R. A. Electric powered wheelchairs. *IEEE Control Systems Magazine*, v. 80, p. 22–34, 2005.
- FARIA, P. M.; BRAGA, R. A. M.; VALGÔDE, E.; REIS, L. P. Interface framework to drive an intelligent wheelchair using facial expressions. *IEEE*, Vigo - Espanha, v. 34, p. 1971–1796, 2007.
- FERREIRA, C. L. L. *Interface de Sopro e Sucção para Controle de Cadeira de Rodas*. Dissertação (Mestrado) — Engenharia Elétrica - Universidade Estadual de Londrina, Londrina - PR, 2008.
- FITZGERALD, A. E.; KINGSLEY, C.; KUSKO, A. *Máquinas elétricas: conversão eletromecânica da energia, processos, dispositivos e sistemas*. São Paulo: McGraw-Hill, 1975.
- GAINO, R.; NOGUEIRA, R. G.; FRANCISCHI, W. E.; VIEIRA, D. P. Uso do labview para controle de um veículo. *Semina: Ciências Exatas e Tecnológica, Londrina*, v. 23(1), p. 47–54, 2002.
- GERMANOVIX, G. R.; CERVANTES, S. G. de S. Implementação do microcontrolador dsPIC33FJ128GP706 para gerência dos processos de uma cadeira de rodas comandada por sopro e sucção. In: *22º Encontro Anual de Iniciação Científica*. Foz do Iguaçu - PR: , 2013.
- GOODWIN, G. C.; GRAEBE, S. F.; SALGADO, M. E. *Control system design*. Nova Jersey - EUA: Prentice Hall Upper Saddle River, 2001.
- HASHIMOTO, M.; TAKAHASHI, K.; SHIMADA, M. Wheelchair control using an EOG- and EMG-based gesture interface. In: *International Conference on Advanced Intelligent Mechatronics Suntec Convention and Exhibition Center*. Singapura: , 2009. v. 1212-1217.
- HONEYWELL. *Honeywell Sensing and Control manual - AWM2100V*. 2005.
- IBGE. *Censo Demográfico 2010 - Características Gerais da População, Religião e Pessoas com Deficiência*. São Paulo, 2010.
- INSTRUMENTS, T. *TMS320C2000 Experimenter Kit Overview*. 2011.
- KIM, K.; SCHAEFER, R. C. Tuning a PID controller for a digital excitation control system. *IEEE Transactions on Industry Applications*, IEEE, v. 41, n. 2, p. 485–492, 2005.
- KOSOW, I. L. *Máquinas elétricas e transformadores*. São Paulo: Globo, 1985.
- LEISHMAN, F.; HORN, O.; BOURHIS, G. Smart wheelchair control throught a deictic approach. *Robotics and Autonomous Systems*, v. 44, p. 1149–1158, 2010.
- LIN, J.-S.; CHEN, K.-C.; YANG, W.-C. EEG and eye-blinking signals through a braincomputer interface based control for electric wheelchairs with wireless scheme. In: *New Trends in Information Science and Service Science (NISS) 4th International Conference*. Korea: , 2010. p. 731–734.

