**Universidade de Brasília**
**Faculdade de Tecnologia**
**Departamento de Engenharia Mecânica**

# Projeto de um Controlador Fuzzy PID em Hardware Embarcado para um Exoesqueleto de Membro Inferior

Rodrigo Bonifácio de Medeiros

DISSERTAÇÃO DE MESTRADO

PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS MECATRÔNICOS

Brasília

2023

**Universidade de Brasília**
**Faculdade de Tecnologia**
**Departamento de Engenharia Mecânica**

# Projeto de um Controlador Fuzzy PID em Hardware Embarcado para um Exoesqueleto de Membro Inferior

Rodrigo Bonifácio de Medeiros

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Orientador: Prof. Dr. Daniel Muñoz Arboleda

Brasília
2023

Bonifácio de Medeiros, Rodrigo.
B769p Projeto de um Controlador Fuzzy PID em Hardware Embarcado para um Exoesqueleto de Membro Inferior / Rodrigo Bonifácio de Medeiros; orientador Daniel Muñoz Arboleda. -- Brasília, 2023.
71 p.

Dissertação de Mestrado (Programa de Pós-Graduação em Sistemas Mecatrônicos) -- Universidade de Brasília, 2023.

1. Otimização multiobjetiva. 2. Controlador PID com pré-compensador Fuzzy. 3. Manipulador robótico de 2 graus de liberdade. 4. Marcha humana. I. Muñoz Arboleda, Daniel, orient. II. Título

**Universidade de Brasília**
**Faculdade de Tecnologia**
**Departamento de Engenharia Mecânica**

# Projeto de um Controlador Fuzzy PID em Hardware Embarcado para um Exoesqueleto de Membro Inferior

Rodrigo Bonifácio de Medeiros

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Trabalho aprovado. Brasília, 31 de Janeiro de 2023:

---

**Prof. Dr. Daniel Muñoz Arboleda,**
**UnB/FT/ENM**
Orientador

---

**Prof. Dr. Guillermo Alvarez Bestard,**
**UnB/FT/ENM**
Examinador interno

---

**Prof. Dr. Abel Guilhermino da Silva Filho,**
**UFPE/CIn**
Examinador externo

---

**Prof. Dr. Renato Coral Sampaio,**
**UnB/FGA**
Suplente

Brasília
2023

# Agradecimentos

Agradeço, em especial, ao meu orientador e professor Daniel Muñoz Arboleda pela sua dedicação e disponibilidade em apoiar o desenvolvimento da minha dissertação. Agradeço também à minha família, à minha namorada Letícia Palheta Buril e ao meu amigo e colega de curso Elpidio C. De A. Bisneto pelo apoio motivacional nos momentos mais difíceis do meu mestrado.

*“O futuro pertence àqueles que acreditam na beleza de seus sonhos.”*
*(Eleanor Roosevelt)*

# Resumo

Os manipuladores robóticos são sistemas de múltiplas entradas e saídas (MIMO) com vários pontos de não-linearidades afetados por inúmeras incertezas e distúrbios. Os controladores Proporcional-Integral-Derivativo (PID) são amplamente utilizados na indústria para controle cinemático e dinâmico, no entanto, quando aplicados a sistemas MIMO, eles não são fáceis de ajustar e requerem melhorias de desempenho. Neste trabalho, é proposto um controlador PID com pré-compensador fuzzy (FP-PID), ajustados pelo algoritmo de otimização por enxame de partículas (PSO), bem como pelos algoritmos de otimização multiobjetivo *Non-dominated Sorting Genetic Algorithm II* (NSGA-II) e *Multi-Objective Differential Evolution* (MODE), para um manipulador robótico de dois graus de liberdade (2-DOF), que representa um exoesqueleto. Para validar o sistema foram utilizados dois conjuntos de dados reais da marcha humana: caminhada normal e subida de escada, para estimar a trajetória de erro do manipulador. As análises estatísticas dos algoritmos com dezesseis experimentos foram satisfatórias e a adição do pré-compensador fuzzy ao PID convencional resultou em uma redução do erro quadrático médio de uma das juntas do manipulador em até oitenta e três por cento em relação ao PID convencional, além da melhora na suavização do torque com os resultados multiobjetivos. O desenvolvimento de um co-projeto hardware-software para o modelo FP-PID do exoesqueleto também é foco, sendo implementado o sistema em um sistema em chip (SoC) baseado em *Field Programmable Gate Array* (FPGA), onde a planta e controlador PID são executados em software e o pré-compensador em hardware. O resultado mostra que a malha de controle permite manter o tempo de resposta dentro da faixa esperada da aplicação.

**Palavras-chave**: Otimização multiobjetiva. Controlador PID com pré-compensador Fuzzy. Manipulador robótico de 2 graus de liberdade. Marcha humana. Co-projeto hardware-software.

# Abstract

*Robotic manipulators are multiple-input multiple-output (MIMO) systems with several points of nonlinearity affected by numerous uncertainties and disturbances. Proportional-Integral-Derivative (PID) controllers are widely used in industry for kinematic and dynamic control, however, when applied to MIMO systems, they are not easy to tune and require performance improvements. In this work, a PID controller with fuzzy pre-compensator (FP-PID) is proposed, tuned by the particle swarm optimization (PSO) algorithm, as well as the multi-objective optimization algorithms Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Multi-Objective Differential Evolution (MODE), for a two-degree-of-freedom (2-DOF) robotic manipulator representing an exoskeleton. To validate the system, two real datasets of human gait were used: normal walking and stair climbing, to estimate the manipulator error trajectory. The statistical analyses of the algorithms with sixteen experiments were satisfactory and the addition of the fuzzy pre-compensator to the conventional PID resulted in a reduction of the mean square error of one of the manipulator joints by up to eighty-three percent compared to the conventional PID, as well as an improvement in torque smoothing with multi-objective results. The development of a hardware-software co-design for the FP-PID model of the exoskeleton is also a focus, with the system being implemented on a Field Programmable Gate Array (FPGA) based system-on-chip (SoC), where the plant and PID controller are executed in software and the pre-compensator in hardware. The result shows that the control loop allows maintaining the response time within the expected range of the application.*

**Keywords**: Multi-objective optimization. Fuzzy pre-compensated PID contoller. 2-DOF robotic manipulator. Human gait. Co-design hardware-software.

# Lista de ilustrações

# Lista de tabelas

# Sumário

# Lista de abreviaturas e siglas

| | |
|---|---|
| GRACO | Grupo de Automação e Controle |
| UnB | Universidade de Brasília |
| 2-DOF | *Two Degree of Freedom* |
| PID | Proporcional-Integral-Derivativo |
| MIMO | *Multiple Input Multiple Output* |
| SoC | *System On Chip* |
| FPGA | *Field Programmable Gate Array* |
| FP-PID | *Fuzzy Pre-compensated Proportional–Integral–Derivative* |
| IEEE | *Institute of Electrical and Electronics Engineers* |
| LARS | *Latin American Robotics Symposium* |
| SBR | *Brazilian Symposium on Robotics* |
| FLCs | *Fuzzy Logic Controllers* |
| MFC | *Membership Functions* |
| PC fuzzy | Pré-compensador fuzzy |
| PSO | *Particle Swarm Optimization* |
| NSGA-II | *Non-dominated Sorting Genetic Algorithm II* |
| MODE | *Multi-Objective Differential Evolution* |
| DE | *Differential Evolution* |
| LUTs | *Look-up Table* |
| CMOS | *Complementary Metal Oxide Semiconductor* |
| DSPs | *Digital Signal Processing* |
| BRAMs | *Block Random Access Memory* |
| I/0 | *Input/Output* |

| | |
|---|---|
| CLBs | *Configurable Logic Block* |
| FFs | *Flip-Flops* |
| MMC | *Mixed-Mode Clock Manager* |
| IP-Core | *Intellectual Property Core* |
| HNP | *Hybrid Neuroprosthesis* |
| BLEYE | *Blind People Eyes* |
| FOFP-FOPID | *Fractional Order Fuzzy - Precompensated Fractional Order* |
| ABC-GA | *Artificial Bee Colony-Genetic Algorithm* |
| VHDL | VHSIC Hardware Description Language |
| N/A | Não se aplica |
| MUX | Multiplexadores |
| NB | *Negative Big* |
| NM | *Negative Medium* |
| NS | *Negative Small* |
| ZE | *Zero* |
| PS | *Positive Small* |
| PM | *Positive Medium* |
| PB | *Positive Big* |
| RMSE | *Root Mean Squared Error* |
| F1 | Função custo 1 |
| F2 | Função custo 2 |
| MOPS | Mega Operações por Segundo |
| SMC | *Sliding Mode Control* |

# 1 Introdução

## 1.1 Contextualização e definição do problema

Exoesqueleto de membro inferior é um dispositivo mecânico externo que é acoplado ao corpo para ajudar a melhorar a mobilidade e força dos membros inferiores. Ele é projetado para ajudar pessoas com lesões na medula espinhal, doenças neurológicas e outras condições que afetam a capacidade de andar. Funcionam por meio de sensores e atuadores que detectam movimentos dos usuários e respondem movendo as pernas mecanicamente (PAMUNGKAS et al., 2019).

A interação entre o usuário e o exosqueleto é realizada principalmente por seu sistema de aquisição de dados, processamento e controle, os quais influenciam significativamente o efeito da assistência mecânica aos membros inferiores. No entanto, devido às limitações das técnicas de controle e às limitações computacionais do sistemas de processamento comumente usados em exoesqueletos, ainda é um desafio para alcançar segurança, estabilidade e conforto ao usuário a fim de popularizar a aplicação em larga escala (SUN et al., 2022).

A motivação para a elaboração deste trabalho é a melhoria do controle do protótipo de um exoesqueleto de membro inferior, vide Figura 1, desenvolvido pelo Grupo de Automação e Controle (GRACO) da Universidade de Brasília (UnB) (MARAFA, 2021) (KOENDJBIHARIE, 2017). De maneira simplificada, este dispositivo pode ser modelado como um manipulador robótico de dois graus de liberdade, ou *Two Degrees of Freedom* (2-DOF).

Manipuladores robóticos possuem uma gama de aplicabilidade, como em processos de manufatura, na indústria nuclear, em aplicações médicas, entre outras (LEE, K. et al., 2018). Para isso são necessários controladores eficientes para o posicionamento e movimentação das articulações com precisão. Entretanto, existem diversos fatores que afetam o desempenho dos controladores, tais como vários pontos de não linearidade, distúrbios oriundos de ruído no processo, saturação de atuadores, erros no processo de modelagem, entre outros (KUMAR, A.; KUMAR, V., 2017).

Os controladores Proporcional-Integral-Derivativo (PID) convencionais, embora simples de implementar e comumente usados na prática, sofrem de baixo desempenho quando aplicados em sistemas com não linearidades significativas e em sistemas de múltiplas entradas e saídas, ou *Multiple Input Multiple Output* (MIMO).

Com o advento do conceito de conjuntos fuzzy e da teoria da lógica fuzzy (ZADEH, 1965), diversas pesquisas na área de sistemas de controle foram desenvolvidas no intuito de facilitar o projeto de controladores para sistemas não lineares e sistemas MIMO como nos trabalhos de (BINGÜL; KARAHAN, 2011) e (ELAWADY; LEBDA; SARHAN, 2020). De

forma geral, os controladores fuzzy dispensam a criação de modelos dinâmicos dos sistemas a serem controlados, assim como a aproximação e linearização de modelos, e permitem o processamento de informação com incertezas (KOVACIC; BOGDAN, 2005). Tais vantagens tem mostrado que, para algumas aplicações, os controladores fuzzy têm resultados superiores às abordagens de controle convencionais, como por exemplo no problema do manipulador robótico de dois graus de liberdade (HULTMANN AYALA; DOS SANTOS COELHO, 2012). Por outro lado, as principais limitações dos controladores fuzzy é que precisam do conhecimento de um especialista para criar o conjunto de regras de inferência que relacionam entradas e saídas do controlador, além de possuir um alto custo computacional, o qual pode ser crítico para aplicações em sistemas embarcados com limitações de desempenho, memória e consumo energético (CHIU; LEE, P.-J., 2017), (MASMOUDI; HACHICHA; KAMOUN, 1999).

Figura 1 – Exosqueleto de membro inferior desenvolvido pelo grupo GRACO/UnB.

O ajuste dos parâmetros dos controladores PID e fuzzy para manipuladores robóticos MIMO é uma tarefa desafiadora porque as respostas de saída do sistema podem ser altamente dependentes. Isso significa que uma mudança em uma entrada pode afetar não apenas o parâmetro da junta de interesse, mas também das demais. Além disso, os sistemas MIMO podem ter respostas dinâmicas complexas e não-lineares, o que pode tornar difícil prever como o sistema irá reagir a uma mudança na configuração dos parâmetros dos controladores. O processo de ajuste requer uma abordagem iterativa e pode ser facilitada usando algoritmos

bioinspirados como nos trabalhos de (KUMAR, A.; KUMAR, V., 2017), (HULTMANN AYALA; DOS SANTOS COELHO, 2012) e (BINGÜL; KARAHAN, 2011).

Os controladores fuzzy podem ser implementados em um *System On Chip* (SoC) *Field-Programmable Gate Array* (FPGA) para melhorar a eficiência e flexibilidade do sistema de controle, permite a realização de operações lógicas e aritméticas de forma rápida e precisa, o que é importante para sistemas de controle em tempo real. Além disso, o FPGA é reconfigurável, o que permite a modificação e otimização do controlador fuzzy sem a necessidade de substituir o hardware. Essa abordagem é apresentada nos trabalhos de (ROMEIRO DE JESUS, 2017) e (CARVALHO FARIA, 2021).

## 1.2 Hipóteses de pesquisa

- Algoritmos bioinspirados podem ser utilizados para auxiliar no ajuste dos parâmetros PID e de um controlador fuzzy para um manipulador de dois graus de liberdade, uma vez que o desenvolvimento analítico pode ser complexo;
- Pré-compensadores fuzzy diminui o erro de trajetória de um manipulador robótico 2-DOF;
- Quanto maior o número de graus de liberdade de um manipulador robótico, maior é a complexidade do sistema e a quantidade de dados que precisam ser processados, portanto essa tarefa é melhor executada em hardware do que em software para atender o tempo de um sistema de controle de um exosqueleto.

## 1.3 Perguntas de pesquisa

- O ajuste dos parâmetros de um controlador PID e um controlador fuzzy para um manipulador robótico de dois graus de liberdade funcionam de maneira adequada para dados reais de uma marcha humana?
- A implementação em um SoC FPGA do controlador fuzzy de um manipulador robótico de dois graus de liberdade resolve o problema de erro de posição com torques aceitáveis para a atuação dos motores?
- O tempo de resposta da arquitetura proposta integrada ao sistema de aquisição e filtragem está dentro da faixa esperada (abaixo de 10 ms) para a aplicação em exosqueletos de membros inferiores?

## 1.4 Objetivos

### 1.4.1 Objetivos gerais

Ajustar os parâmetros de um controlador PID com um pré-compensador fuzzy, ou *Fuzzy Pre-compensated Proportional–Integral–Derivative* (FP-PID), utilizando algoritmos bioinspirados e implementar o sistema em um SoC FPGA para um manipulador robótico de dois graus de liberdade como modelo de simulação de um exosqueleto de membro inferior.

### 1.4.2 Objetivos específicos

Os principais objetivos específicos deste trabalho são:

- Formulação do problema de otimização matemática para a ajuste dos parâmetros do controle PID e do pré-compensador fuzzy através de algoritmos bioinspirados;
- Projetar e ajustar os parâmetros do modelo de controle PID da planta do manipulador robótico de dois graus de liberdade;
- Projetar e ajustar os parâmetros do modelo do pré-compensador fuzzy para melhorar o controle cinemático da planta;
- Desenvolvimento de um sistema embarcado baseado em co-projeto hardware/software usando SoC FPGA para implementação do controlador FP-PID e do modelo da planta.

## 1.5 Contribuições do trabalho

As contribuições deste trabalho são:

- O uso de dois conjuntos de dados reais de marchas humanas para validar o movimento de um manipulador robótico de dois graus de liberdade;
- A inserção de um pré-compensador fuzzy para melhorar a trajetória e o torque do manipulador robótico, e sua respectiva validação usando conjuntos de dados da marcha humana;
- O uso de um algoritmo de otimização bioinspirado em duas fases, uma para o ajuste dos parâmetros do PID e outra para ajuste dos parâmetros do pré-compensador fuzzy, tanto com algoritmos mono-objetivos como multiobjetivos;
- Validação da implementação física em um co-projeto hardware/software da arquitetura proposta;
- Artigo publicado no IEEExplore (MEDEIROS; MUÑOZ, 2022) e apresentado na *19th IEEE Latin American Robotics Symposium* LARS-SBR-2022.

## 1.6 Organização do trabalho

Em termos estruturais a organização deste documento é da seguinte maneira:

O capítulo 2 apresenta todas as fundamentações teóricas relacionadas ao projeto, como a modelagem matemática do manipulador robótico, controladores PID, controladores de lógica fuzzy, algoritmos bioinspirados, hardware reconfigurável e o estado da arte. O capítulo 3 apresenta a metodologia proposta para a implementação do pré-compensador fuzzy, o projeto do controlador FP-PID e sua implementação em um SoC FPGA. O capítulo 4 apresenta os resultados do ajuste dos parâmetros dos controladores pelos algoritmos bioinspirados, bem como a caracterização da arquitetura em hardware por meio de análises de consumo de síntese, potência, *layout* e validação dos testes do co-projeto hardware e software. Por fim, o capítulo 5 apresenta as discussões e conclusões referente ao projeto, sua viabilidade, e trabalhos futuros.

# 2 Fundamentação teórica

## 2.1 Modelagem matemática do manipulador robótico 2-DOF

De maneira simplificada, uma perna humana pode ser modelada como um controlador robótico de dois graus de liberdade, ou *Two-Degree of Freedom* (2-DOF), mostrado na Figura 2. Nela, há duas juntas com seus respectivos comprimentos $l_1$, $l_2$, massas $m_1$, $m_2$ e posições angulares $\theta_1$ e $\theta_2$.

Cada junta é submetida a um torque $\tau$ para iniciar o movimento. O modelo dinâmico para duas juntas ($n = 2$) que governa a movimentação desse manipulador é dado pela equação 2.1 (HULTMANN AYALA; DOS SANTOS COELHO, 2012).