- LJUNG, L. *System identification: theory for the user*. Nova Jersey - EUA: Prentice-Hall, 1987. (Prentice-Hall information and system sciences series). ISBN 9780138816407.
- LJUNG, L. *System Identification Toolbox 7: User's Guide*. : MathWorks, Incorporated, 2007.
- LOPES, G. M. G. Implementação de sistema de controle analógico para cadeira de rodas elétrica utilizada por tetraplégicos com interface de comando por sopro e sucção. *TCC - Engenharia Elétrica - Universidade Estadual de Londrina*, 2012.
- MAZO, M.; RODRIGUEZ, F. J.; LÁZARO, J. L.; UREÑA, J.; GARCIA, J. C.; SANTISO, E.; REVENGA, P. A. Electronic control of a wheelchair guided by voice commands. *Control Engineering Practice*, v. 3, p. 665–674, 1995.
- MCDONALD, J. W.; SADOWSKY, C. Spinal-cord injury. *The Lancet*, v. 359, p. 417–425, 2002.
- MELO, L. F. de. *Proposta de Simulador Virtual para Sistema de Navegação de Robôs Móveis Utilizando Conceitos de Prototipagem Rápida*. Tese (Doutorado) — Universidade Estadual de Campinas UNICAMP, Campinas - SP, 2007.
- NAM, Y.; ZHAO, Q.; CICHOCKI, A. Tongue-rudder: A glossokinetic - potential-based tongue - machine interface. *IEEE Transactions on Biomedical Engineering*, v. 59, n 1, p. 290–299, 2012.
- NISE, N. S. *Control Systems Engineering*. Nova York: Wiley, 2010. 926 pages p.
- OGATA, K. *Engenharia de Controle Moderno*. São Paulo: Pearson Prentice Hall, 2010.
- ONISHI, M.; KOBAYASHI, K.; WATANABE, K. A study of intuitive electric wheelchair control interface based on weight shift. In: *SICE Annual Conference*. Japão: , 2008. p. 2710–2713.
- RODRIGUES, G. G. *Identificação de Sistemas Dinâmicos Não-Lineares Utilizando Modelos NARMAX Polinomiais- Aplicação a Sistemas Reais*. Tese (Doutorado) — Universidade Federal de Minas Gerais, Uberlândia - MG, 1996.
- ROSSINI, F. L. *Projeto de Controlador Robusto à Cadeira de Rodas Móveis Via Abordagem por LMIs*. Dissertação (Mestrado) — Universidade Estadual de Londrina, Londrina - PR, 2013.
- SANTOS, N. M. dos; ROSSINI, F. L.; COVACIC, M. R.; GAINO, R. Controle robusto de veículos sobre plataforma com rodas e tração diferencial utilizando LMIs. In: *Simpósio Brasileiro de Automação Inteligente*. Fortaleza - CE: , 2013. p. 1–6.
- SATO, N.; YODA, I.; INOUE, T. Shoulder gesture interface for operating electric wheelchair. In: *IEEE 12th International Conference on Computer Vision Workshops*. Kyoto - Japão: , 2009. p. 2048–2055.
- SHIM, H. M.; HONG, J. P.; CHUNG, S. B.; HONG, S. H. A powered wheelchair controller based on master-slave control architecture. In: *IEEE International Symposium on Industrial Electronics*. South Korea: , 2001. v. 3, p. 1553–1556.

SILVA, N. da. *Contribuições ao Estudo, Projeto e Aplicação de Filtros Ativos Híbridos de Potência*. Tese (Doutorado) — Universidade Estadual de Campinas, Campinas - SP, 2012a.

SILVA, R. C. de M. Técnicas de identificação e controle analógico e digital com labview e um kit DSP da texas instruments. *TCC - Engenharia Elétrica - Universidade Estadual de Londrina*, 2012b.

SOBRINHO, A. S. F.; FELIZARDO, K. R.; SILVA, M. A. da; OLIVEIRA, H. P.; LONE, L. P.; GERMANOVIX, W.; GAINO, R. Cadeira de rodas controlada por sopros e sucções. *Revista Semina Ciências Exatas e Tecnológicas*, v. 21, n 4, p. 3–7, 2000.

SOBRINHO, A. S. F.; SUZUKI, I.; GERMANOVIX, W.; GAINO, R. Implementação e análise do controle de uma cadeira de rodas através de sopros e sucções. In: *VI Simpósio Brasileiro de Automação Inteligente*. Bauru - SP: , 2003. p. 1–4.

SODERSTROM, T.; STOICA, P. *System Identification*. Nova York: Prentice Hall, 1989. (Prentice Hall International Series In Systems And Control Engineering, v. 4). ISBN 9780138812362.

SOUZA, K. M. de. *Identificação e Controle em Robôs Móveis Utilizando Kit Robótico Lego Mindstorms*. Dissertação (Mestrado) — Engenharia Elétrica - Universidade Estadual de Londrina, Londrina - PR, 2014.

TEIXEIRA, M. C. M.; ASSUNÇÃO, E.; COVACIC, M. Proportional controllers: Direct method for stability analysis and MATLAB implementation. *IEEE Transactions on Education*, v. 50, n. 1, p. 74–78, 2007.

TIAN, Z.; XU, W. Electric wheelchair controller based on parameter self adjusting fuzzy PID. In: *Computational Intelligence and Natural Computing*. China: , 2009. p. 358–361.

WANG, H.; SALATIN, B.; GRINDLE, G. G.; DING, D.; COOPER, R. A. Real-time model based electrical powered wheelchair control. *National Institutes of Health*, v. 26, p. 1–33, 2010.

WIDYOTRIATMO, A.; RAUZANFIQR, S. K.; SUPRIJANTO. A modified PID algorithm for dynamic control of an automatic wheelchair. In: *IEEE Conference on Control System and Industrial Informatics*. Indonésia: , 2012. p. 64–68.

YAMAMOTO, M.; IKEDA, T.; SASAKI, Y. Real-time analog input device using breath pressure for the operation of powered wheelchair. In: *IEEE International Conference on Robotics and Automation*. Califórnia - EUA: , 2008. p. 1–6.

## Apêndice A – Disseminações

### A.1 Patente Requerida

O projeto descrito nessa dissertação possui patente requerida. A patente possui número de solicitação **BR 10 2014 018375 2** e resumo: “A presente invenção consiste de um sistema micro controlado que permite a movimentação de uma cadeira de rodas através de sopros e sucções realizadas através de um bocal, visando uma melhor qualidade de vida para pacientes. Trata-se de um dispositivo que funciona através de interface com o usuário e controle dos motores, e poderá ser instalado em qualquer cadeira de rodas movida por motores elétricos. Para dirigir a cadeira o usuário usará apenas comandos de sopros e/ou sucções em um bocal plástico acoplado a um suporte próximo à boca.”

A patente está sob a responsabilidade da Aintec/UEL, os direitos autorais são cedidos à Universidade Estadual de Londrina e os inventores são:

- *Prof. Dr. Ruberlei Gaino*
- *Prof. Dr. Walter Germanovix*
- *Prof. Dr. Márcio Roberto Covacic*
- *Profa. Dra. Silvia Galvão de Souza Cervantes*
- *Eng. Edno Gentilho Junior*

### A.2 Publicações Diretamente Relacionadas ao Tema

1. **GENTILHO JR, E.**; ROSA FILHO, J. E. A.; BIAZETO, A. R.; COVACIC, M. R.; CARVALHO, A.A.; TEIXEIRA, M.C.M.; SANCHES, M.A.A.; KOZAN, R.F.; JUNQUEIRA, M.V.N.; GAINO, R., *Controle Implementado em DSP para cadeira de rodas acionada por sopro e sucção*. In: Anais do



11º Simpósio Brasileiro de Automação Inteligente (SBAI- DINCON 2013), Fortaleza-CE, 6 páginas.

2. ROSA FILHO, J. E. A.; **GENTILHO JR, E.**; COVACIC, M. R.; GAINO, R., *Técnica de Controle Otimizado Aplicada a uma Cadeira de Rodas Controlada por Sopro e Sucção*. In: Anais do 11º Simpósio Brasileiro de Automação Inteligente (SBAI- DINCON 2013), Fortaleza-CE, 6 páginas.

3. **GENTILHO JR, E.**; BIAZETO, A. R.; KAYAMORI, H. K.; COVACIC, M. R.; GAINO, R.. *Controle Implementado em DSP para Cadeira de Rodas Acionada por Sopro e Sucção*. In: Vicente de Lima Gongora; Daniela Resende Faria; Adriana Giseli Leite. (Org.). Inovação e Tecnologia. 1ed. Londrina: Faculdade de Tecnologia SENAI Londrina, 2014, v. 1, p. 236-248

### A.3 Publicações Relacionadas com a Área de Pesquisa

1. SOUZA, K. M.; **GENTILHO JR, E.**; GAINO, R.; COVACIC, M. R. *Identificação e Controle em Robôs Móveis Utilizando Kit Robótico Lego Mindstorms*. In: Anais do 11º Simpósio Brasileiro de Automação Inteligente (SBAI- DINCON 2013), Fortaleza-CE, 5 páginas.

2. SANCHES, M. A. A.; KOZAN, R. F.; JUNQUEIRA, M. V. N.; URBAN, M. F. R.; COSTA, G. Z.; PONTES, W.; MACHADO, E. R. M.; TEIXEIRA, M. C. M.; CARVALHO, A. C.; GAINO, R.; **GENTILHO JR, E.**, *Metodologia para Análise e Experimentação de Controladores e Plantas Utilizando DSP's em Sistemas com malha fechada*. In: Anais do 11º Simpósio Brasileiro de Automação Inteligente (SBAI- DINCON 2013), Fortaleza-CE, 5 páginas.