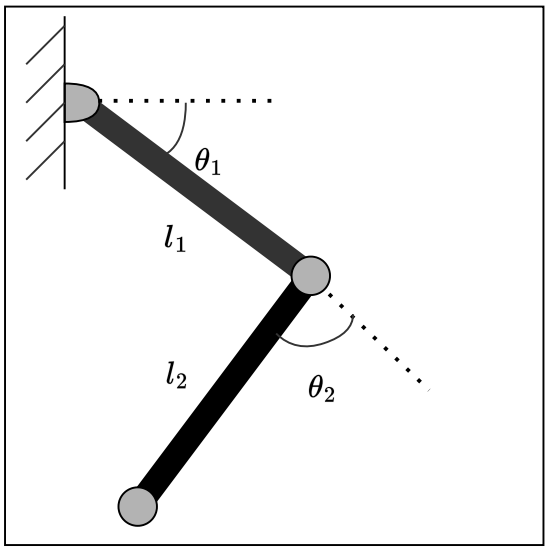


Figura 2 – Modelo 2-DOF.

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) \tag{2.1}$$

Onde $M(\theta) : \epsilon : \Re^{nxn}$ é a matriz positiva do sistema, $C(\theta,\dot{\theta}) : \epsilon : \Re^{nx1}$ é o vetor que representa os efeitos centrífugos e dos torques de Coriolis, $\tau : \epsilon : \Re^{nx1}$ é o vetor dos torques das juntas e, finalmente, $\theta, \dot{\theta}, \ddot{\theta}$ representam, respectivamente, a posição, velocidade e aceleração angular de cada junta. Todas as grandezas apresentadas aqui seguem a Sistema Internacional de Unidades. As formulações 2.2 e 2.3 mostram o desenvolvimento da equação diferencial.

$$\begin{aligned}\tau_1 &= m_2 l_2^2(\ddot{\theta_1} + \ddot{\theta_2}) + m_2 l_1 l_2 c_2(2\ddot{\theta_1} + \ddot{\theta_2}) \\ &+ (m_1{+}_2) l_1^2 \ddot{\theta_1} \cdots \\ &- m_2 l_1 l_2 s_2 \dot{\theta}_2^2 - 2 m_2 l_1 l_2 s_2 \dot{\theta_1}\dot{\theta_2} \cdots \\ &+ m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1\end{aligned} \tag{2.2}$$

$$\begin{aligned}\tau_2 &= m_2 l_1 l_2 c_2 \ddot{\theta_1} + m_2 l_1 l_2 s_2 \dot{\theta}_1^2 \cdots \\ &+ m_2 l_1 g c_{12} + m_2 l_2(\ddot{\theta_1} + \ddot{\theta_2})\end{aligned} \tag{2.3}$$

Onde $s_2 = \sin(\theta_2)$, $c_1 = \cos(\theta_1)$, $c_2 = \cos(\theta_2)$ e $c_{12} = \cos(\theta_1 + \theta_2)$, sendo que os índices 1 e 2 nas variáveis representam os parâmetros das juntas 1 e 2, respectivamente. Os valores utilizados para simulação, a partir da base de dados biomecânicos da marcha humana de uma pessoa de 56.7 kg, são: $l_1 = 0.314$ m, $l_2 = 0.425$ m, $m_1 = 5.670$ kg, $m_2 = 2.636$ kg (WINTER, 2009).

## 2.2 Controladores PID

Controladores PID são algoritmos de controle que utilizam o método Proporcional-Integral-Derivativo (PID) para controlar sistemas dinâmicos. O controlador proporcional age de acordo com a diferença entre a saída desejada e a saída atual, o controlador integral age para corrigir o erro acumulado ao longo do tempo, e o controlador derivativo age para prever e antecipar a variação futura do erro. Estes três componentes trabalham juntos para ajustar a saída do sistema para acompanhar a entrada desejada, e assim alcançar estabilidade e precisão no controle do processo, como mostra o segundo bloco pontilhado à direita da Figura 12. Eles são relativamente fáceis de projetar, implementar e ajustar, e têm uma boa estabilidade e desempenho e são geralmente mais adequados para sistemas lineares (OGATA, 2010).

Controlador PID são amplamente utilizados em aplicações industriais como controle de temperatura, pressão, velocidade, entre outros (SENTHIL KUMAR; KARTHIGAI AMUTHA, 2014).

## 2.3 Controladores de lógica fuzzy

Os controladores de lógica fuzzy ou *Fuzzy Logic Controllers* (FLCs) são um tipo de controlador que utilizam a teoria da inferência fuzzy para tomar decisões. Eles são baseados em regras fuzzy, que são definidas com base na experiência humana ou em dados históricos.

Essas regras são usadas para mapear as entradas do sistema para saídas, permitindo que o controlador tome decisões baseadas em dados imprecisos e incertos. Um conjunto Fuzzy difere do conjunto binário em relação aos valores que pode assumir, enquanto o conjunto binário é limitado a apenas dois elementos, geralmente representados como 0 e 1, um conjunto fuzzy permite a atribuição de valores diferentes de 0 e 1, possibilitando a representação de conceitos que não são facilmente quantificáveis em valores booleanos (CHEN; PHAM, 2019).

A estrutura básica de um controlador fuzzy consiste em três partes principais: fuzzificação (*fuzzification*), inferência fuzzy (*fuzzy inference*) e defuzzificação (*defuzzification*), como mostra a Figura 3.

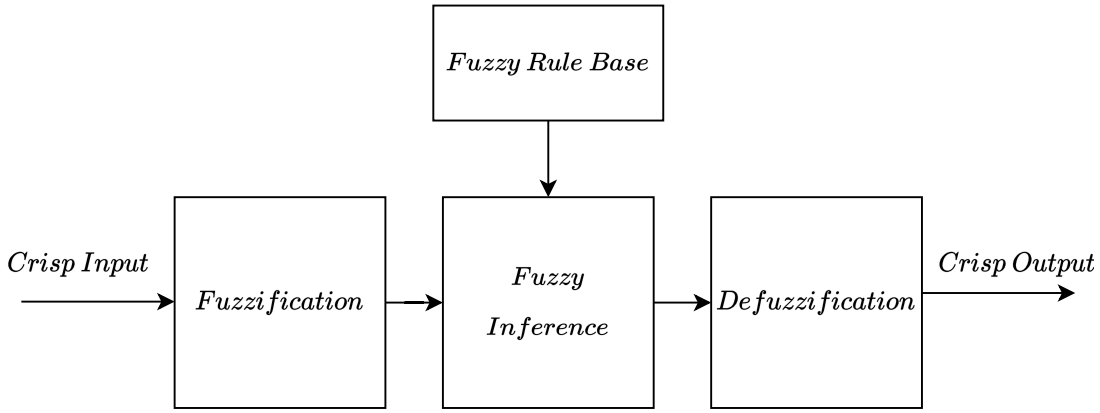


Figura 3 – Diagrama de blocos do FLC.

A fuzzificação é a etapa inicial e consiste em transformar dados *crisp*, que são dados numéricos precisos, em dados fuzzy, ou seja, variáveis linguísticas. Posteriormente, através do uso de funções de pertinência, ou Membership Functions (MFC), são atribuídos graus de pertencimento a diferentes valores de entrada. Por exemplo, se estiver trabalhando com temperatura, pode-se usar uma função de pertinência ($\mu$) para atribuir um grau de pertencimento de "quente" para valores acima de 30ºC e um grau de pertencimento de "frio" para valores abaixo de 10ºC. Existem várias MFC, tais como: Triângulo, Trapézio, Gaussiana, etc. A Figura 4 apresenta um exemplo. Em uma escala de 0 a 10, em que medida a comida é deliciosa? A nota 8 da comida é transformada em uma variável *crisp* e, em seguida, é qualificada como a variável linguística "deliciosa" usando uma função de pertinência. Neste caso, classificamos a comida como 8, que, dada a definição gráfica de "deliciosa", corresponde a $\mu$ = 0,7 para a função de pertinência de "deliciosa". (MATLAB, 2018) (CHEN; PHAM, 2019).

Dependendo da aplicação, é possível utilizar de diferentes formas e com diferentes parâmetros para representar a imprecisão e a incerteza de forma adequada

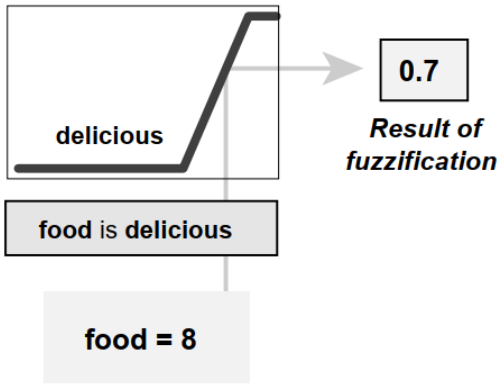


Figura 4 – Exemplo de fuzzificação (MATLAB, 2018).

A segunda etapa da inferência fuzzy é o processo de aplicação de regras (*fuzzy rule base*) para mapear as entradas (antecedentes) em saídas (consequentes) fuzzy, com base na combinação de regras e operações "E" (*AND*) ou "OU" (*OR*). Existem vários métodos de inferência, como o método Mamdani e o método Sugeno. Cada um possui suas próprias vantagens e desvantagens e é adequado para diferentes tipos de aplicações. No método Mamdani, as regras fuzzy são combinadas usando os operadores *AND* ou *OR*. A saída é gerada pela implicação e implementada para cada regra. Dois métodos integrados são suportados: mínimo (*min*), que trunca o conjunto fuzzy de saída, e produto (*prod*), que dimensiona o conjunto fuzzy de saída. A Figura 5 mostra um exemplo do método Mamdani explicado anteriormente. Já no método Sugeno, a saída é baseada em equações lineares para combinar as regras e determinar a saída (MATLAB, 2018).

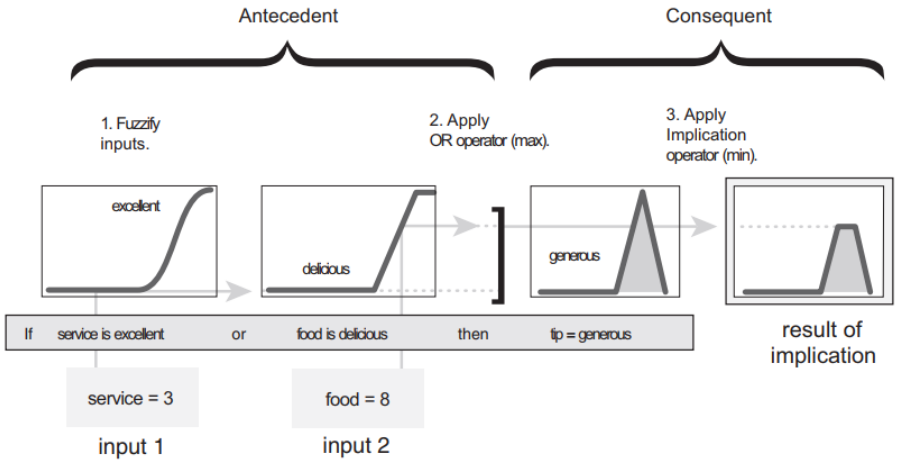


Figura 5 – Exemplo de inferência fuzzy Mamdani(MATLAB, 2018).

A última etapa é a defuzzificação (*defuzzification*) que consiste em transformar as saídas linguística em dados *crips*. A defuzzificação é necessária para que o sistema possa produzir saídas numéricas precisas e concretas, que possam ser interpretadas e utilizadas de forma eficaz.

Existem vários métodos de defuzzificação, incluindo o método centroide, o método da média dos máximos e o método da média ponderada. O método centroide, também conhecido como *fuzzy mean*, é o mais usado, pois é preciso e fácil de implementar. Consiste na média aritmética dos centros de gravidade dos conjuntos fuzzy aos quais o elemento pertence, ponderados pelo grau de pertinência. A equação 2.4 apresenta o fundamento matemático.

$$crisp = \frac{\sum_{i=1}^{j} \mu_i . c_i}{\sum_{i=1}^{j} \mu_i} \tag{2.4}$$

Onde *crisp* é saída do defuzificador, $j$ é a quantidade de conjuntos fuzzy aos quais o elemento pertence, $\mu_i$ $c_i$ é o grau de pertinência e o centro de gravidade, respectivamente. A Figura 6 apresenta uma representação gráfica do método.

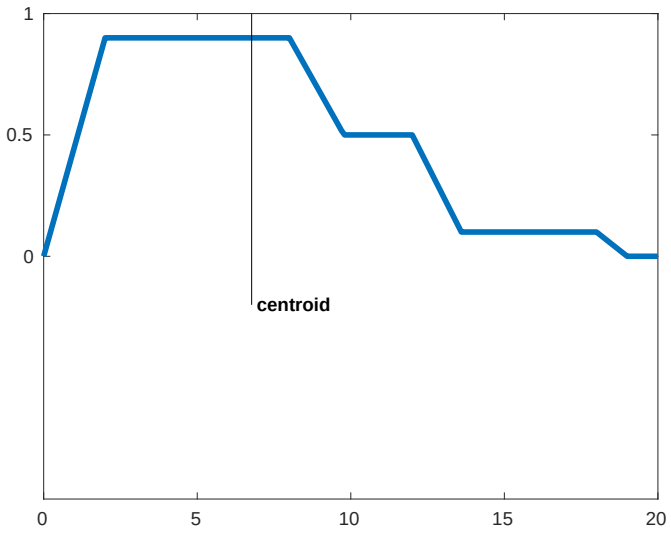


Figura 6 – Defuzzificação pelo método centroide (MATLAB, 2018).

Os FLCs têm sido recomendados em conjunto com os controladores PID convencionais devido aos seus vários benefícios, como incorporação de conhecimento humano especializado, não ser necessário modelo dinâmico exato, menos desenvolvimentos e custos de manutenções. Além disso, existem técnicas de otimização que podem ser utilizadas para ajustar os parâmetros dos PIDs e FLCs e melhorar sua performance, como o uso de algoritmos bioinspirados (HULTMANN AYALA; DOS SANTOS COELHO, 2012), (KUMAR, A.; KUMAR, V., 2017).

## 2.4 Algoritmos bioinspirados

Os algoritmos bioinspirados são métodos de busca e otimização que se inspiram em mecanismos de inteligência encontrados na natureza. Eles são projetados para encontrar soluções ótimas em problemas complexos e não lineares. Os algoritmos bioinspirados incluem algoritmos genéticos, algoritmos de enxame de partículas, algoritmos de colônia de formigas, algoritmos de busca de árvore de decisão e algoritmos de busca harmônica. Esses algoritmos são amplamente utilizados em aplicações como otimização de rotas, programação genética, aprendizado de máquina e inteligência artificial (YANG, 2010).

Este trabalho se concentra no algoritmo de otimização mono-objetivo PSO e em dois algoritmos de otimização multiobjetivo chamados NSGA-II e MODE, que serão mostrados nas seções seguintes.

### 2.4.1 Algoritmo mono-objetivo PSO

O algoritmo *Particle Swarm Optimization* (PSO), desenvolvido por Kennedy e Eberhart, é bioinspirado no comportamento de social de bandos de aves ou afins. Cada indivíduo é chamado de partícula e a população total de enxame. As vantagens do algoritmo são a busca global e fácil implementação sem necessidade de cálculo de gradiente. O objetivo final é a maximização ou minimização de uma função custo, sendo que a posição de uma partícula é uma possível solução do problema. As equações 2.5 e 2.6 mostram o fundamento matemático do algoritmo (KENNEDY; EBERHART, 1995).

$$x_{ij}^{(t+1)} = x_{ij}^{t} + v_{ij}^{(t+1)} \tag{2.5}$$

$$v_{ij}^{(i+1)} = \overbrace{w v_{ij}^{(i)}}^{\text{inércia}} + \overbrace{c_1 U_{1j}(y_{ij}^{(i)} - x_{ij}^{(i)})}^{\text{componente cognitivo}} + \overbrace{c_2 U_{2j}(y_{sj}^{(i)} - x_{ij}^{(i)})}^{\text{componente social}} \tag{2.6}$$

onde $i$ é a iteração atual, as grandezas $v_{ij}$ e $x_{ij}$, são a velocidade e posição de uma partícula, respectivamente. As atualizações das posições no espaço de busca N-dimensional em função da velocidade é mostrada na equação 2.5.

A variável $y_{ij}$ é a melhor posição individual, $y_{sj}$ a melhor posição global entre todas as partículas, $c_1$ é o coeficiente cognitivo, $c_2$ o coeficiente social, $U_1$ e $U_2$ são números entre 0 e 1 distribuídos uniformemente usados para adicionar aleatoriedade ao algoritmo a fim de haver uma exploração em diferentes regiões. Para evitar fuga no espaço de busca, é utilizado o fator de inércia $w$ que diminui linearmente durante a execução do algoritmo. A equação 2.6 mostra como essas velocidades são atualizadas em função dos parâmetros mencionados. Um valor alto de $c_1$ indica maior confiança na experiência individual, representada pela

componente cognitiva, enquanto um valor alto de $c_2$ indica maior confiança no enxame, representada pela componente social (KENNEDY; EBERHART, 1995).

O pseudocódigo é mostrado no Algoritmo 1 e possui três etapas: avaliação e identificação da melhor posição individual; identificação da melhor posição global e atualização dos parâmetros e partículas no espaço de busca.

**Algorithm 1** Particle swarm optimization algorithm

**Require:** $S, N, C_1, C_2, x_{max}, v_{max}, MAX_{iter}, threshold$
**Ensure:** $g_{best}$ and $f(g_{best})$
  Initialization ▷ swarm startup
  **while** $f(y_s < threshold)$ **do**
    **for** $k = 1$ to $S$ **do** ▷ finding the best individual
      **if** $f(x_k) \leq f(y_{ik})$ **then**
        $y_{ik} = x_k$
      **end if**
    **end for**
    calculate $y_s$ using the $S$ values $f(y_{ik})$ ▷ best global
    **for** $k = 1$ to $S$ **do** ▷ update
      **for** $j = 1$ to $N$ **do**
      $v_{kj} = wv_{kj} + c_1U_1(y_{ikj} - x_{kj}) + c_2U_2(x_{sj} - x_{kj})$
      $x_{kj} = x_{kj} + v_{kj}$
      **end for**
      Check bounds
    **end for**
  **end while**

### 2.4.2 Algoritmo multiobjetivo NSGA-II

NSGA-II (*Non-dominated Sorting Genetic Algorithm* II) é um algoritmo genético multiobjetivo que foi desenvolvido para lidar com problemas de otimização multiobjetivo. Há dois parâmetros importante, como a taxa de recombinação (*crossover*) que é um operador genético que combina informações genéticas de dois ou mais indivíduos da população para criar novos indivíduos com características diferentes. A taxa de mutação, por outro lado, é a probabilidade de um gene de um indivíduo sofrer uma mutação durante o processo de evolução. A mutação é uma mudança aleatória no gene que pode introduzir variação na população e ajudar a evitar a convergência prematura para soluções subótimas. O NSGA-II é baseado no algoritmo original NSGA, mas inclui algumas melhorias para tornar o algoritmo mais eficiente e eficaz. O algoritmo funciona da seguinte maneira (DEB et al., 2002):

- Inicialmente, é gerada uma população inicial de soluções candidatas aleatoriamente. Cada solução é avaliada em relação aos objetivos definidos no problema;
- As soluções são classificadas com base na sua não-dominação, ou seja, são agrupadas de acordo com seus níveis de dominação. As soluções que não são dominadas por

outras soluções são chamadas de soluções "não-dominadas" e fazem parte do conjunto de soluções Pareto-ótimas. Uma solução A domina outra solução B se A é igual ou melhor do que B em todos os objetivos e é melhor em pelo menos um objetivo. Uma solução é não-dominada se não é dominada por nenhuma outra solução do conjunto;

- Uma função de seleção é aplicada para selecionar os indivíduos para reprodução. A seleção é baseada na não-dominação e no valor de diversidade, para evitar que o algoritmo se torne preso em um subconjunto da solução Pareto-ótima;
- Os indivíduos selecionados são então submetidos a operações de reprodução, como o (*crossover*) e taxa de mutação, para gerar novos indivíduos;
- A nova geração é então avaliada e classificada com base na não-dominação, e o processo é repetido até que se atinja uma solução satisfatória.

### 2.4.3 Algoritmo multiobjetivo MODE

MODE (*Multi-Objective Differential Evolution*) é um algoritmo de otimização multi-objetivo baseado em evolução diferencial, ou *Differential Evolution* (DE) (STORN; PRICE, 1997). O algoritmo funciona criando uma população inicial de soluções candidatas chamadas de cromossomos. Cada vetor é avaliado de acordo com as funções custo. Então, o algoritmo começa a evoluir a população através de gerações sucessivas, usando operações genéticas, com a taxa de mutação e *crossover*, para criar novos indivíduos que são melhores adaptados ao problema. O algoritmo segue os seguintes passos (ROBIČ; FILIPIČ, 2005a):

1. Inicialize a população inicial de soluções candidatas (cromossomos);
2. Avalie cada indivíduo de acordo com as funções custo;
3. Enquanto o critério de parada não for atingido:
   - **a** Selecione três indivíduos diferentes da população atual para gerar uma nova solução.
   - **b** Aplique a operação de mutação para gerar uma nova solução;
   - **c** Aplique a operação de *crossover* para gerar uma nova solução;
   - **d** Selecione a melhor solução entre a nova solução gerada e o indivíduo original.
   - **e** Avalie a nova solução;
   - **f** Adicione a nova solução à população atual;
4. Retorne a frente de Pareto.

A principal vantagem do NSGA-II e MODE é a sua capacidade de gerar soluções Pareto-ótimas, ou seja, soluções que não podem ser melhoradas em relação a um objetivo sem

prejudicar outro objetivo. Além disso, o algoritmo também é capaz de lidar com problemas de alta dimensionalidade e é relativamente fácil de implementar.

## 2.5 Hardware reconfigurável

Hardwares reconfiguráveis são dispositivos baseados em lógica programável, ou seja, possuem operações internas que podem ser definidas pelo usuário, o que permite o desenvolvimento rápido de protótipos de circuitos digitais (GARCIA et al., 2006).

### 2.5.1 FPGAs

Os *Field Programmable Gate Arrays* (FPGAs) são um tipo de hardware reconfigurável composto por uma matriz bidimensional de portas lógicas programáveis através de *Look-up Tables* (LUTs), lógicas fixas, recursos de roteamento implementados na tecnologia *Complementary Metal Oxide Semiconductor* (CMOS), blocos de processamento digital de sinais, ou *Digital Signal Processing* (DSPs), blocos de implementação de memórias, ou *Block Random Access Memory* (BRAMs) e blocos para a interface de entrada e saída, ou *Input/Output* (I/O).

A Figura 7 mostra a arquitetura de alto nível de um FPGA. O bloco *Programmable interconnect* possui interconexões programáveis por software e implementadas em hardware. O bloco de I/O é composto por circuitos para acesso a periféricos externos, como LEDs, chaves, *displays*, entre outros (INSTRUMENTS, 2013).

Os blocos lógicos são compostos por *Configurable Logic Blocks* (CLBs) e a Figura 8 mostra a arquitetura desses blocos para a família 7 da Xilinx fabricada usando a tecnologia de processo de 28 nm (XILINX, 2016). Cada CLB é composto por 2 *slices* independentes, cada um possui:

- 4 *logic-function generator* LUTs de 6 entradas e 2 saídas para a formação de lógica combinacional;
- 8 elementos de armazenamento da saída do LUT, *Flip-Flops* (FFs) do tipo D, sensíveis ao nível de subida ou descida do sinal de *clock*;
- Multiplexadores (MUX) de diversos propósitos, por exemplo, para direcionar lógica ou para associar CLBs e criar circuitos mais complexos;
- *Carry* lógico para a melhora na performance de funções aritméticas como somadores, subtratores e comparadores.

Para funções mais complexas, como multiplicações e filtros, o FPGA utiliza os blocos DSPs que estão implementados em silicio e permitem trabalhar a frequência de operação de até 500 MHz.

Os itens listados são responsáveis pela implementação de funções lógicas, aritméticas e de memória distribuída. Alguns *slices* possuem capacidade de armazenar dados nos blocos de memória BRAM.

A Figura 9 mostra um exemplo de implementação de um multiplexador 16:1 em uma *slice*, observe o uso de LUTs na entrada dos sinais, multiplexadores nas saídas das LUTs e as saídas armazenadas em um FF ativo pela borda de *clock*.

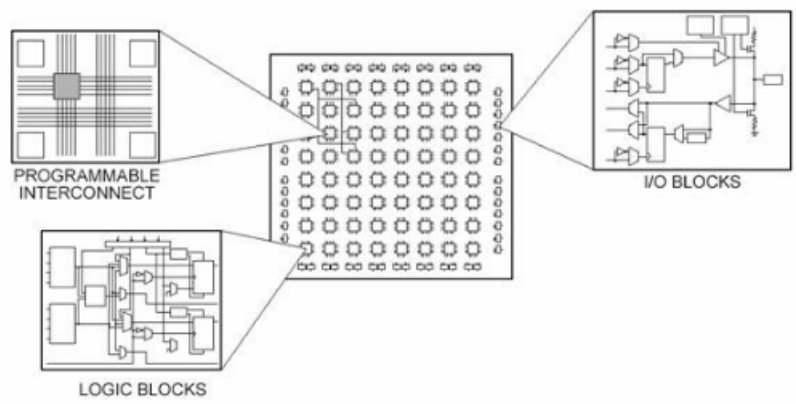


Figura 7 – Arquitetura de alto nível de um FPGA (INSTRUMENTS, 2013).

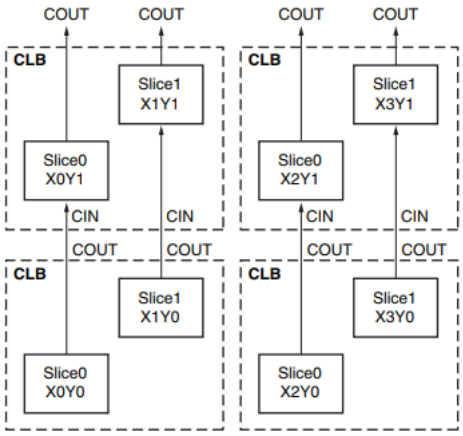


Figura 8 – Arquitetura de uma CLB (XILINX, 2016).

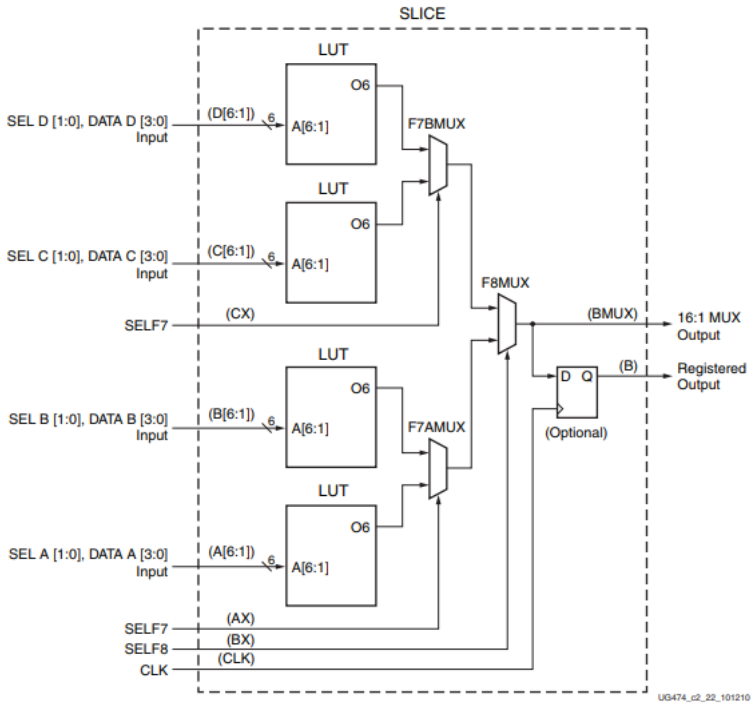


Figura 9 – Implementação de um MUX 16:1 em uma *slice* (XILINX, 2016).

A Tabela 1 enumera as vantagens e desvantagens no uso de FPGAs para desenvolvimento de projetos de sistemas embarcados.

Tabela 1 – Vantagens e desvantagens no uso do FPGA (GARCIA et al., 2006).

| | |
|---|---|
| VANTAGENS: | Maior velocidade de processamento em relação ao software; prototipagem de circuitos integrados; redundância para sistemas críticos e paralelização de processos. |
| DESVANTAGENS: | Alto custo de aquisição e desenvolvimento; alto consumo de potência para funcionamento. |

### 2.5.2 Sistema em chip Zynq-7000

O Zynq 7000 é uma série de dispositivos *System-on-Chip* (SoC) desenvolvidos pela Xilinx que combina um processador ARM Cortex-A9 com um FPGA (*Field-Programmable Gate Array*) programável em um único chip. A *Zynq 7000 Processing System*, é uma parte importante do SoC Zynq 7000, que fornece uma estrutura de gerenciamento de sistema e recursos de processamento para o FPGA programável.

Alguns dispositivos da família *Zynq 7000 Processing System* incluem dois núcleos ARM Cortex-A9 que podem ser usados para executar tarefas de alto nível, como comunicação de rede, gerenciamento de memória e sistemas operacionais. Ela também inclui um conjunto completo de periféricos, como interfaces de rede, USB, I2C, SPI, UART e GPIO, que podem

ser facilmente acessados pelos núcleos ARM ou pelos circuitos programáveis do FPGA como mostra a Figura 10. Também inclui o *AXI (Advanced eXtensible Interface) interconnect*, que é usado para conectar os núcleos ARM e periféricos mapeados no FPGA. Isso permite que os usuários criem soluções personalizadas (*ad-hoc*) combinando processamento de alto nível e lógica programável para atender às necessidades específicas de seus sistemas (ZYNQ-7000…, 2018).

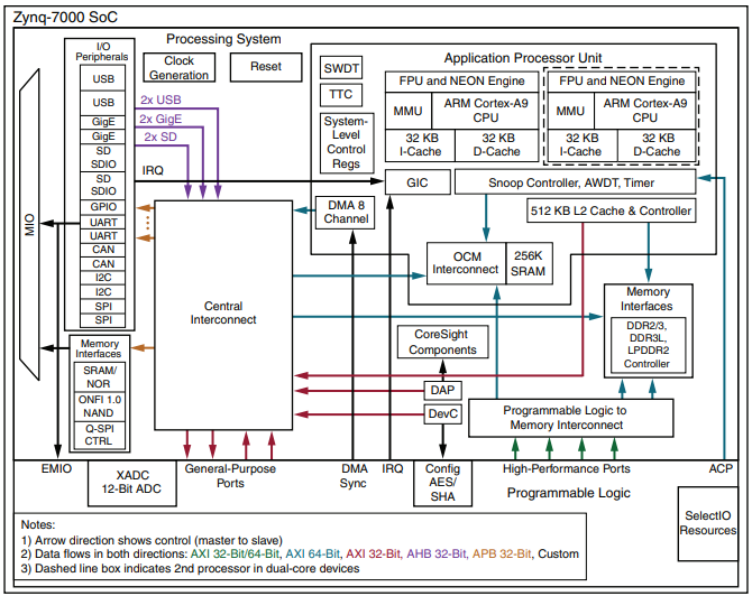


Figura 10 – Visão geral da arquitetura do SoC Zynq-7000 (ZYNQ-7000…, 2018).

### 2.5.3 Kit de desenvolvimento PYNQ-Z2

A PYNQ-Z2 é uma placa de desenvolvimento baseada no SoC Zynq-7020 da Xilinx, o qual possui um FPGA da família Artix-7 (dispositivo XC7Z020) da Xilinx. A arquitetura desse modelo utiliza 85000 células de lógica programável; 53200 *Slices* LUTs; 106400 FFs; 220 DSP *Slices*; memória RAM de 4.9 Mb distribuídos em 140 blocos de 36 Kb; 4 *Mixed-Mode Clock Manager* (MMCM) para geração de diferentes sinais de *clock*.

A PYNQ-Z2 foi projetada para ser usada com o sistemas operacional Ubuntu e a biblioteca PYNQ, que é uma plataforma de desenvolvimento de alto nível para o FPGA. Possui 512MB de memória DDR3, 16MB de flash SPI, interfaces USB, Ethernet, HDMI, entre outras. Ela possibilita o acesso às capacidades de processamento de alto desempenho e lógica programável do FPGA, através de uma interface Python de alto nível chamada de overlay. Isso permite que os desenvolvedores criem soluções personalizadas sem precisar lidar com a complexidade da programação de baixo nível dos FPGAs (PYNQ-Z2…, 2018).

A PYNQ-Z2 também inclui uma série de exemplos prontos para uso e projetos de demonstração, que mostram como acessar os recursos do FPGA, como interfaces de rede, de vídeo, de áudio, entre outros, através da biblioteca PYNQ. A PYNQ-Z2 é amplamente

utilizada em aplicações como automação industrial, visão computacional, comunicações, armazenamento e sistemas embarcados (SEVAK; PAWAR, 2022).

## 2.6 Estado da arte

A tabela 2 apresenta uma comparação entre vários estudos relacionados a este trabalho no âmbito de pesquisas principais para manipuladores robóticos. Cada entrada, com a ordem do ano de publicação, destaca as seguintes características: autor, tipo de manipulador robótico utilizado, trajetória do manipulador e tipo de ajuste dos parâmetros.

Tabela 2 – Trabalhos correlatos para manipuladores robóticos.

| Autor | Manipulador robótico | Trajetória do manipulador | Ajustes dos parâmetros |
|---|---|---|---|
| **Este trabalho** | **2-DOF** | **Marcha humana** | **PSO, NSGA-II, MODE** |
| (GARCÍA-MARTÍNEZ et al., 2020) | Não | Polinomial | PID-Type FLC |
| (AZAR; SERRANO, 2019) | 2-DOF | Polinomial | Não |
| (LI et al., 2018) | Exosqueleto | Marcha humana | Não |
| (KUMAR, A.; KUMAR, V., 2017) | 2-DOF | Polinomial | ABC-GA |
| (HULTMANN AYALA; DOS SANTOS COELHO, 2012) | 2-DOF | Polinomial | NSGA-II |
| (BINGÜL; KARAHAN, 2011) | 2-DOF | Polinomial | PSO |
| (KIM, J.-H.; KIM, K.-C.; CHONG, 1994) | Não | Polinomial | Não |

Os autores (GARCÍA-MARTÍNEZ et al., 2020) propõem uma nova estratégia de ajuste de controlador fuzzy do tipo PID baseada em relações fuzzy diretas para calcular as constantes do PID. O algoritmo de controle de movimento é composto por FLC do tipo PID e perfil de velocidade em forma de S. O controlador de auto-ajuste é realizado online, com auxílio de um FPGA, e depende do erro e da mudança no erro para se adaptar de acordo com as variações do sistema. Os resultados experimentais foram obtidos em uma plataforma linear integrada por um motor DC conectado a um codificador para medir a posição, Os valores de tempo de subida, *overshoot* e tempo de acomodação medidos na experimentação são 0,124 s, 8,985% e 0,248 s, respectivamente. O desempenho do controlador é comparado com diferentes arquiteturas de controle relatadas na literatura após a aplicação de um sinal de entrada de etapa ao motor DC.

No trabalho realizado por (AZAR; SERRANO, 2019), é descrita uma técnica para controle da posição e trajetória de braços robóticos. O método emprega um controlador PID 2-DOF para cada articulação, além de um compensador fuzzy tipo-2 centralizado para controlar as variáveis de orientação. Os resultados obtidos através de simulação numérica demonstraram que a técnica proposta aprimora a resistência e o desempenho do sistema, mesmo quando há perturbações externas. A utilização do compensador fuzzy tipo-2 apresenta benefícios como a capacidade de modelar o sistema como um sistema linear, e a possibilidade de ajustar os parâmetros do controlador PID numericamente para atender requisitos específicos. O estudo conclui que a estratégia desenvolvida pode ser aplicada em diferentes configurações de controladores PID.

O artigo de (LI et al., 2018) descreve o desenvolvimento e implementação de uma arquitetura de sistema de controle embutido modular para uma neuroprótese híbrida, ou *Hybrid Neuroprosthesis* (HNP), que visa restaurar as funções de locomoção em indivíduos com lesão medular. O sistema consiste em componentes eletrônicos e firmware personalizados que permitem a conectividade sem fio para o controle em malha fechada e estimulação neural. Testes de bancada e experimentação em indivíduos com lesão medular comprovaram a eficácia do sistema HNP. O trabalho futuro envolverá a melhoria da potência computacional e desempenho em tempo real para pesquisas de assistência à marcha mais avançadas e a investigação de uma arquitetura distribuída para permitir a modularidade máxima do sistema. O sistema HNP é um exoesqueleto totalmente funcional para caminhar após a paraplegia e tem potencial para ser uma plataforma de pesquisa para aplicações de robótica assistiva.

Em (KUMAR, A.; KUMAR, V., 2017), foi desenvolvido um novo tipo de controlador para um manipulador 2-DOF, chamado de *Fractional Order Fuzzy-Precompensated Fractional Order* PID (FOFP-FOPID). Ele foi projetado para resolver o problema de seguir uma trajetória polinomial específica com alta precisão. Para ajustar os parâmetros do controlador de maneira eficiente, foi utilizada uma técnica de otimização chamada algoritmo genético-colônia de abelhas artificiais, ou *Artificial Bee Colony-Genetic Algorithm* (ABC-GA). A eficácia do controlador FOFP-FOPID foi demonstrada por meio da comparação com outros controladores existentes e análises de robustez foram realizadas para garantir que o sistema seja capaz de lidar com variações nos parâmetros e perturbações externas. Os resultados da simulação mostraram que o controlador FOFP-FOPID é capaz de alcançar um rastreamento preciso da trajetória e melhorar a robustez do sistema.