3. BIAZETO, A. R.; **GENTILHO JR, E.**; COVACIC, M. R.; GNANN, B. A.; GAINO, R.. *Controle Paralelo Distribuído Com Rastreamento Aplicado Ao Modelo Não Linear Do Movimento Do Complexo Canela-Pé Do Paciente Paraplégico*. In: Vicente de Lima Gongora; Daniela Resende Faria; Adriana Giseli Leite. (Org.). Profissionais da Indústria. 1ed. Londrina: Faculdade de Tecnologia SENAI Londrina, 2014, v. 1, p. 116-128.

4. SOUZA, K. M.; COVACIC, M. R.; GAINO, R.; **GENTILHO JR, E.**; *Identificação e Controle Discreto Utilizado em Robôs Móveis*. Submetido à Revista Semina: Ciências Exatas e Tecnológicas (2014).

5.SANCHES, M. A. A.; GAINO, R.; KOZAN, R. F.; TEIXEIRA, M. C. M.; CARVALHO, A. A.; COVACIC, M. R.; ALVES, C. A.; URBAN, M. F. R.; JUNQUEIRA, M. V. N.; CARDIM, R.; ASSUNCAO, E. A.; **GENTILHO JR, E.** *Digital Controller Design Considering Hardware Constraints: Application in a Paraplegic Patient*. Revista Brasileira de Engenharia Biomédica (Impresso), v. 30, p. 232-241, 2014.

Este artigo é resultado de 15 anos de pesquisa nos laboratórios da FEIS-UNESP. A tese de doutorado de Ruberlei Gaino em 2009, a dissertação de mestrado de Renan Fernandes Kozan e a tese de doutorado de Marcelo Augusto Assunção Sanches, viabilizando este importante artigo. Na última fase de compilação do artigo o aluno Edno Gentilho Junior, em visita aos laboratórios dos Prof. Dr. Aparecido Augusto de Carvalho em 2013, foi inserido para auxiliar nos ajustes de programação com linguagem C e uso do *software* Proteus.

Todo o processo de aprendizagem sobre o assunto de Controle de Paciente Paraplégico utilizando modelos Fuzzy T-S e neuroestimulador foi concebido nos laboratórios da FEIS-UNESP com autorização do Prof. Dr. Aparecido Augusto de Carvalho, no qual a continuidade de qualquer forma de pesquisa deve ser consultado junto com os autores Ruberlei Gaino, Marcelo Augusto Assunção Sanches e Renan Fernandes Kozan.

## A.4 Projetos de Pesquisa Envolvidos

Este trabalho foi parte realizada do projeto de pesquisa cadastrado e financiado pela Fundação Araucária do Paraná, para a modernização de uma cadeira de rodas elétrica. Os projetos envolvidos são os citados abaixo:

1.FUNDAÇÃO ARAUCÁRIA - CONV. 200/2010 - CHAMADA N° 08/2009  
- protocolo - 19530 - verba total, R\$ 41.210,00.

2.FUNDAÇÃO ARAUCÁRIA - CONV. 459/2010 - CHAMADA N° 14/2009  
- protocolo - 18332 - verba total R\$ 16.910,00.

Participantes: *Prof. Dr. Ruberlei Gaino, Prof. Dr. Leonimer Flávio de Melo, Prof. Dr. Walter Germanovix, Prof. Dr. Márcio Roberto Covacic e Profa. Dra. Silvia Galvão de Souza Cervantes.*

## Apêndice B – Código do Bloco “ACIONAMENTO” (MatLab) do Programa Implementado no DSP

```
%***** EDNO GENTILHO JUNIOR *****
%***** (edno.gentilho@hotmail.com) *****
%***** 10/01/14 *****
%      Código de acionamento da cadeira de rodas      *
%      utilizando 2 portas AD                          *
%*****setando entradas e saidas:*****
% s1,s2,s3,s4: saidas booleanas para controle da ponte-H dupla
% Sensor1,Sensor2: entra da sinal dos sensores (0 a 3V)
% sel: seletor de controlador
function [sel,v,w,s1,s2,s3,s4] = fcn(Sensor1,Sensor2)
%estado das variáveis caso AD não se encaixe em nenhuma opção abaixo
s1=false;
s2=false;
s3=false;
s4=false;
v=0;
w=0;
sel=1;
if Sensor1<1 %rotina para movimento para tras
    s1=false;
    s2=true;
    s3=true;
    s4=false;
    v=77;
    w=0;
    sel=1;
end
end
```

```
if Sensor1>2 %rotina para movimento frente
    s1=true;
    s2=false;
    s3=false;
    s4=true;
    v=77;
    w=0;
    sel=1;
end
```

```
if Sensor2<1 %rotina para movimento esquerda
    s1=true;
    s2=false;
    s3=true;
    s4=false;
    v=0;
    w=77;
    sel=2;
end
```

```
if Sensor2>2 %rotina para movimento direita
    s1=false;
    s2=true;
    s3=false;
    s4=true;
    v=0;
    w=77;
    sel=2;
end
```