Uma das principais referências deste trabalho é o apresentado pelo autor (HULTMANN AYALA; DOS SANTOS COELHO, 2012), em que utiliza uma abordagem de otimização para projetar e ajustar os parâmetros dos controladores PID para um manipulador 2-DOF. Utiliza o algoritmo NSGA-II para resolver o problema de otimização multiobjetivo, que envolve a busca por um conjunto de soluções Pareto-ótimas que satisfaçam simultaneamente o objetivo de minimizar o erro de trajetória e suavizar o torque do manipulador. Os resultados numéricos de simulação mostram que o método proposto baseado no NSGA-II é eficaz para encontrar soluções simples e robustas que proporcionam bom desempenho de rastreamento de referência em malha fechada de uma função polinomial para o manipulador robótico.

No trabalho de (BINGÜL; KARAHAN, 2011), foi realizado o controle de um manipulador robótico 2-DOF utilizando um FLC ajustado através do algortimo PSO. Para uma trajetória específica, os parâmetros FLC do tipo Mamdani, tais como funções de pertinência gaussiana nas entradas e saídas, foram otimizados utilizando três diferentes funções custo. A fim de comparar o desempenho do FLC otimizado com outros controladores, um controlador PID também foi ajustado através do PSO. Para avaliar a robustez dos controladores ajustados,

foram realizados testes onde os parâmetros do modelo e a trajetória foram modificados, e ruído branco foi adicionado ao sistema. Os resultados da simulação demonstraram que o FLC ajustado obteve um desempenho superior e foi robusto do que o controlador PID ajustado da mesma maneira, no que se refere ao controle de trajetória do robô.

O trabalho clássico na área de pré-compensadores (PC fuzzy) é apresentado por (KIM, J.-H.; KIM, K.-C.; CHONG, 1994). O artigo apresenta uma solução para melhorar o desempenho de controladores PID em sistemas que contêm não linearidades desconhecidas, como zonas mortas, saturação e histerese. Para resolver este problema, propõe-se uma abordagem de PC baseada em lógica fuzzy. Os resultados dos experimentos realizados em um banco de testes de controle de posição de um servomotor DC sob condições de carga variáveis mostram que os controladores PID com PC fuzzy têm um desempenho superior em comparação com os controladores PID convencionais. Além disso, a solução proposta é robusta a variações na carga, correspondendo a variações no nível de não linearidades na planta.

A tabela 3 apresenta principalmente pesquisas na área de arquiteturas em hardware. Cada entrada destaca as seguintes características: tipo de sistema fuzzy, plataforma, co-projeto hardware/software, representação numérica e gerador automático de códigos. Os trabalhos de (CARVALHO FARIA, 2021) e (LI et al., 2018) são apresentados novamente por apresentarem características importantes para hardware. Não se aplica (N/A) significa que não foram encontradas informações suficientes sobre a característica.

Tabela 3 – Trabalhos correlatos para arquiteturas em hardware.

| Autor | Tipo de sistema fuzzy | Plataforma | Co-projeto hardware/software | Representação numérica | Gerador automático de códigos |
|---|---|---|---|---|---|
| Este trabalho | Mamdani | FPGA | Sim | Ponto fixo | Sim |
| (CARVALHO FARIA, 2021) | Mamdani | FPGA | Sim | Ponto flutuante | Não |
| (GARCÍA-MARTÍNEZ et al., 2020) | N/A | FPGA | Sim | N/A | Não |
| (LI et al., 2018) | N/A | Microcontrolador | Sim | N/A | Não |
| (ROMEIRO DE JESUS, 2017) | Mamdani/Sugeno | FPGA | Não | Ponto flutuante | Sim |
| (SÁNCHEZ-SOLANO; BROX, 2014) | Mamdani/Sugeno | FPGA | Não | Ponto fixo | Não |

O autor (CARVALHO FARIA, 2021) criou uma arquitetura em hardware reconfigurável usando FPGA, a etapa de defuzzificação de sistemas fuzzy que utilizam o método de inferência Mamdani. O sistema utiliza um *Intellectual Property Core* (IP-Core) em ponto flutuante para acelerar o cálculo do centroide e máquinas de estados finitos para realizar os processos de cálculo. A análise de desempenho comparou os resultados obtidos no modelo de referência em linguagem C com os valores provenientes das simulações em hardware, apresentando um erro quadrático médio em torno de $10^{-7}$.

No estudo de (ROMEIRO DE JESUS, 2017) é apresentado um sistema chamado *Blind People Eyes* (BLEYE), que tem como objetivo ajudar deficientes visuais a se locomoverem em ambientes fechados utilizando óculos equipados com sensores de medição de distância. Para evitar obstáculos, é utilizado um controlador fuzzy, que foi desenvolvido com o paralelismo dos algoritmos usando arquiteturas de hardware mapeadas em dispositivos FPGAs e representação numérica em ponto flutuante. Os testes realizados mostraram que o sistema Bleye

tem um tempo de execução de 17.5 $\mu$s. A escolha da representação de uma representação numérica de 27 bits foi eficiente em relação ao consumo de recursos. Além da criação de um gerador automático de código *Very high speed integrated circuits Hardware Description Language* (VHDL) a partir de um modelo de alto nível de controladores fuzzy Takagi-Sugeno genéricos.

Já em (SÁNCHEZ-SOLANO; BROX, 2014), duas estratégias de projeto para a síntese automática de controladores fuzzy embarcados foram descritas neste capítulo. Elas demonstram que a disponibilidade de um fluxo de projeto, suportado pelo uso de bibliotecas e ferramentas de síntese de hardware fornecidas pelo ambiente Xfuzzy, acelera consideravelmente a implementação de sistemas fuzzy em hardware, facilitando a exploração do espaço de projeto para uma determinada aplicação. Uma das estratégias de projeto é focada em implementações de hardware de sistemas fuzzy em FPGAs da Xilinx. Como demonstrado pelos resultados obtidos, a opção baseada em VHDL geralmente permite alcançar melhores resultados em termos de desempenho e consumo de recursos.

Com base nos resultados e difusão dos trabalhos acima e de outros trabalhos correlatos, justifica-se o uso de manipuladores robóticos 2-DOF, a implementação do PC fuzzy em hardware e a combinação com o PID e a realização do ajuste dos parâmetros com os algoritmos mono-objetivo PSO e multiobjetivo NSGA-II. Para fins de comparação, também é utilizado o algoritmo multiobjetivo MODE.

# 3 Metodologia

## 3.1 Projeto do controlador FP-PID para o manipulador robótico 2-DOF

O controlador *Fuzzy Pre-compensated Proportional Integral Derivative* (FP-PID) proposto neste trabalho é composto por duas arquiteturas idênticas, representando a junta 1 e 2. Cada arquitetura possui dos componentes: o pré-compensador (PC) Fuzzy e o controlador PID convencional, representadas pelas linhas tracejadas na Figura 12. O primeiro controlador tem a função de modificar o sinal de controle para compensar os erros nas respostas de saídas. O segundo controlador tem como objetivo controlar as variáveis do manipulador robótico $\theta_1(t)$ e $\theta_2(t)$ em radianos, aproximando-as dos valores desejados da marcha humana $\theta_{d1}(t)$ e $\theta_{d2}(t)$, que também estão em radianos. As equações que descrevem a dinâmica são apresentadas em 3.1, onde $n = 1{,}2$ representam as juntas 1 e 2, respecitvamente (KUMAR, A.; KUMAR, V., 2017).

$$e_n(t) = \theta d_n(t) - \theta_n(t) \tag{3.1}$$

Onde $e_n(t)$ é a diferença entre o valor da trajetória desejada $\theta_{dn}(t)$ e o valor de saída da planta do manipulador $\theta_n(t)$. A diferença entre o erro atual e o erro do tempo anterior é apresentada na equação 3.2 a seguir.

$$\Delta e_n(t) = e_n(t) - e_n(t-1) \tag{3.2}$$

O cálculo das variáveis anteriores é importante para definir as entradas dos blocos $FLC_n$. As entradas $e_n(t)$ e $\Delta e_n(t)$ são multiplicadas pelas constantes $K_{E_n}$ e $K_{ED_n}$, respectivamente. A saída gerada pelo processamento fuzzy é multiplicada pela constante $K_{Sn}$ para fornecer a saída a ser somada a $\theta_{d_n}$ e gerar o valor pré-compensado $\theta_{pre_n}(t)$, como mostrado na equação 3.3.

$$\theta pre_n(t) = K_{Sn}.FLC_n(K_{E_n}.e_n(t), K_{ED_n}.\Delta e_n(t)) + \theta d_n(t) \tag{3.3}$$

Agora, o novo erro $e'n(t)$ para a entrada do controlador PID é a diferença entre a posição desejada modificada $\theta pre_n(t)$ e a posição atual da planta $\theta_n(t)$, mostrado na equação 3.4.

$$e'_n(t) = \theta pre_n(t) - \theta_n(t) \tag{3.4}$$

O torque $\tau(t)$, que é a entrada da planta do manipulador, é calculado pela fórmula do PID convencional mostrada em 3.5.

$$\tau(t) = K_{Pn}.e'_n(t) + K_{Dn}.\Delta e'_n(t) + K_{In}.\int e'_n(t)dt \tag{3.5}$$

Ao final do processo, o torque gerado é aplicado às juntas do manipulador 2-DOF, utilizando a planta comum para ambas, juntamente com as equações apresentadas na seção 2.1, para determinar as posições $\theta_1(t)$ e $\theta_2(t)$, que serão responsáveis pelo movimento do manipulador. Essas posições são então enviadas de volta ao sistema de controle para finalizar o ciclo.

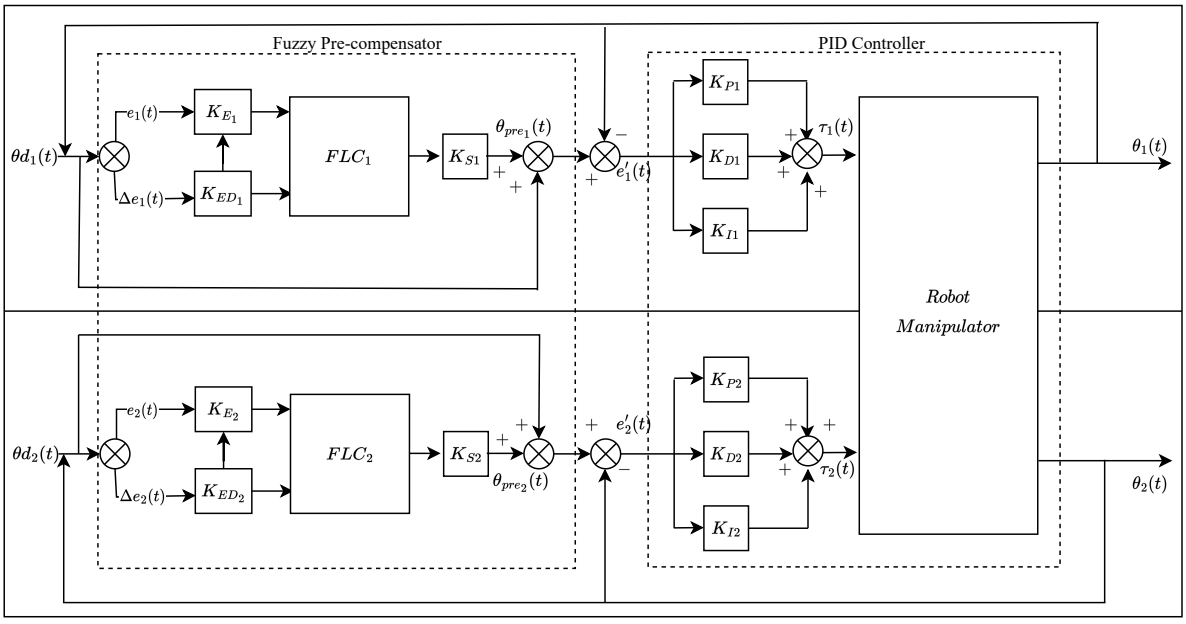


Figura 12 – Diagrama do controlador FP-PID para o manipulador robótico 2-DOF.

## 3.2 Projeto do PC fuzzy para o manipulador robótico 2-DOF

O software Matlab R2017b (MATLAB, 2018) foi utilizado tanto para a implementação do controlador fuzzy PC e para posterior ajuste dos parâmetros do controlador FP-PID baseado em algoritmos bioinspirados. Lembrando que o FLC possui três blocos: fuzzificação, inferência e defuzzificação. Para cada junta do manipulador robótico, haverá um PC fuzzy e as entradas dos controladores serão $\theta_n(t)$ e $\Delta_n(t)$, e a saída será $\theta_{pre_n}(t)$ sem o fator de escala

$K_{Sn}$, todos normalizados entre [-1, 1], de acordo com a Figura 12. Todas as escolhas foram baseadas na facilidade de implementação em hardware e nos trabalhos de (KIM, J.-H.; KIM, K.-C.; CHONG, 1994) e (KUMAR, A.; KUMAR, V., 2017).

A Figura 13 mostra a função de pertinência triangular para ambas entradas (antecedentes) e a saída (consequente). As sete variáveis linguísticas são : NB (*Negative Big*), NM (*Negative Medium*), NS (*Negative Small*), ZE (*Zero*), PS (*Positive Small*), PM (*Positive Medium*), PB (*Positive Big*).

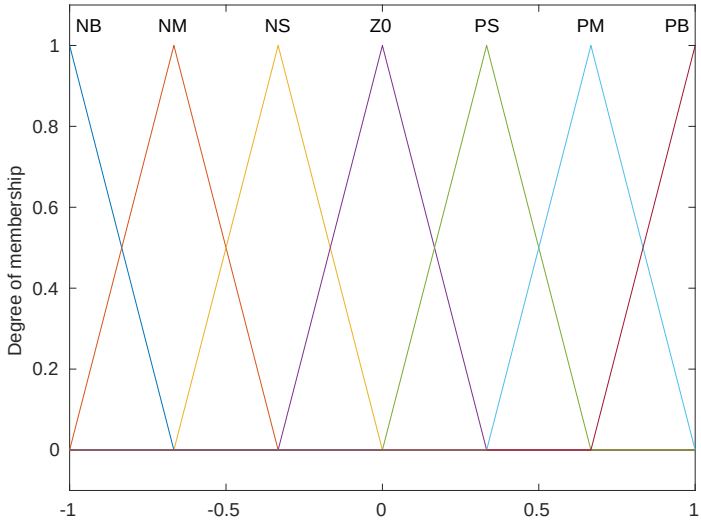


Figura 13 – Funções de pertinência para o método de fuzzificação.

Na inferência fuzzy, é utilizado o método de Mamdani para gerar a ação de controle adequada a partir da base de regras da Tabela 4. Essa associação é baseada em conhecimento humano especializado ou experimentos, baseado no trabalho de (KIM, J.-H.; KIM, K.-C.; CHONG, 1994) . Há um total de 27 regras de associação dos antecedentes para a geração do consequente, como por exemplo "SE o $e$ for ZE e $\Delta e$ NM, ENTÃO a saída é NM". Este controle serve para compensar a entrada do PID e melhorar o controle da planta, como o erro acumulado e o *overshoot*. Para efeitos de implementação, os espaços vazios na Tabela 4 são preenchidos com funções de pertinência semelhantes ao impulso, ou seja, um triângulo com a largura da base tendendo a zero.

A Figura 14 mostra a superfície de contorno da relação entre os antecedentes e consequente do FLC do PC fuzzy.

Tabela 4 – Base de regras para a inferência fuzzy.

| $e/\Delta e$ | NB | NM | NS | ZE | PS | PM | PB |
|---|---|---|---|---|---|---|---|
| NB | | | | NB | NB | | |
| NM | NB | | | NB | NB | | |
| NS | NB | | | NM | NM | NM | PM |
| ZE | NB | NM | NS | ZE | PS | PM | PB |
| PS | NM | | PS | PS | PM | | |
| PM | | | | PM | PB | PB | |
| PB | | | PM | PM | PB | | |

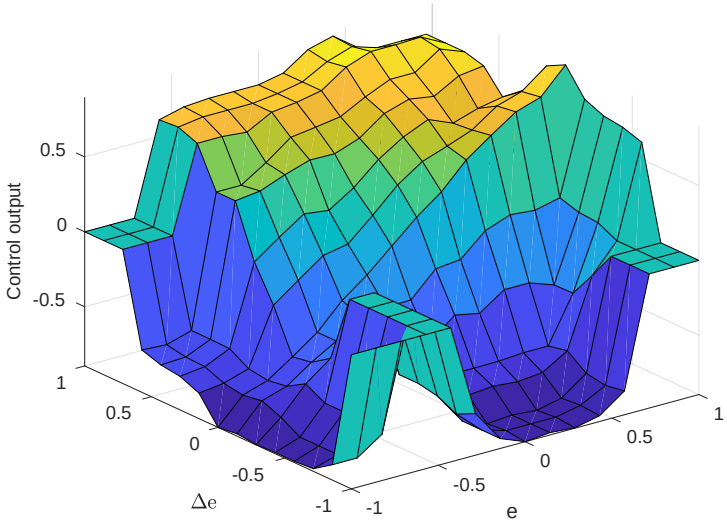


Figura 14 – Superfície de contorno da relação entre os antecedentes e consequentes do FLC.

Para a defuzzificação é utilizado o método do centroide, ou *fuzzy mean*. A figura 15 apresenta um exemplo gráfico desse método na ferramenta Xfuzzy, que será detalhada nas seções seguintes. Observe que apenas as regras 31 e 38 estão ativadas e a função de pertinência resultante é a soma das duas, com a marcação no centroide.

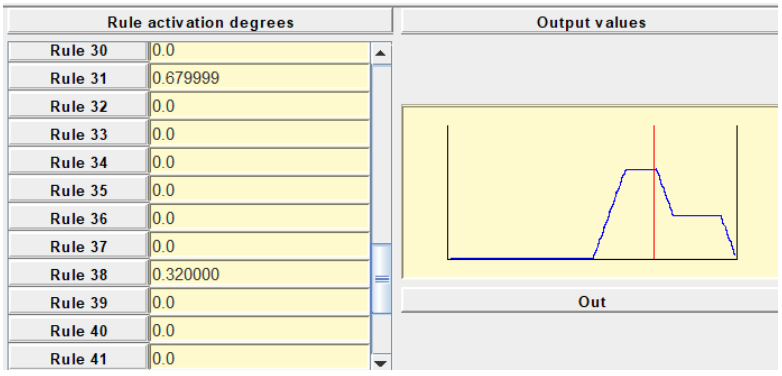


Figura 15 – Exemplo de cálculo de centroide para o PC fuzzy.

## 3.3 Formulação do problema de otimização mono-objetivo

O controlador PID e o PC fuzzy requerem o ajuste dos seus parâmetros, tais como $K_{Pn}$, $K_{In}, K_{Dn}$, e $K_{Sn}, K_{En}$ e $K_{DEn}$ como mostrado na Figura 12. No entanto, como quase não existem modelos matemáticos bem definidos para manipuladores robóticos MIMO, neste trabalho é utilizado algoritmos bioinspirados para ajustá-los, como descrito em (KUMAR, A.; KUMAR, V., 2017). Dois conjuntos de dados biomecânicos da marcha humana foram utilizados para extrair o valor de referência das posições das juntas, a marcha normal (WINTER, 2009) e a condição de subir escadas (LUO et al., 2020). Os parâmetros do manipulador robótico baseado em uma perna humana real são $l_1 = 0{,}314$ m, $l_2 = 0{,}425$ m, $m_1 = 5{,}670$ kg, $m_2 = 2{,}636$, para ambos os conjuntos de dados.

Para modularizar o processo dos ajuste dos parâmetros utilizando o algoritmo PSO (BINGÜL; KARAHAN, 2011). O ajuste do controlador PID e do PC fuzzy foram realizadas de forma independente, cada um com suas particularidades. No entanto, os parâmetros do PSO, determinados a partir de várias experimentações, são compartilhados, tais como os coeficientes cognitivos e sociais $c_1 = 2{,}0$, $c_2 = 2{,}1$ e o fator de inércia $w$, que varia entre 0,9 e 0,1 com uma taxa de decaimento constante em relação ao número de iterações.

### 3.3.1 Formulação do problema do controlador PID

Considerando duas juntas, e um controlador PID para cada uma, obtém-se um problema de otimização de seis variáveis de decisão. O espaço de busca foi restrito a valores entre o intervalo [0;350] para $K_{Pn}$ e entre [0;50] para $K_{Dn}$ e $K_{In}$. O ajuste dos parâmetros é baseado em um ponto inicial e um ponto final e o objetivo é encontrar um conjunto desses valores que minimize o critério determinado pela raiz do erro quadrático médio, ou *Root Mean Squared Erro*r (RMSE), entre o valor desejado $\theta d_n$ e o valor obtido pela planta $\theta_n$, como mostrado na equação 3.6, sendo $k$ o número de amostras (OGATA, 2010).

$$cf_{an} = \sqrt{\frac{1}{k}\sum_{i=1}^{\mathrm{k}}(\theta d_{n,k} - \theta_{n,k})^2} \tag{3.6}$$

Outro critério importante em um sistema de controle é o sobre-impulso ou *overshoot* ($cf_{bn}$), que indica quanto o sinal excede o valor desejado, conforme descrito na equação 3.7. Esse critério é importante para evitar movimentos indesejados do manipulador robótico, como colisões, já que o *overshoot* pode fazer com que o manipulador se mova além do ponto desejado e colida com outros objetos ou pessoas, e danos aos componentes, já que o movimento excessivo do manipulador pode causar tensões excessivas em seus componentes (OGATA, 2010).

$$cf_{bn} = \frac{||max(\theta_n)| - |\theta d_n||}{|\theta d_n|} \tag{3.7}$$

A Figura 16 apresenta um exemplo de representação gráfica das duas funções custo de um sistema de segunda ordem. O RMSE pode ser considerado como a área sob a curva e o *overshoot* é o sobressinal em relação à referência. No caso do manipulador 2-DOF, quanto menores esses valores, melhor é o comportamento do exoesqueleto.

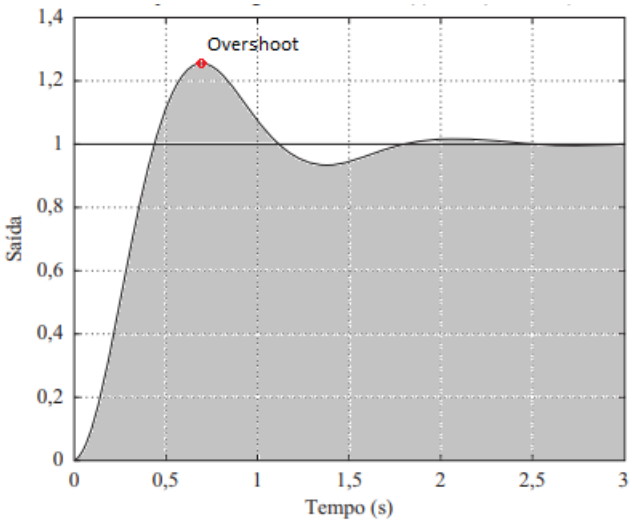


Figura 16 – Análise do RMSE e *overshoot* em um sistema de segunda ordem (OGATA, 2010) modificado.

Combinando os critérios das equações 3.6 e 3.7, a função custo final é dada pela equação (3.8).

$$fitness = min\,(q_1.cf_{a1} + q_2.cf_{b1} + q_3.cf_{a2} + q_4.cf_{b2}) \tag{3.8}$$

Observe que há apenas uma função custo a ser minimizada, pois trata-se de um algoritmo mono-objetivo. Na equação acima, os pesos $q_1 = 0.3$, $q_2 = 0.2$, $q_3 = 0.3$ e $q_4 = 0.2$ são atribuídos à soma dos valores RMSE e os *overshoots* das juntas 1 e 2 do manipulador robótico. Os valores $q_1$ e $q_3$ são maiores para garantir maior importância para a métrica RMSE.

### 3.3.2 Formulação do problema do PC fuzzy

O ajuste do PC fuzzy com o PSO é aplicado somente aos parâmetros $K_{S1}$ e $K_{S2}$, pois obtiveram melhores resultados para este algoritmo. Em vez de partir dos pontos iniciais e finais do conjunto de dados, como no controlador PID, o manipulador robótico foi amostrado ponto a ponto a cada iteração para gerar um conjunto de soluções para todo o sistema. Para

este propósito, foi usado todo o conjunto de dados da marcha humana, consistindo em $\theta d_1$ e $\theta d_2$ obtidos em intervalos de tempo iguais. Assim, no cenário do ajuste dos parâmetros do PC fuzzy, a função custo é a soma do RSME, representada na equação 3.6, das juntas individuais e a equação final é apresentada na equação 3.9. O espaço de busca na busca da minimização da função de custo foi restrito ao intervalo de [0.5;1.5] para ambos os parâmetros $K_{S1}$ e $K_{S2}$.

$$fitness = min\left(cf_{a1} + cf_{a2}\right) \tag{3.9}$$

É importante salientar que para os casos da otimização mono-objetivo e multiobjetivo apresentado nas próximas seções, o processo de ajuste dos parâmetros do PC fuzzy foi feito após o ajuste do controlador PID.

## 3.4 Formulação do problema de otimização multiobjetivo

O processo de ajuste dos parâmetros do controlador PID e do PC fuzzy foi realizado utilizando os algoritmos multiobjetivo NSGA-II e MODE. O processo de ajuste é semelhante ao descrito na seção anterior, exceto que a abordagem possui mais de uma função custo, como mostrado a seguir.

A função custo (F1) a ser minimizada para o ajuste dos parâmetros do controlador PID e do PC fuzzy é apresentada na equação 3.10, que representa a soma dos erros absolutos de posição angular das duas juntas. A função custo (F2), apresentada na equação 3.11, corresponde à soma das variações dos torques de cada junta, onde $tf$ é o tempo total da simulação. O objetivo de F2 é minimizar as variações do torque no manipulador robótico, o que permite aumentar o tempo de vida e a saúde dos atuadores. Esta abordagem é baseada no trabalho de (HULTMANN AYALA; DOS SANTOS COELHO, 2012).

As duas funções custo 3.10 e 3.11 são usadas tanto para o controlador PID como para o PC fuzzy. Um ponto a destacar é que no ajuste dos parâmetros do PC fuzzy são ajustados os três parâmetros $K_{En}$, $K_{DEn}$ e $K_{Sn}$, com o espaço de busca restrito a [0;1] para $K_{En}$ e $K_{DEn}$ e [0;10] para $K_{Sn}$.

$$F_1 = \sum_{t=1}^{tf} |\theta d_1 - \theta_1(t))| + \sum_{t=1}^{tf} |\theta d_2 - \theta_2(t)| \tag{3.10}$$

$$F_2 = \sum_{t=1}^{tf} |\tau_1(t) - \tau_1(t-1))| + \sum_{t=1}^{tf} |\tau_2(t) - \tau_2(t-1))| \tag{3.11}$$

Os parâmetros característicos do algoritmo NSGA-II são taxa de recombinação *crossover* igual a 90% e taxa de mutação de 10% (OSYCZKA, 1985). Já os do algoritmo MODE são o fator de escala igual a 30% e a recombinação *crossover* de 10% (ROBIČ; FILIPIČ, 2005b). Essas configurações foram determinadas por meio das referências citadas e experimentos em execuções preliminares.

## 3.5 Implementação PC fuzzy em hardware embarcado

Todas as abordagens e métodos da seção 3.2 são aplicados para a implementação do PC fuzzy em hardware. A Figura 17 apresenta a arquitetura criada pelo software Xfuzzy 3.5 (FUZZY..., 2018), que gera automaticamente projetos de controladores fuzzy em código VHDL para algumas famílias de FPGAs da Xilinx. Portanto, todos os código VHDL do PC fuzzy proposto foram gerados automaticamente, lembre-se que serão criadas duas arquiteturas idênticas representando as juntas 1 e 2 do manipulador 2-DOF. A arquitetura foi implementada usando uma representação com palavra de 10 bits, na qual as entradas e saídas normalizadas são codificadas em uma resolução em que cada amostra é representada por um número entre 0 e 1023 ($2^{10} - 1$) em binário. Após a geração dos códigos VHDL pela ferramenta Xfuzzy, a síntese lógica e a implementação física do circuito foram realizadas na ferramenta Vivado 2018.3 da Xilinx (VIVADO..., 2019).

A arquitetura proposta baseia-se em mecanismos para reduzir o consumo de recursos e aumentar a velocidade das inferências com o auxílio do paralelismo dos FPGAs. A estratégia do Xfuzzy é avaliar somente a contribuição das regras ativas para restringir a forma e o grau de sobreposição das funções de pertinência de entrada e utilizar somente métodos de defuzzificação simplificados que não requerem a varredura de todos os elementos do universo de discurso das variáveis de saída.

No bloco de fuzzificação (*fuzzification*) há a avaliação das entradas de erro (*error*) e derivada do erro (*error_derivate*), a partir das funções de pertinência (MFC) do tipo triangular apresentadas na Figura 13. As MFCs fornecem os rótulos $L_1$ e $L_2$ e os graus de pertinências $\mu_1$ e $\mu_2$.

A ferramenta Xfuzzy permite realizar inferência do tipo Mamdani, através de uma seleção sequencial das regras ativas, com a ajuda de um circuito composto por um *array* de multiplexadores controlado por um contador e um multiplexador (MUX). Em cada ciclo de relógio, os graus de pertinência ($\mu_i$) dos antecedentes da regra são combinados dentro do operador conectivo *Prod/Max* para calcular o grau de ativação da regra $h_i$. Este operador foi escolhido pela facilidade de implementação em hardware (FUZZY..., 2018). Dessa forma, a combinação dos rótulos antecedentes endereça o local da memória da regra que contém os parâmetros que definem o correspondente consequente $p_i$.

No estágio de saída, um valor *crisp* (valor numérico da saída) é obtido a partir do

processamento dos consequentes das regras com seus diferentes graus de ativação, de acordo com o método de defuzzificação *fuzzy mean*. O bloco de controle (*control*) é responsável por controlar o fluxo de dados entre os blocos mencionados (SÁNCHEZ-SOLANO; BROX, 2014).

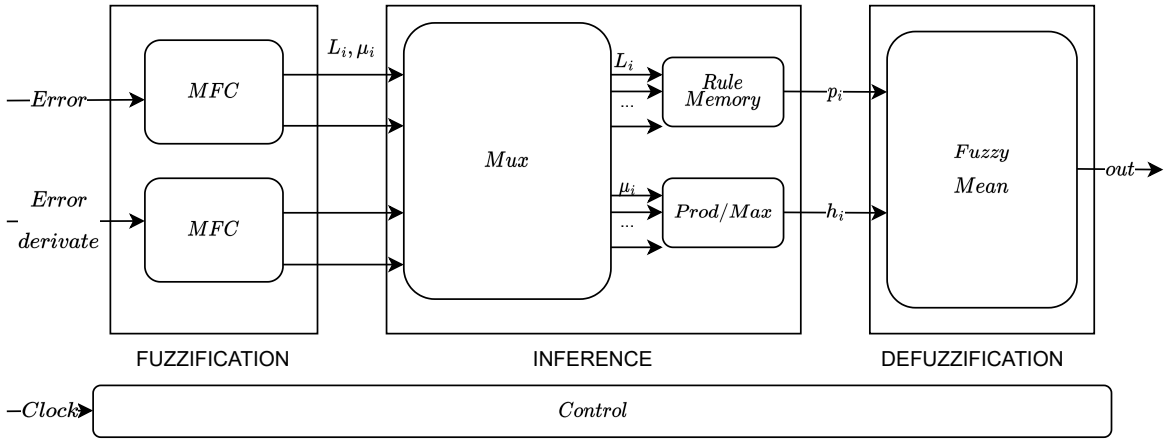


Figura 17 – Arquitetura do pré-compensador usando Xfuzzy (SÁNCHEZ-SOLANO; BROX, 2014).

## 3.6 Implementação do co-projeto hardware/software

Para a implementação do co-projeto hardware/software no kit de desenvolvimento PYNQ-Z2, foi desenvolvida uma arquitetura em que a planta do manipulador robótico e o controlador PID são desenvolvidos na linguagem de programação Python processados no ARM e apenas o PC fuzzy é embarcado no FPGA do SoC Zynq-7000. Outro ponto importante é o desenvolvimento do módulos iguais e paralelos do PC fuzzy em hardware, representando a junta 1 e junta 2, respectivamente. Para efeitos de comparação, o PC Fuzzy também foi desenvolvido na linguagem C, doravante chamado de arquitetura em software

A Figura 18 apresenta a implementação do sistema embarcado baseado em co-projeto hardware-software usando SoC FPGA para implementação do controlador FP-PID.

O bloco Zynq 7000 *Processing System* fornece uma estrutura de gerenciamento de sistema e recursos de processamento para o FPGA, como o *clock* e o *reset*. Também envia e recebe os dados das entradas e saída, de tamanho de 32 bits, do PC fuzzy a partir do *AXI interconnect*.

O *AXI interconnect* é usado para conectar os núcleos ARM e periféricos mapeados no FPGA. Isso permite a comunicação entre a planta e o controlador PID com o encapsulamento do bloco *AXI* fuzzy no FPGA, que engloba o PC fuzzy da arquitetura mostrada na Figura 17.

O protocolo *AXI4 Lite* é uma sub-versão do *AXI interconnect*, projetado para lidar com comunicações simples entre IPs no sistema. É menos complexo, mas ainda oferece recursos importantes para garantir a integridade dos dados e escalabilidade do sistema. A principal diferença é que o *AXI4 Lite* possui apenas um canal de dados bidirecional, enquanto o *AXI*

*interconnect* possui três canais de dados (leitura, escrita e bidirecional). Isso significa que o *AXI4 Lite* é adequado para comunicações simples, como acesso a registradores, enquanto o *AXI interconnect* é adequado para comunicações mais complexas, como acesso a memória (AXI..., 2012).

Na Figura 18, o *AXI4 Lite* é responsável por gerenciar quais entradas do Zynq 7000 serão enviadas ao PC fuzzy, bem como por enviar de volta a saída representada pelo registrador "Reg3". Os registradores de entrada "Reg1" e "Reg2" representam, respectivamente, as entradas *Error* e *Error derivate*. É importante observar que, como o PC fuzzy foi modelado para processar palavras de 10 bits e o *AXI4 Lite* trabalha apenas com registradores de 32 bits, este módulo também gerencia a transformação necessária para compatibilizar os dados.

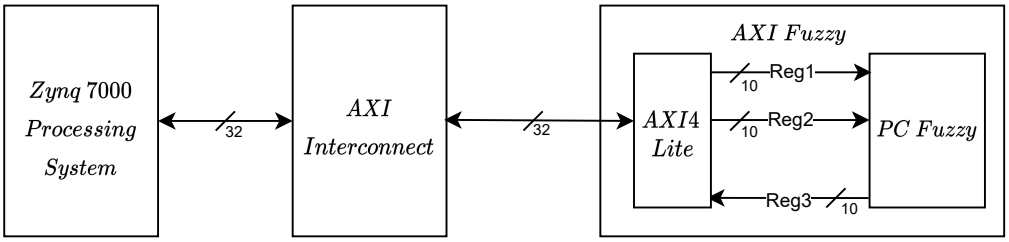


Figura 18 – Arquitetura do co-projeto hardware/software.

# 4 Resultados

## 4.1 Ajuste dos parâmetros do controlador fuzzy PID utilizando algoritmo mono-objetivo PSO

O ajuste dos parâmetros do controlador PID foi realizado utilizando 20 partículas e 10 iterações, definidos a partir de vários testes preliminares. Foram realizados 16 experimentos para validar estatisticamente o processo. O valor inicial das juntas foi $[\theta_1\ \theta_2] = [3.184\ 0.815]$, e o *set-point* foi de [2.993 0.295], ambos dados em radianos.

Os valores estatísticos para a média, mediana, mínimo e desvio padrão da função custo para os 16 experimentos são apresentados na Tabela 5. Os valores dos parâmetros PID para o valor mínimo da função custo são apresentados na Tabela 6. A curva à esquerda na Figura 19 ilustra a curva de convergência do processo de otimização do melhor experimento, em que o eixo X representa o número de iterações e o eixo Y representa o valor da função custo. É possível observar que o algoritmo decresce até atingir o mínimo com o aumento do número de iterações.

Tabela 5 – Estatística para o ajuste dos parâmetros do controlador PID utilizando algoritmo PSO.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| Função custo | 0.460 | 0.461 | 0.460 | 0.001 |

Tabela 6 – Parâmetros do controlador PID para as juntas 1 e 2 utilizando o algoritmo PSO.

| $K_{P1}$ | $K_{D1}$ | $K_{I1}$ | $K_{P2}$ | $K_{D2}$ | $K_{I2}$ |
|---|---|---|---|---|---|
| 338.423 | 49.996 | 45.553 | 336.057 | 44.293 | 44.592 |

O ajuste dos parâmetros do PC fuzzy foi realizado utilizando 20 partículas e 10 iterações, definidos a partir de vários testes preliminares. Esses números são menores se comparado com o ajuste dos parâmetros do controlador PID, devido ao tempo de processamento elevado para cada iteração.

A Tabela 7 apresenta os valores estatísticos, onde $K_{S1} = 1.213$ e $K_{S2} = 1.167$ são os valores encontrados o para valor mínimo da função custo em todos os experimentos. A curva à direita na Figura 19 ilustra a curva de convergência.