## Apêndice C – Programa de Controle Desenvolvido em Linguagem C

```

/* Autor: Edno Gentilho Junior
/* edno.gentilho@hotmail.com

#include "flash.h"
#include "flash_private.h"
BlockIO_implement_PID_duplo_0_T flash_B;
D_Work_implement_PID_duplo_00_T f_DWork;
RT_MODEL_implement_PID_duplo__T flas_M_;
RT_MODEL_implement_PID_duplo__T *const flas_M = & flas_M_;
void flash_step(void)
{
    real_T saturador;
    real_T sinal1;
    real_T ganho2;
    real_T ganho1;
    real_T v;
    real_T w;
    real_T soma;
    real_T filt_m;
    real_T filt_k;
    {
        AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;
        AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;
        AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;
        while (AdcRegs.ADCST.bit.INT_SEQ1 == 0)
    {}
    asm(" RPT #11 || NOP");

```

```
    out1 = (AdcRegs.ADCRESULT0)>> 4;
    out2 = (AdcRegs.ADCRESULT1)>> 4;
    out3 = (AdcRegs.ADCRESULT2)>> 4;
}

ganho2 = ganho02 *out2;
ganho1 = ganho01 *out3;

if (ganho2 > 2.0) {
    s1 = TRUE;
    s2 = FALSE;
    s3 = FALSE;
    s4 = TRUE;
    v = 60.0;
w = 0;
}
if (ganho1< 1.0) {
    s1 = TRUE;
    s2 = FALSE;
    s3 = TRUE;
    s4 = FALSE;
    v = 0;
w = 60;
}
if (ganho1 > 2.0) {
    s1 = FALSE;
    s2 = TRUE;
    s3 = FALSE;
    s4 = TRUE;
    v = 0;
w = 60;
}
{
    GpioDataRegs.GPBSET.bit.GPIO34 = (s1 != 0);
    GpioDataRegs.GPBCLEAR.bit.GPIO34 = !(s1 != 0);
}

{
```

```

    GpioDataRegs.GPBSET.bit.GPIO32 = (s2 != 0);
    GpioDataRegs.GPBCLEAR.bit.GPIO32 = !(s2 != 0);
}

{
    GpioDataRegs.GPASET.bit.GPIO30 = (s3 != 0);
    GpioDataRegs.GPACLEAR.bit.GPIO30 = !(s3 != 0);
}

{
    GpioDataRegs.GPASET.bit.GPIO28 = (s4 != 0);
    GpioDataRegs.GPACLEAR.bit.GPIO28 = !(s4 != 0);
}

ganho1 = 1.0 / clock * constante * time;

{
    if (EQep2Regs.QFLG.bit.UTO == 1)
{
        EncoderDireito = EQep2Regs.QPOSLAT;
//EQep2Regs.QCLR.bit.UTO == 1;
}

    saturador = 1.0 / ganho1 * EncoderDireito;

    ganho2 = (pooled4[0] * saturador -pooled3[0] *
    FILT_STATE[0]) - pooled3[1] * FILT_STATE[1];

    soma= (pooled2[0] * ganho2+ pooled2[1] * FILT_STATE[0])
        + pooled2[2] * FILT_STATE[1];

    FILT_STATE[1] = FILT_STATE[0];
    FILT_STATE[0] = ganho2;

    GeneratedFilterBlock = pooled4[1] * soma;

    saturador = 1.0 / Passos_Value * GeneratedFilterBlock *