Tabela 7 – Estatística para o ajuste dos parâmetros do PC fuzzy utilizando o algoritmo PSO.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| Função custo | 0.078 | 0.079 | 0.078 | 0.001 |

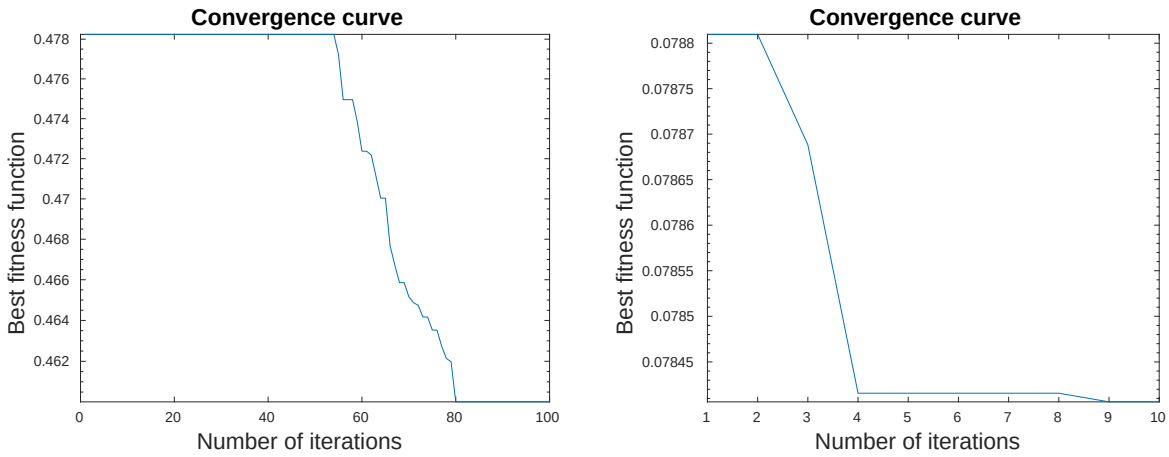


Figura 19 – Curva de convergência para o ajuste dos parâmetros. Esquerda: PID; Direita: PC fuzzy.

Após o ajuste foram obtidas as trajetórias de cada junta com e sem pre-compensação fuzzy, utilizando como referência as bases de dados de marcha humana em condição normal e subindo escadas. A Figura 20 mostra as trajetórias para $\theta_1$ e $\theta_2$ utilizando a marcha humana em condição normal. Em azul apresenta-se a referência, em laranja a trajetória sem PC fuzzy, e em verde a trajetória com pré-compensação fuzzy.

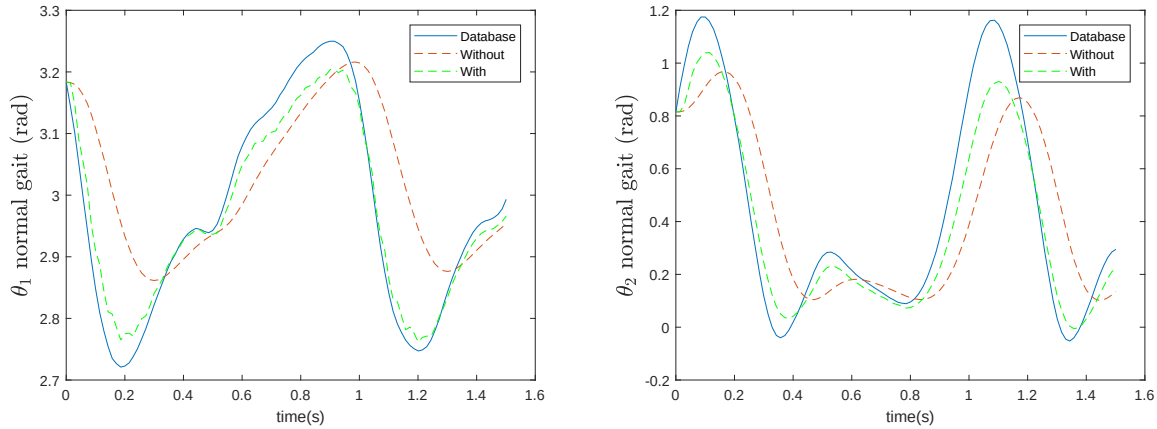


Figura 20 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo PSO. Esquerda: $\theta_1$; Direita: $\theta_2$.

Para validar os parâmetros obtidos para os controladores, foi utilizada a base de dados da marcha humana subindo escadas, ou do inglês *upstairs*, como apresentado na Figura 21.

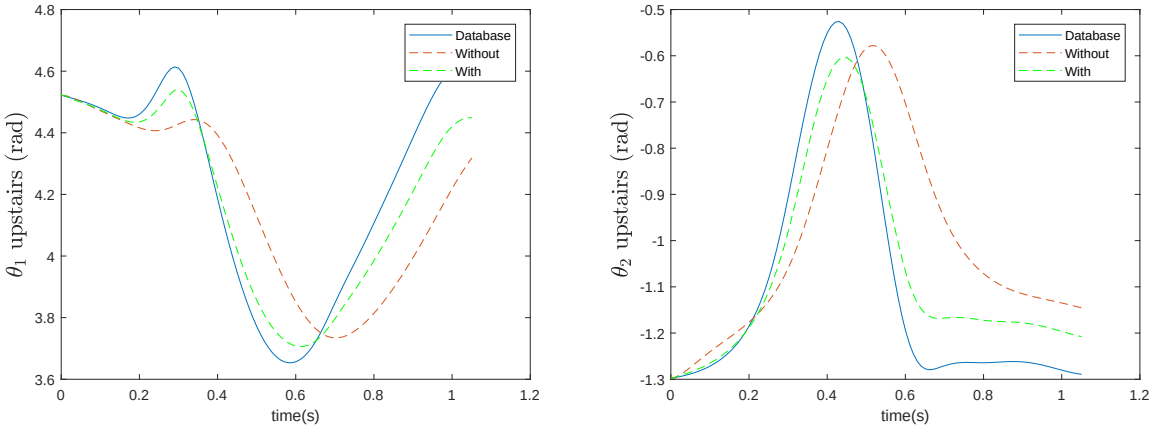


Figura 21 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo PSO. Esquerda: $\theta_1$; Direita: $\theta_2$.

Apesar da melhora na trajetória para ambas as juntas, constatou-se que os torques dos atuadores do controlador com o PC fuzzy oscilaram várias vezes ao tentar atingir o *set point* de cada ponto para a condição da marcha normal, conforme mostra a Figura 22. Este comportamento pode ser explicado pelas mudanças bruscas da superfície de contorno (vide Figura 14) quando as regras de inferência do PC fuzzy são ativadas ou desativadas. Esse fenômeno, também visto em (KUMAR, A.; KUMAR, V., 2017), pode prejudicar a vida útil ou a integridade dos atuadores do manipulador robótico e uma possível solução a este problema será discutida nas seções subsequentes.

Para a condição da marcha subindo escadas, os torques dos atuadores são apresentados na Figura 23. Observa-se que a oscilação do torque não é tão alta, pois $\theta_1$ e $\theta_2$ não mudam de direção rapidamente como na marcha anterior.

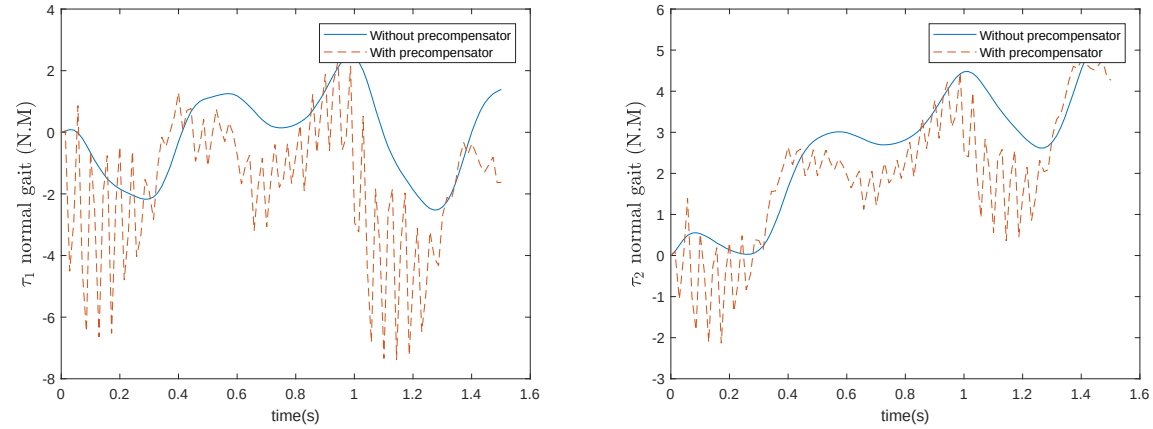


Figura 22 – Curva de torque dos atuadores com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo PSO. Esquerda: $\tau_1$; Direita: $\tau_2$.

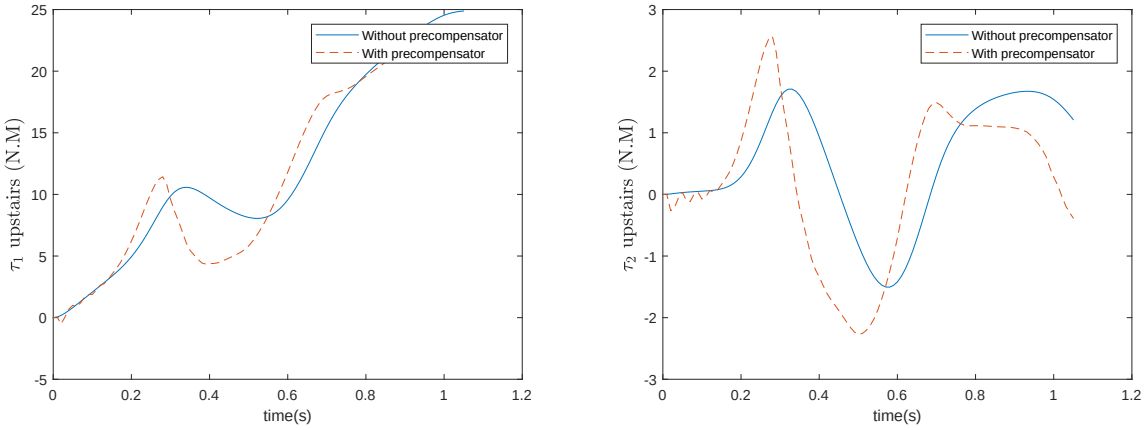


Figura 23 – Curva de torque dos atuadores do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo PSO. Esquerda: $\tau_1$; Direita: $\tau_2$.

## 4.2 Ajuste dos parâmetros do controlador fuzzy PID utilizando algoritmo multiobjetivo NSGA-II

O ajuste dos parâmetros do controlador PID utilizando o algoritmo de otimização multiobjetivo NSGA-II consistiu em utilizar 40 indivíduos com 200 gerações em cada experimento (HULTMANN AYALA; DOS SANTOS COELHO, 2012). Foram realizados 16 experimentos para validar estatisticamente o processo. O valor inicial das juntas foi $[\theta_1\ \theta_2]$ = [3.184 0.815], e o *set-point* foi de [2.993 0.295], ambos dados em radianos.

As abordagens para determinar os valores dos parâmetros PID são apresentadas na primeira coluna da Tabela 8. Para um melhor entendimento, vamos seguir um exemplo: *"Experimentos em centro de Pareto"* significa usar o centro da frente de Pareto da última iteração e registrar os mínimos da função custo (F1) e (F2). Isso é feito para cada experimento, totalizando 16 valores. A primeira curva da Figura 24 apresenta as três possibilidades de escolha: *best F1*, *Pareto front center* e *best F2*. Após armazenar esses resultados, é decidido qual mínimo das duas funções custo será escolhido para a seleção dos parâmetros, como é representado pela primeira sentença *"Melhor F1"* ou *"Melhor F2"* da primeira coluna da Tabela 8. É importante ressaltar que é necessário adotar um critério e equilibrar as funções custo F1 e F2 para escolher a melhor abordagem. Essa escolha é pessoal e deve ser baseada nas necessidades do projeto. A seguir, serão apresentados os critérios adotado neste trabalho.

O método de experimentação consistiu em executar todas as abordagens possíveis e verificar qual é a melhor, a qual é destacada em negrito. As terceira e quarta colunas apresentam o RMSE de $\theta_1$ e $\theta_2$ do manipulador robótico em relação à referência da base de dados. As quarta e quinta colunas apresentam os valores da funções custo F1 e F2.

Para o ajuste dos parâmetros do controlador PID, foi escolhida a abordagem *"Melhor F1/Experimentos em F1"* e a frente de Pareto do melhor experimento é apresentada na curva

da esquerda da Figura 24. Essa escolha de determinar o valor mínimo a partir de F1 segue a característica do ajuste dos parâmetros do controlador PID, que busca minimizar o erro de posição do manipulador robótico.

Tabela 8 – Valores de RMSE de $\theta_1$ e $\theta_2$ e o mínimo de F1 ou F2 para as diferentes abordagens do ajuste dos parâmetros do controlador PID utilizando o algoritmo NSGA-II.

| Abordagem | RMSE $\theta_1$ | RMSE $\theta_2$ | F1 | F2 |
|---|---|---|---|---|
| Melhor F1/Experimentos em centro de Pareto | 0.148 | 0.344 | 22.150 | 161.033 |
| Melhor F2/Experimentos em centro de Pareto | 0.164 | 0.398 | 24.223 | 68.561 |
| ***Melhor F1/ Experimentos em F1*** | ***0.096*** | ***0.257*** | ***19.301*** | ***463.341*** |
| Melhor F1/Experimentos em F2 | 0.113 | 0.290 | 20.307 | 313.897 |
| Melhor F2/Experimentos em F1 | 0.316 | 0.404 | 29.066 | 36.699 |
| Melhor F2/Experimentos em F2 | 0.324 | 0.522 | 38.609 | 5.556 |

Os valores estatísticos de F1 para a abordagem escolhida são apresentados na Tabela 9, e os parâmetros PID são mostrados na Tabela 10.

Tabela 9 – Estatística para o ajuste dos parâmetros do controlador PID utilizando o algoritmo NSGA-II.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| F1 | 19.737 | 19.708 | **19.301** | 0.264 |

Tabela 10 – Parâmetros do controlador PID para as juntas 1 e 2 utilizando o algoritmo NSGA-II.

| $K_{P1}$ | $K_{D1}$ | $K_{I1}$ | $K_{P2}$ | $K_{D2}$ | $K_{I2}$ |
|---|---|---|---|---|---|
| 350.000 | 27.270 | 21.583 | 250.572 | 25.659 | 42.477 |

Para o ajuste dos parâmetros do PC fuzzy, a abordagem *"Melhor F2/Experimentos em F1"*, conforme mostrado na Tabela 11. O centro de Pareto do melhor experimento é apresentada na segunda curva da Figura 24. Essa escolha sugere que o algoritmo buscou um equilíbrio entre as duas funções custo, sendo que F2 é característica do ajuste dos parâmetros do PC fuzzy, que busca suavizar o torque do atuador do manipulador robótico, resolvendo o problema de oscilação dos torques dos atuadores do manipulador robótico, quando usado o algoritmo mono-objetivo PSO, apresentado na Figura 22.

Tabela 11 – Valores de RMSE de $\theta_1$ e $\theta_2$ e o mínimo de F1 ou F2 para as diferentes abordagens do ajuste dos parâmetros do PC fuzzy utilizando o algoritmo NSGA-II.

| Abordagem | RMSE $\theta_1$ | RMSE $\theta_2$ | F1 | F2 |
|---|---|---|---|---|
| Melhor F1/Experimentos em centro de Pareto | 0.026 | 0.043 | 5.523 | 81.861 |
| Melhor F2/Experimentos em centro de Pareto | 0.059 | 0.106 | 13.856 | 51.663 |
| Melhor F1/ Experimentos em F1 | 0.015 | 0.029 | 3.353 | 96.291 |
| Melhor F1/Experimentos em F2 | 0.015 | 0.029 | 3.353 | 96.291 |
| ***Melhor F2/Experimentos em F1*** | ***0.049*** | ***0.049*** | ***8.264*** | ***59.016*** |
| Melhor F2/Experimentos em F2 | 0.081 | 0.234 | 26.066 | 30.209 |

Os valores estatísticos de F1 para a abordagem escolhida são apresentados na Tabela 12, e os parâmetros do PC fuzzy são mostrados na Tabela 13.

Tabela 12 – Estatística para o ajuste dos parâmetros do PC fuzzy utilizando o algoritmo NSGA-II.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| F1 | 23.572 | 20.504 | **8.264** | 6.573 |

Tabela 13 – Parâmetros do PC fuzzy para as juntas 1 e 2 utilizando o algoritmo NSGA-II.

| $K_{E1}$ | $K_{D1}$ | $K_{S1}$ | $K_{E2}$ | $K_{D2}$ | $K_{S2}$ |
|---|---|---|---|---|---|
| 0.000 | 0.038 | 1.869 | 0.978 | 0.017 | 5.375 |

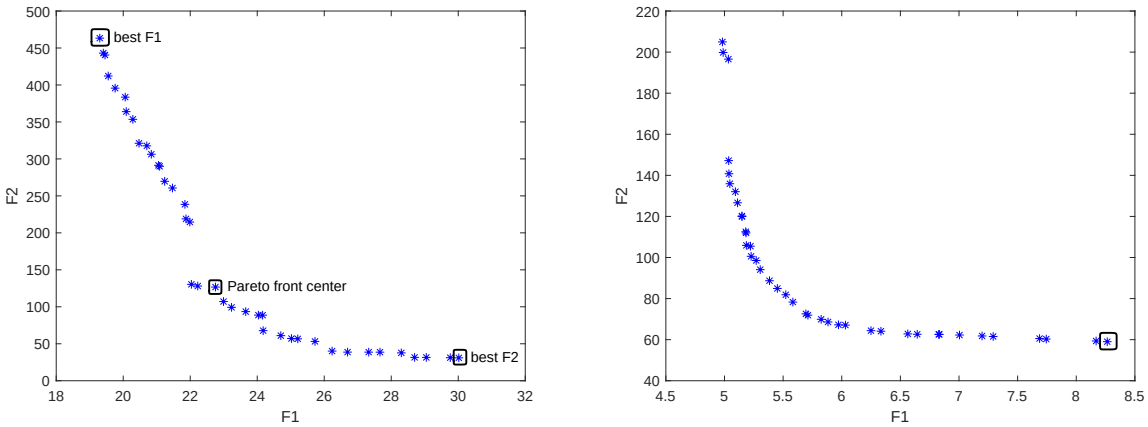


Figura 24 – Frente de Pareto do ajuste dos parâmetros do controlador PID e PC fuzzy utilizando o algoritmo NSGA-II. Esquerda: Controlador PID; Direita: PC fuzzy.