```

```

Min1_Gain * RPMtoWr_Gain;

{
    if (EQep1Regs.QFLG.bit.UTO == 1)
        EncoderEsquerdo = EQep1
Regs.QPOSLAT;// RegisterEQep1Regs.QCLR.bit.UTO == 1;
}

ganho1 = 1.0 / ganho1 * EncoderEsquerdo;

ganho2 = (pooled4[0] * ganho1 - ipooled3[0] * FILT_STA_o[0])
- pooled3[1] * FILT_STA_o[1];

filt_k = (pooled2[0] *ganho2 + pooled2[1] * FILT_STA_o[0])
+ pooled2[2] * FILT_STA_o[1];

FILT_STA_o[1] = FILT_STA_o[0];
FILT_STA_o[0] = ganho2;
GeneratedFilterBlock = pooled4[1] * filt_k;

ganho1 = 1.0 / Passos_Value * GeneratedFilterBlock
* Min_Gain * RPMtoWl_Gain;
ganho2 = (v - (saturador + ganho1) * R2_Gain) * R_Gain;
sinal1 = (0.0 - (saturador - ganho1) * RD_Gain) * D2R_Gain;

soma = (ganho2 + sinal1) - saturador;

saturador = Filter_DSTATE;

w = (DerivativeGain_Gain * soma - saturador) * FilterCoefficient_Gain;

saturador = Filter_DSTATE_p;

filt_m = (DerivativeGain_Gain_c * soma - saturador)
* FilterCoefficient_Gain_g;

```



```

    saturador = (ProportionalGain_Gain_a * soma + Integrator_DSTATE)+ w;

    saturador *= Normalizao018001_Gain;

    saturador = saturador >= Saturation_UpperSat ? Saturation_UpperSat :
saturador <= Saturation_LowerSat ? Saturation_LowerSat : saturador;

    {
        EPwm3Regs.CMPA.half.CMPA = (uint16_T)((uint32_T)EPwm3Regs.TBPRD
* saturador / 100);
    }

    ganho1 = (ganho2- sinal1) - ganho1;

    sinal1 = implement_PID_duplo_004_f_DWork.Filter_DSTATE_d;

    v = (DerivativeGain_Gain_cz * ganho1 -sinal1) * FilterCoefficient_Gain_f;

    sinal1 = iFilter_DSTATE_do;

    filt_k = (DerivativeGain_Gain_b * ganho1 - sinal1)
* FilterCoefficient_Gain_j;

    sinal1 = (ProportionalGain_Gain_m *ganho1 + Integrator_DSTATE_g) + v;

    sinal1 = Normalizao2018001_Gain;

    sinal1 = sinal1 >= Saturation1_UpperSat ?
    Saturation1_UpperSat : sinal1 <= Saturation1_LowerSat ?

```

```

    Saturation1_LowerSat : sinal1;

    {
        EPwm2Regs.CMPA.half.CMPA = (uint16_T)((uint32_T)EPwm2Regs.TBPRD
* sinal1 / 100);
    }

    sinal1 = Amplitude * sin (FreqNorm);

    FreqNorm = Frequency * 0.0062831853071795866 + FreqNorm;

    if (FreqNorm >= 6.2831853071795862)
    {
        FreqNorm = FreqNorm - 6.2831853071795862;
    } else
    {
        if (FreqNorm < 0.0) {
            FreqNorm = FreqNorm + 6.2831853071795862;
        }
    }

    Integrator_DSTATE = IntegralGain_Gain_b * soma *
Integrator_gainval + Integrator_DSTATE;

    Filter_DSTATE = Filter_gainval * w + Filter_DSTATE;

    Integrator_DSTATE_i = IntegralGain_Gain * soma *
Integrator_gainval_m + Integrator_DSTATE_i;

    Filter_DSTATE_p = Filter_gainval_p * FilterCoefficient_m +
Filter_DSTATE_p;

    Integrator_DSTATE_g = IntegralGain_Gain_c * ganho1 *
Integrator_gainval_b + Integrator_DSTATE_g;

    Filter_DSTATE_d = Filter_gainval_k * v + Filter_DSTATE_d;

    Integrator_DSTATE_o = IntegralGain_Gain_d * ganho1 *