Após o ajuste, foram obtidas as trajetórias de cada junta com e sem PC fuzzy, utilizando como referência as bases de dados de marcha humana em condição normal e subindo escadas. A Figura 25 mostra as trajetórias de $\theta_1$ e $\theta_2$ utilizando a marcha humana em condição normal.

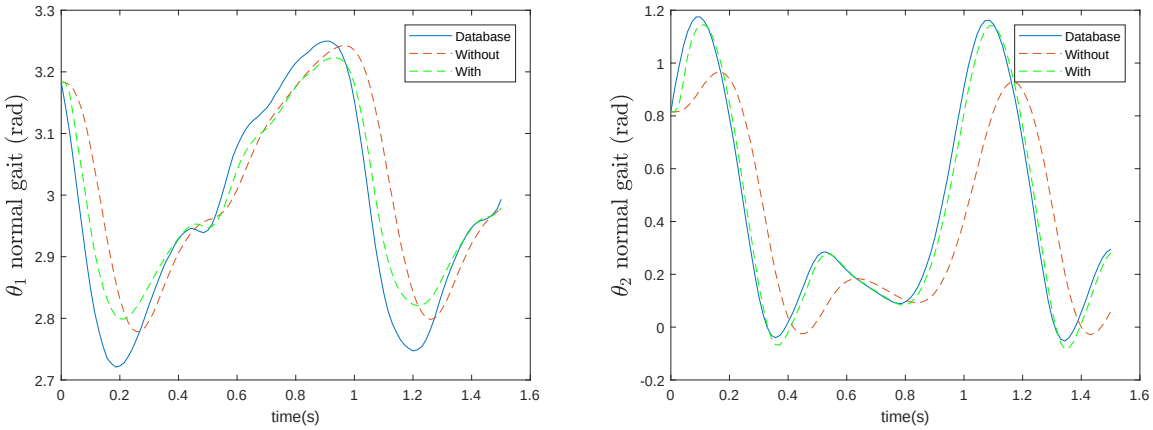


Figura 25 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo NSGA-II. Esquerda: $\theta_1$; Direita: $\theta_2$.

Para validar os parâmetros obtidos para os controladores, foi utilizada a base de dados de marcha humana subindo escadas, conforme apresentado na Figura 26.

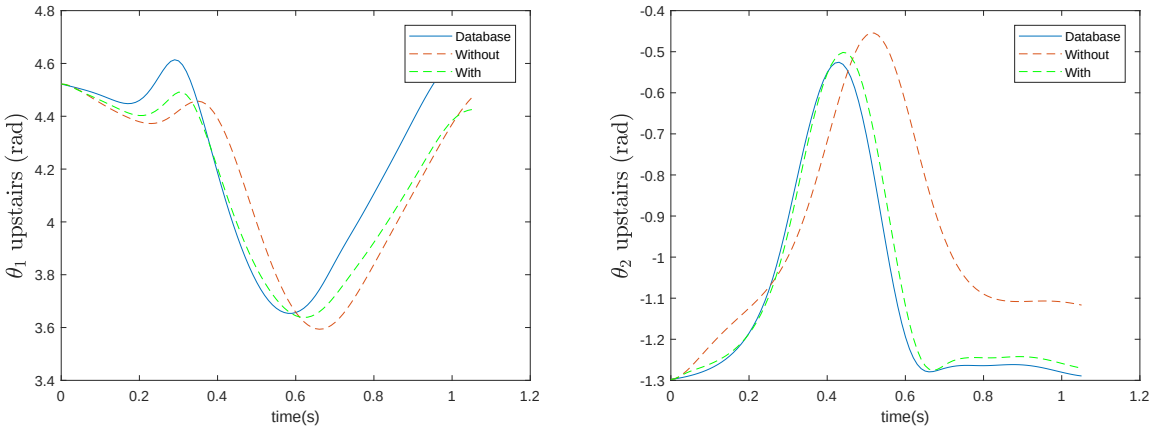


Figura 26 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo NSGA-II. Esquerda: $\theta_1$; Direita: $\theta_2$.

Além de melhorar a trajetória, constatou-se, com a adição da função custo F2, uma significativa suavização dos torques dos atuadores de ambas as juntas, problema discutido na seção anterior. Essa é a justificativa para a escolha de algoritmos multiobjetivos, como também pode ser visto no trabalho de (HULTMANN AYALA; DOS SANTOS COELHO, 2012). As Figuras 27 e 28 apresentam esses resultados para as duas condições de marcha humana.

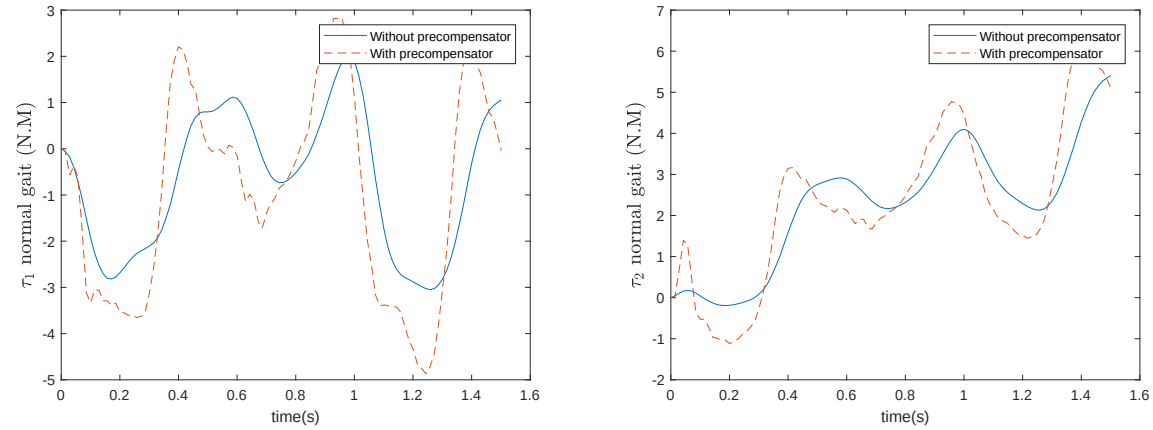


Figura 27 – Curva de torque dos atuadores com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo NSGA-II. Esquerda: $\tau_1$; Direita: $\tau_2$.

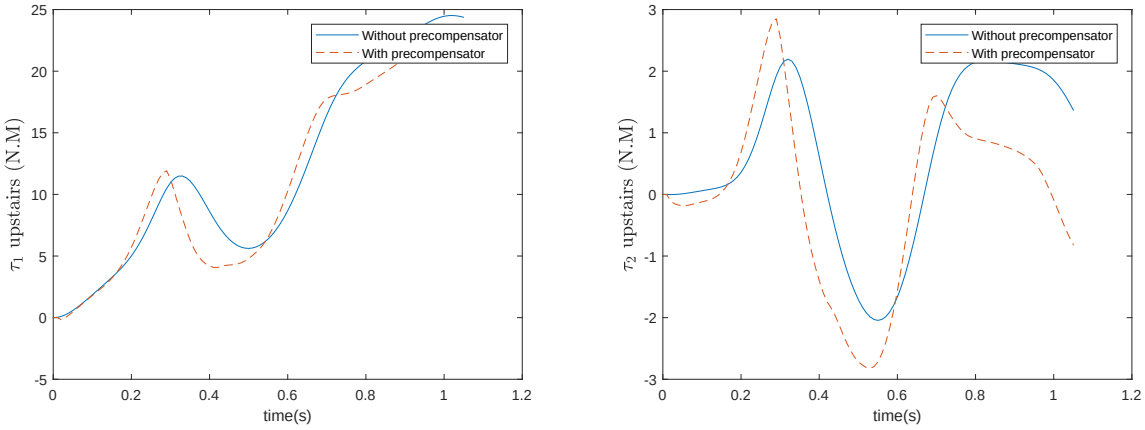


Figura 28 – Curva de torque dos atuadores do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo NSGA-II. Esquerda: $\tau_1$; Direita: $\tau_2$.

## 4.3 Ajuste dos parâmetros do controlador fuzzy PID utilizando algoritmo multiobjetivo MODE

Todos os métodos de análise apresentados na seção anterior foram utilizados para o ajuste dos parâmetros do controlador PID e PC fuzzy usando o algoritmo multiobjetivo MODE.

Semelhante ao algoritmo NSGA-II, a melhor abordagem utilizada foi *"Melhor F1/Experimentos em F1"* para o ajuste dos parâmetros do controlador PID utilizando o algoritmo MODE, como mostra a Tabela 14. A frente de Pareto do melhor experimento é apresentada na primeira curva da Figura 29.

Tabela 14 – Valores de RMSE de $\theta_1$ e $\theta_2$ e o mínimo de F1 ou F2 para as diferentes abordagens do ajuste dos parâmetros do controlador PID utilizando o algoritmo MODE.

| Abordagem | RMSE $\theta_1$ | RMSE $\theta_1$ | F1 | F2 |
|---|---|---|---|---|
| Melhor F1/Experimentos em centro de Pareto | 0.149 | 0.339 | 22.346 | 165.606 |
| Melhor F2/Experimentos em centro de Pareto | 0.180 | 0.394 | 23.400 | 85.706 |
| ***Melhor F1/ Experimentos em F1*** | ***0.096*** | ***0.234*** | ***19.158*** | ***540.692*** |
| Melhor F1/Experimentos em F2 | 0.109 | 0.295 | 19.950 | 319.395 |
| Melhor F2/Experimentos em F1 | 0.308 | 0.418 | 28.335 | 26.647 |
| Melhor F2/Experimentos em F2 | 0.310 | 0.522 | 31.372 | 23.701 |

Os valores estatísticos de F1 para a abordagem escolhida são apresentados na Tabela 15, e os parâmetros PID são mostrados na Tabela 16.

Tabela 15 – Estatística para o ajuste dos parâmetros do controlador PID utilizando o algoritmo MODE.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| F1 | 19.559 | 19.619 | **19.158** | 0.231 |

Tabela 16 – Parâmetros do controlador PID para as juntas 1 e 2 utilizando o algoritmo MODE.

| $K_{P1}$ | $K_{D1}$ | $K_{I1}$ | $K_{P2}$ | $K_{D2}$ | $K_{I2}$ |
|---|---|---|---|---|---|
| 349.780 | 27.278 | 43.628 | 347.784 | 31.581 | 47.971 |

Para a otimização do controlador fuzzy PC, a abordagem *"Melhor F1/Experimentos no centro de Pareto"* foi selecionada, conforme mostrado na Tabela 17. A frente de Pareto do melhor experimento é apresentada na curva da direita da Figura 29. Diferentemente do algoritmo NSGA-II, os valores no centro de Pareto com o melhor F1 encontraram um melhor equilíbrio entre as funções custo F1 e F2.

Outro resultado importante é a abordagem *"F2/Experimentos no centro de Pareto"*, no qual há uma diminuição na função custo F1, mas há uma melhoria na suavização do torque dos atuadores com a otimização de F2 em comparação com a abordagem selecionada.

Tabela 17 – Valores de RMSE de $\theta_1$ e $\theta_2$ e o mínimo de F1 ou F2 para as diferentes abordagens do ajuste dos parâmetros do PC fuzzy utilizando o algoritmo MODE.

| Abordagem | RMSE Theta1 | RMSE Theta2 | F1 | F2 |
|---|---|---|---|---|
| ***Melhor F1/Experimentos em centro de Pareto*** | ***0.044*** | ***0.050*** | ***7.989*** | ***59.140*** |
| Melhor F2/Experimentos em centro de Pareto | 0.043 | 0.095 | 11.339 | 49.320 |
| Melhor F1/ Experimentos em F1 | 0.013 | 0.030 | 3.303 | 108.267 |
| Melhor F1/Experimentos em F2 | 0.013 | 0.033 | 3.569 | 84.134 |
| Melhor F2/Experimentos em F1 | 0.071 | 0.196 | 22.086 | 31.950 |
| Melhor F2/Experimentos em F2 | 0.075 | 0.219 | 24.305 | 31.778 |

Os valores estatísticos de F1 para a abordagem escolhida são apresentados na Tabela 18, e os parâmetros do PC fuzzy são mostrados na Tabela 19.

Tabela 18 – Estatística para o ajuste dos parâmetros do PC fuzzy utilizando o algoritmo MODE.

| Estatística | Média | Mediana | Mínimo | Desvio padrão |
|---|---|---|---|---|
| F1 | 10.553 | 10.177 | **7.989** | 1.031 |

Tabela 19 – Parâmetros do PC fuzzy para as juntas 1 e 2 utilizando o algoritmo MODE.

| $K_{E1}$ | $K_{D1}$ | $K_{S1}$ | $K_{E2}$ | $K_{D2}$ | $K_{S2}$ |
|---|---|---|---|---|---|
| 0.000 | 0.578 | 0.033 | 0.010 | 2.603 | 6.787 |

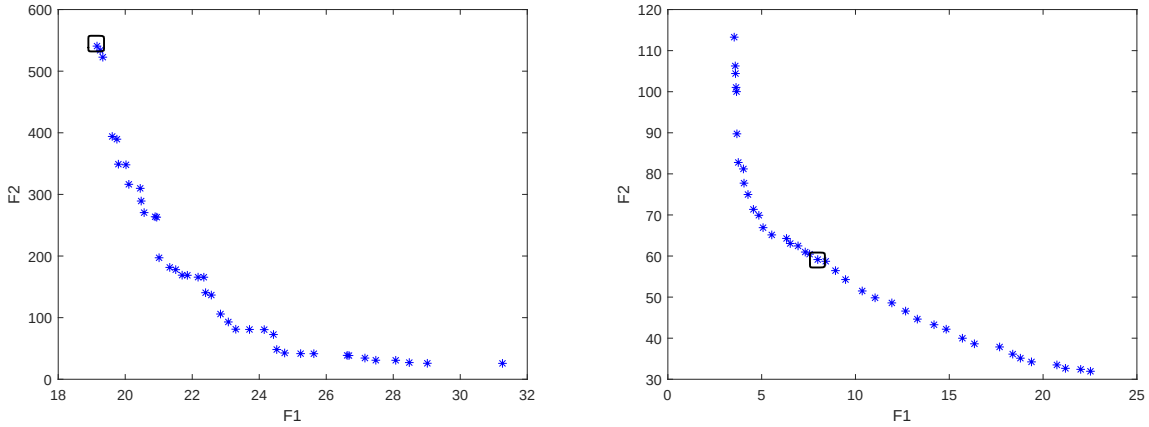


Figura 29 – Frente de Pareto do ajuste dos parâmetros do controlador PID e PC fuzzy utilizando o algoritmo MODE. Esquerda: Controlador PID; Direita: PC fuzzy.

Após o ajuste, foram obtidas as trajetórias de cada junta com e sem PC fuzzy, utilizando como referência as bases de dados de marcha humana em condição normal e subindo escadas. A Figura 30 mostra as trajetórias de $\theta_1$ e $\theta_2$ utilizando a marcha humana em condição normal.

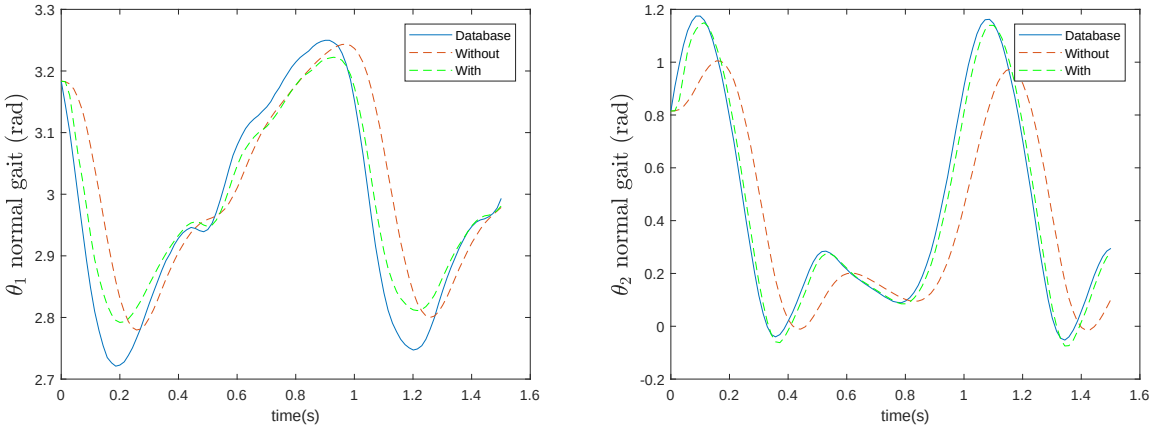


Figura 30 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo MODE. Esquerda: $\theta_1$; Direita: $\theta_2$.

Para validar os parâmetros obtidos para os controladores, foi utilizada a base de dados de marcha humana subindo escadas, conforme apresentado na Figura 31.

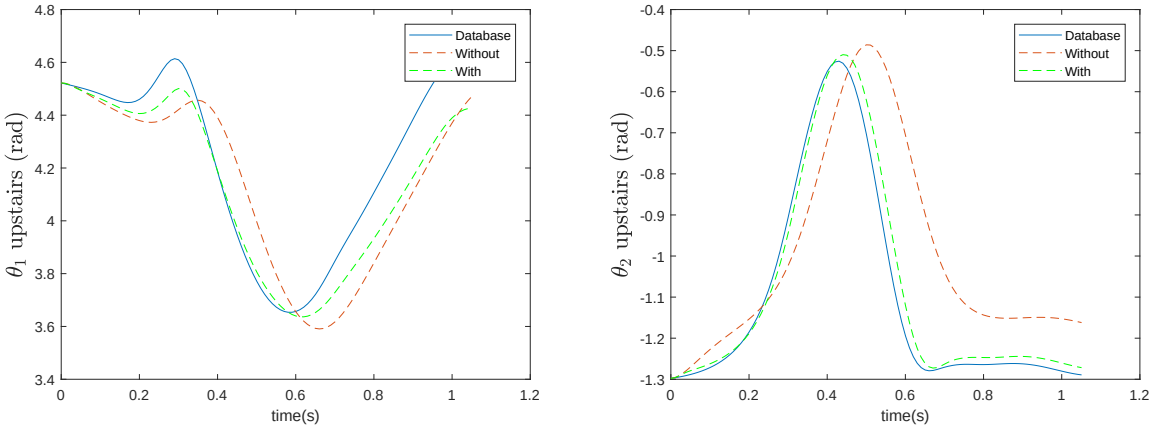


Figura 31 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo MODE. Esquerda: $\theta_1$; Direita: $\theta_2$.

Da mesma forma que o algoritmo NSGA-II, nota-se uma melhoria na suavização do torque dos atuadores das juntas do manipulador robótico. As Figuras 32 e 33 mostram esses resultados para as duas condições de marcha humana.