```

```
Integrator_gainval_e + Integrator_DSTATE_o;

    Filter_DSTATE_do = Filter_gainval_kg * FilterCoefficient_k
+ Filter_DSTATE_do;
}

void flash_initialize(void)
{

    (void) memset(((void *) &flash_B), 0,
                  sizeof(BlockIO_implement_PID_duplo_0_T));
    {
        out1 = 0.0;
        out2 = 0.0;
        out3 = 0.0;
        EncoderDireito = 0.0;
        EncoderEsquerdo = 0.0;
        GeneratedFilterBlock = 0.0;
    }
    /* states (dwork) */
    GeneratedFilterBlock_FILT_STATE[0] = 0.0;
    GeneratedFilterBlock_FILT_STATE[1] = 0.0;
    GeneratedFilterBlock_FILT_STA_o[0] = 0.0;
    GeneratedFilterBlock_FILT_STA_o[1] = 0.0;
    Integrator_DSTATE = 0.0;
    Filter_DSTATE = 0.0;
    Integrator_DSTATE_i = 0.0;
    Filter_DSTATE_p = 0.0;
    Integrator_DSTATE_g = 0.0;
    Filter_DSTATE_d = 0.0;
    iIntegrator_DSTATE_o = 0.0;
    Filter_DSTATE_do = 0.0;
    FreqNorm = 0.0;
    {
        real_T arg;

        InitAdc();
        config_ADC_A (2U, 528U, 0U, 0U, 0U);
```

```
EALLOW;
GpioCtrlRegs.GPBPUD.all = 0U;
GpioCtrlRegs.GPBMUX1.all &= 4294967247U;
GpioCtrlRegs.GPBDIR.all |= 4U;
EDIS;
```

```
EALLOW;
GpioCtrlRegs.GPBPUD.all = 0U;
GpioCtrlRegs.GPBMUX1.all &= 4294967292U;
GpioCtrlRegs.GPBDIR.all |= 1U;
EDIS;
```

```
EALLOW;
GpioCtrlRegs.GPAPUD.all = 0U;
GpioCtrlRegs.GPAMUX2.all &= 3489660927U;
GpioCtrlRegs.GPADIR.all |= 1073741824U;
EDIS;
```

```
EALLOW;
GpioCtrlRegs.GPAPUD.all = 0U;
GpioCtrlRegs.GPAMUX2.all &= 4244635647U;
GpioCtrlRegs.GPADIR.all |= 268435456U;
EDIS;
```

```
config_QEP_eQEP2(65000U, 0, 65000U, 0, 0, 34816, 12302, 32768,
32768, 0);
```

```
config_QEP_eQEP1(65000U, 0, 65000U, 0, 0, 34816, 12302, 32768,
32768, 0);
```

```
{
    EPWMPARAMS EPwm3Params;

    EPwm3Params.TBPRD = 18;
    EPwm3Params.TBCTL_CTRMODE = 2;
    EPwm3Params.TBCTL_SYNCSEL = 1;
    EPwm3Params.TBCTL_PHSEN = 0;
```

```
EPwm3Params.TBCTL_PHSDIR = 0;
EPwm3Params.TBPHS = 0;
EPwm3Params.TBCTL_HSPCLKDIV = 5;
EPwm3Params.TBCTL_CLKDIV = 4;

EPwm3Params.CMPCTL_LOADAMODE = 0;
EPwm3Params.CMPCTL_LOADBMODE = 0;
EPwm3Params.CMPA = 0;
EPwm3Params.CMPB = 0;

EPwm3Params.AQCTLA = 144;
EPwm3Params.AQCTLB = 144;
EPwm3Params.AQCSFRC_CSFA = 0;
EPwm3Params.AQCSFRC_CSFB = 0;
EPwm3Params.AQCSFRC_RLDCSF = 0;

EPwm3Params.DBCTL_OUT_MODE = 0;
EPwm3Params.DBCTL_IN_MODE = 0;
EPwm3Params.DBCTL_POLSEL = 0;
EPwm3Params.DBRED = 0.0;
EPwm3Params.DBFED = 0.0;

EPwm3Params.ETSEL_SOCAEN = 0;
EPwm3Params.ETSEL_SOCASEL = 1;
EPwm3Params.ETPS_SOCAPRD = 1;
EPwm3Params.ETSEL_SOCBEN = 0;
EPwm3Params.ETSEL_SOCBSEL = 1;
EPwm3Params.ETPS_SOCBPRD = 1;
EPwm3Params.ETSEL_INTEN = 0;
EPwm3Params.ETSEL_INTSEL = 1;
EPwm3Params.ETPS_INTPRD = 1;

EPwm3Params.PCCTL_CHPEN = 0;
EPwm3Params.PCCTL_CHPFREQ = 0;
EPwm3Params.PCCTL_OSHTWTH = 0;
EPwm3Params.PCCTL_CHPDUTY = 0;

EPwm3Params.TZSEL = 0;
```

```
EPwm3Params.TZCTL_TZA = 3;
EPwm3Params.TZCTL_TZB = 3;
EPwm3Params.TZEINT_OST = 0;
EPwm3Params.TZEINT_CBC = 0;

config_ePWMRegs(&EPwm3Regs, &EPwm3Params);
}

{
    EPWMPARAMS EPwm2Params;
    /*-- Setup Time-Base (TB) Submodule --*/
    EPwm2Params.TBPRD = 18;
    EPwm2Params.TBCTL_CTRMODE = 2;
    EPwm2Params.TBCTL_SYNCSEL = 1;
    EPwm2Params.TBCTL_PHSEN = 0;
    EPwm2Params.TBCTL_PHSDIR = 0;
    EPwm2Params.TBPHS = 0;
    EPwm2Params.TBCTL_HSPCLKDIV = 5;
    EPwm2Params.TBCTL_CLKDIV = 4;

    EPwm2Params.CMPCTL_LOADAMODE = 0;
    EPwm2Params.CMPCTL_LOADBMODE = 0;
    EPwm2Params.CMPA = 0;
    EPwm2Params.CMPB = 0;

    EPwm2Params.AQCTLA = 144;
    EPwm2Params.AQCTLB = 144;
    EPwm2Params.AQCSFRC_CSFA = 0;
    EPwm2Params.AQCSFRC_CSFB = 0;
    EPwm2Params.AQCSFRC_RLDCSF = 0;

    EPwm2Params.DBCTL_OUT_MODE = 0;
    EPwm2Params.DBCTL_IN_MODE = 0;
    EPwm2Params.DBCTL_POLSEL = 0;
    EPwm2Params.DBRED = 0.0;
    EPwm2Params.DBFED = 0.0;

    EPwm2Params.ETSEL_SOCAEN = 0;
```

```

    EPwm2Params.ETSEL_SOCASEL = 1;
    EPwm2Params.ETPS_SOCAPRD = 1;
    EPwm2Params.ETSEL_SOCBEN = 0;
    EPwm2Params.ETSEL_SOCBSEL = 1;
    EPwm2Params.ETPS_SOCBPRD = 1;
    EPwm2Params.ETSEL_INTEN = 0;
    EPwm2Params.ETSEL_INTSEL = 1;
    EPwm2Params.ETPS_INTPRD = 1;

    EPwm2Params.PCCTL_CHPEN = 0;
    EPwm2Params.PCCTL_CHPFREQ = 0;
    EPwm2Params.PCCTL_OSHTWTH = 0;
    EPwm2Params.PCCTL_CHPDUTY = 0;

    EPwm2Params.TZSEL = 0;
    EPwm2Params.TZCTL_TZA = 3;
    EPwm2Params.TZCTL_TZB = 3;
    EPwm2Params.TZEINT_OST = 0;
    EPwm2Params.TZEINT_CBC = 0;

    config_ePWMRegs(&EPwm2Regs, &EPwm2Params);
}

for (arg = Phase; arg >= 6.2831853071795862;
arg -= 6.2831853071795862) {
    while (arg < 0.0) {
        arg += 6.2831853071795862;
    }
    FreqNorm = arg;
}
{
    real_T arg;
    Integrator_DSTATE = Integrator_IC;
    Filter_DSTATE = Filter_IC;
    Integrator_DSTATE_i = Integrator_IC_h;
    Filter_DSTATE_p = Filter_IC_e;
    Integrator_DSTATE_g = Integrator_IC_l;

```

```
Filter_DSTATE_d =Filter_IC_p;

    Integrator_DSTATE_o =Integrator_IC_b;

    Filter_DSTATE_do = Filter_IC_m;

    for (arg =Phase; arg >= 6.2831853071795862;
arg -= 6.2831853071795862) {
        }
        while (arg < 0.0)
{arg += 6.2831853071795862;
        }
        FreqNorm = arg;

    }
}
void terminate(void)
{}
*/
```