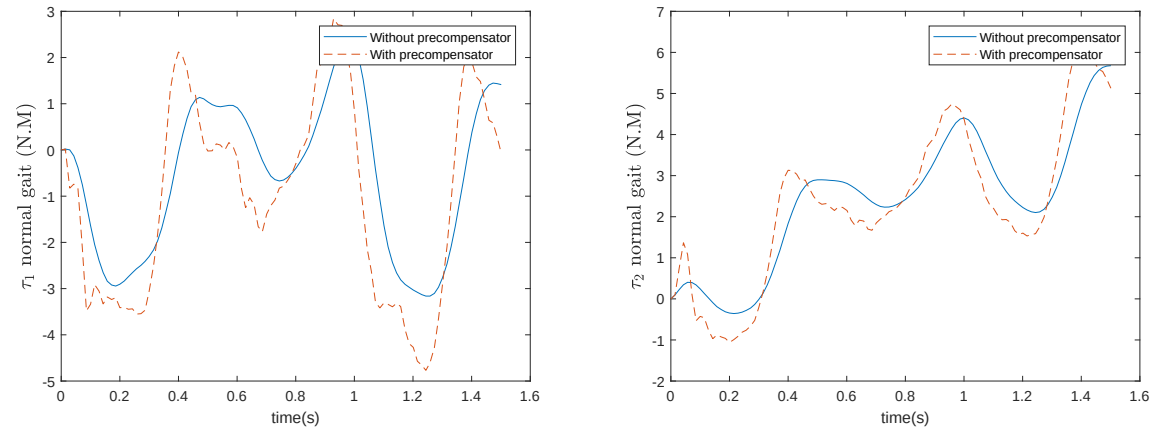


Figura 32 – Curva de torque dos atuadores com e sem PC fuzzy para a marcha humana em condição normal utilizando o algoritmo MODE. Esquerda: $\tau_1$; Direita: $\tau_2$.

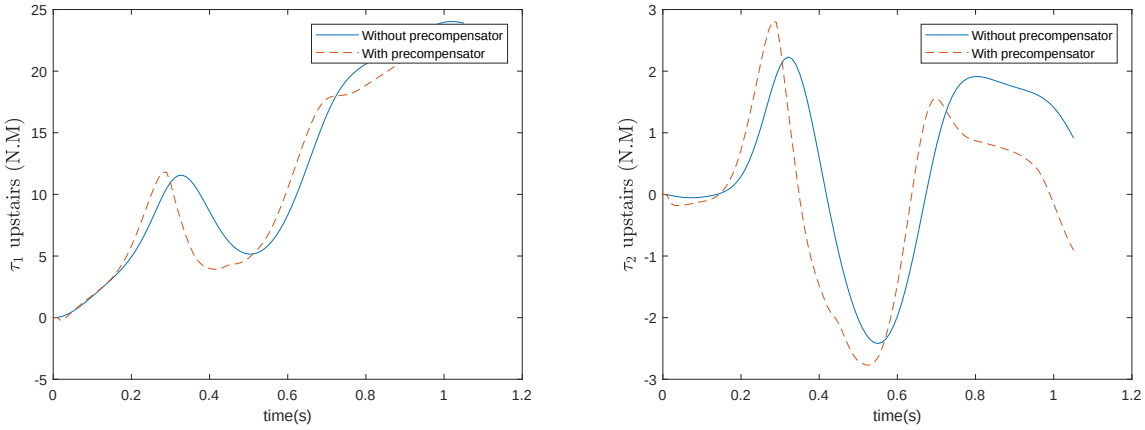


Figura 33 – Curva de torque dos atuadores do manipulador com e sem PC fuzzy para a marcha humana subindo escadas utilizando o algoritmo MODE. Esquerda: $\tau_1$; Direita: $\tau_2$.

## 4.4 Comparação entre os algoritmos

A Tabela 20 apresenta o RMSE de cada junta quando comparado com a referência de cada base de dados, bem como a porcentagem de melhoria quando usado o PC fuzzy para cada algoritmo. À direita, para o PSO, observa-se que a melhoria foi superior a 50% em todas as condições, alcançando um valor máximo de 73,846% para a junta 1 da marcha normal.

No centro, para o NSGA-II, a melhoria foi superior a 30% em todas as condições, alcançando um valor máximo de 83,75% para a junta 2 da marcha humana subindo escadas. À esquerda, para o MODE, é notável que a melhoria foi superior a 35% em todas as condições, alcançando um valor máximo de 81,095% para a junta 2 da marcha humana subindo escadas.

| Junta/RMSE | Sem | Com | Melhoria % |
|---|---|---|---|
| Normal gait $\theta_1$ | 0.130 | 0.034 | **73.846** |
| Normal gait $\theta_2$ | 0.271 | 0.113 | 50.302 |
| Upstairs $\theta_1$ | 0.245 | 0.097 | 60.408 |
| Upstairs $\theta_2$ | 0.222 | 0.079 | 64.414 |

| Junta/RMSE | Sem | Com | Melhoria % |
|---|---|---|---|
| Normal gait $\theta_1$ | 0.096 | 0.049 | 48.958 |
| Normal gait $\theta_2$ | 0.257 | 0.049 | 80.934 |
| Upstairs $\theta_1$ | 0.190 | 0.128 | 32.632 |
| Upstairs $\theta_2$ | 0.240 | 0.039 | **83.750** |

| Junta/RMSE | Sem | Com | Melhoria % |
|---|---|---|---|
| Normal gait $\theta_1$ | 0.096 | 0.049 | 48.958 |
| Normal gait $\theta_2$ | 0.257 | 0.049 | 80.934 |
| Upstairs $\theta_1$ | 0.190 | 0.128 | 32.632 |
| Upstairs $\theta_2$ | 0.240 | 0.039 | **83.750** |

Tabela 20 – RMSE com e sem PC fuzzy. Esquerda: PSO; Centro: NSGA-II; Direita: MODE

Além da verificação dos RMSE de cada algoritmo, testes não paramétricos são usados para comparar a performance entre os algoritmos multiobjetivos NSGA-II e MODE no ajuste dos parâmetros do controlador PID e do PC fuzzy, utilizando as amostras dos mínimos obtidos em 16 experimentos na abordagem escolhida, conforme apresentado nas seções 4.2 e 4.3. O algoritmo mono-objetivo PSO não está incluído na comparação, pois possui abordagens, quantidade de iterações e número de partículas diferentes dos outros. Todos os testes têm nível de confiança de 95%.

O primeiro teste executado é Kolmogorov-Smirnov (K-S) que é um teste estatístico não paramétrico, ou seja, não requer nenhuma suposição sobre a forma da distribuição dos dados. Ele é amplamente utilizado para verificar se uma amostra de dados é consistente com

uma distribuição normal. A hipótese nula $h = 0$ e a significância $p > 0.05$ mostra que os dados amostrados possuem distribuição normal, caso $h = 1$ e $p < 0.05$ pode-se considerar uma distribuição não normal. A Tabela 21 conclui, com $h = 1$ e $p \sim 0$ (aproximadamente zero), que as amostras do ajuste dos parâmetros pelos algoritmos NSGA-II e MODE para o controlador PID e PC fuzzy não seguem uma distribuição normal (MONTGOMERY; RUNGER; HUBLE, 2011).

Tabela 21 – Teste não paramétrico Kolmogorov-Smirnov (K-S).

| Parâmetro | Controlador PID | PC fuzzy |
|---|---|---|
| h | 1 | 1 |
| p | ∼0 | ∼0 |

O segundo teste executado é o Wilcoxon ou *ranksum*, um teste não-paramétrico para comparar duas amostras independentes de dados, frequentemente utilizado quando as amostras de dados não seguem uma distribuição normal. A hipótese nula é que as duas amostras provêm da mesma distribuição, Se isso é rejeitado, conclui-se que existe diferença entre as duas amostras. Após isso, compara-se os valores das medidas de posição, como a mediana, entre as duas amostras para verificar qual é melhor (MONTGOMERY; RUNGER; HUBLE, 2011).

A Tabela 22 conclui que para o ajuste dos parâmetros do controlador PID, não houve diferença estatística significativa entre os dois algoritmos, mas para o PC fuzzy, com $h = 1$, o ajuste dos parâmetros pelos os algortimos NSGA-II e MODE são estatisticamente diferentes. A partir das Tabelas 12 e 18, é possível verificar que a mediana do NSGA-II é igual a 20.504, enquanto a do MODE é igual a 10.177, o que torna este último algoritmo considerado melhor.

Tabela 22 – Teste não paramétrico Wilcoxon.

| Parâmetro | Controlador PID | PC fuzzy |
|---|---|---|
| h | 0 | 1 |
| p | 0.440 | ∼0 |

## 4.5 Caracterização da arquitetura do pré-compensador em hardware utilizando Xfuzzy

Esta seção apresenta a caracterização em hardware do PC fuzzy, usando a arquitetura mostrada na Figura 18. A arquitetura foi mapeada corretamente para o chip Zynq xc7z020-1CLG400C SoC FPGA Artix-7 dispositivo XC7Z020 com uma frequência de 73.923 MHz e não houve violações de *timing* na implementação do projeto.

A Tabela 23 mostra o consumo dos recursos da arquitetura geral. O Módulo *Top module* corresponde à integração entre o os módulos Zynq 7000 *processing System*, o barra-

mento *AXI4-Lite* e PC fuzzy. De modo geral, em relação à totalidade dos recursos do FPGA, há pouco consumo do sistema integrado, com menos de 1% para LUTs e FFs e menos de 2% para DSPs e não há consumo de blocos BRAM em nenhum dos módulos.

A Tabela 24 discrimina o consumo de recursos do módulo do PC fuzzy. Em relação às LUTs, há um maior consumo para o MFC, seguido do *Controller*, MUX e *Connective operator*. Observe que a *Rule memory* e o bloco *Defuzzification* não utilizam LUTs. O bloco MUX e o *Connective operator* não possuem FFs por usarem apenas lógica combinacional. Adicionalmente, o operador possui um DSP para a operacão de multiplicação. O MFC também integra memória ROM para a fuzzificação.

Tabela 23 – Consumo dos recursos da arquitetura geral no FPGA.

| Módulo | LUTs<br>Disponível: 53200 | FFs<br>Disponível: 106400 | Blocos DSP<br>Disponível: 220 |
|---|---|---|---|
| *Top Module* | 510 (0.959%) | 611 (0.574%) | 4 (1.818%) |
| Outros módulos | 351 (0.660%) | 433 (0.407%) | 0 |
| PC Fuzzy + *AXI4 Lite* | 159 (0.299%) | 178 (0.167%) | 4 (1.818%) |
| PC Fuzzy | 109 (0.205%) | 72 (0.068%) | 2 (0.909%) |

Tabela 24 – Consumo dos recursos do PC fuzzy no FPGA.

| Módulo | LUTs<br>Disponível: 109 | FFs<br>Disponível: 72 | Blocos DSP<br>Disponível: 2 |
|---|---|---|---|
| *Controller* | 37 (33.955%) | 4 (5.556%) | 0 |
| MFC | 41 ( 37.615%) | 58 (80.556%) | 0 |
| MUX | 20 (18.349) | 0 | 0 |
| *Rule memory* | 0 | 0 | 0 |
| *Connective operator* | 11 (10.091) | 0 | 1 (50.000%) |
| *Defuzzification* | 0 | 10 (13.889%) | 1 (50.000%) |

A Figura 34 mostra o *layout* da implementação física da arquitetura geral do PC fuzzy com encapsulamento AXI4-Lite no SoC FPGA. O sistema de processamento é representado pela cor verde, o *AXI4-Lite* em amarelo e o PC fuzzy em rosa. A Figura 35 mostra apenas os sistemas do PC fuzzy, sendo que o *Controller* é representado em amarelo, o *Defuzzification* em verde-claro, o MFC em rosa, *Connective operator* em vermelho e o MUX em marrom.

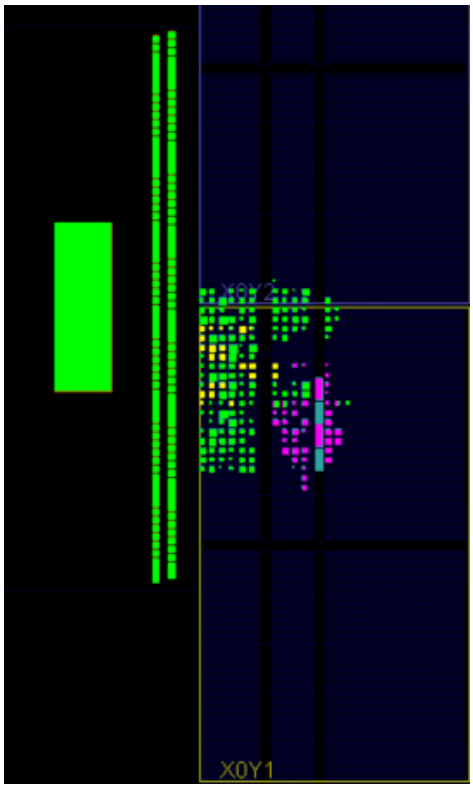


Figura 34 – *Layout* da arquitetura geral. Sistema de processamento em verde, parte do *AXI4 Lite* em amarelo e o PC fuzzy em rosa.

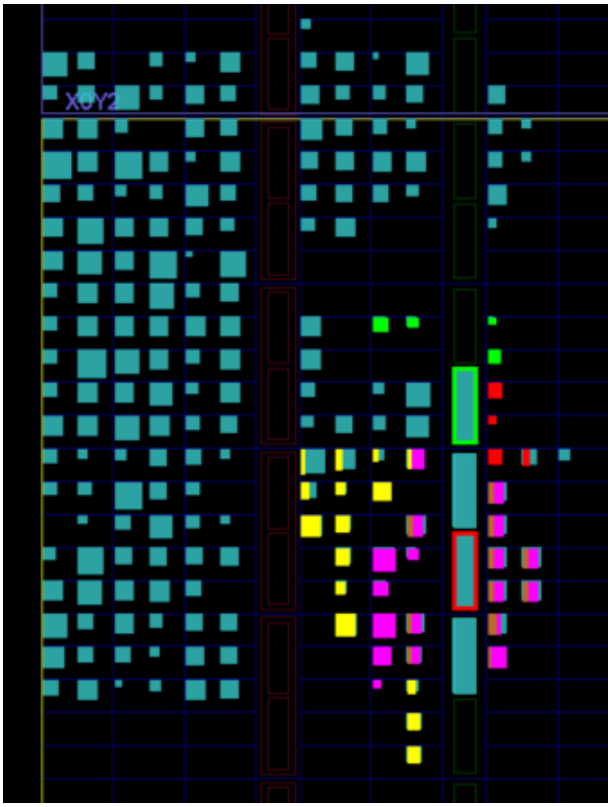


Figura 35 – *Layout* do PC fuzzy . *Controller* em amarelo, *Defuzzification* em verde-claro, MFC em rosa, *Connective operator* pelo vermelho e o MUX pelo marrom.

A Figura 36 detalha o consumo de potência da arquitetura de hardware proposta para uma temperatura de junção de 41.10° C. É uma estimativa de confiabilidade média, pois não utiliza o modelo de ativação, que estima a potência em termos de um fator de atividade, representando a probabilidade de um determinado bloco ou recurso do FPGA ser ativado em um determinado momento. Apesar disso, é possível obter uma estimativa de potência.

O consumo total foi de 1399 mW, sendo que 134 mW, cerca de 10%, é de potência estática. Os outros 90% de potência dinâmica, sendo que 96% deste total são usados pelo Zynq 7000 *processing System* (PS7), e 4% pelo PC fuzzy e outros sistemas. Supondo uma bateria com tensão de 3,7 V e capacidade de 4000 mAh, o tempo de funcionamento do circuito lógico do exoesqueleto seria de aproximadamente 10,58 horas, consumidos principalmente pelo Zynq 7000.

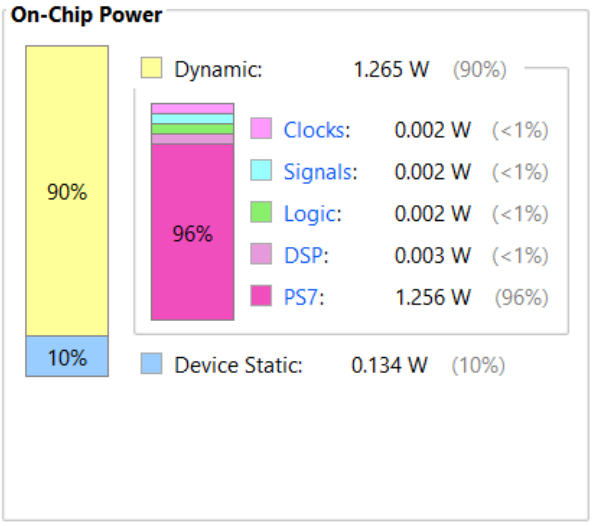


Figura 36 – Consumo de potência para a arquitetura proposta no FPGA.

A Figura 37 mostra o tempo de resposta do PC fuzzy. São necessários 7 ciclos de *clock* desde o recebimento das entradas (*Input*) até a saída (*Output*) estar disponível. Considerando a frequência de operação máxima do sistema de 73,9MHz, o *throughput* do PC fuzzy é de aproximadamente $1/(7 \times 73{,}9MHz) = 10{,}5$ MOPS (Mega Operações por Segundo). Uma estimação mais precisa da taxa de processamento deve considerar a latência do barramento de transmissão de entradas e saídas através do processador ARM Cortex A9. Tais resultados serão apresentados na seguinte subseção.

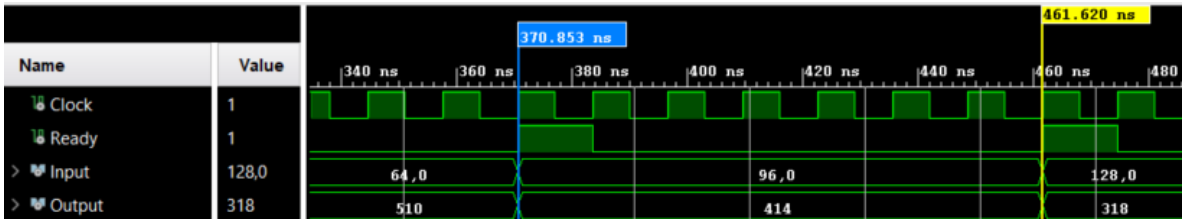


Figura 37 – Simulação do tempo de resposta em hardware do pré-compensador fuzzy.

## 4.6 Resultados da implementação do co-projeto hardware/software

Esta seção apresenta os resultados do co-projeto proposto implementado no SoC FPGA tendo como referência a base de dados da marcha humana normal e os parâmetros da abordagem do NSGA-2 apresentados na seção 4.2.

A segunda e terceira colunas da Tabela 25 apresentam os valores do RMSE para $\theta_1$ e $\theta_2$ do manipulador robótico em relação à referência da base de dados no co-projeto hardware/software. É importante notar que somente o PC fuzzy é executado em hardware, enquanto a planta e o controlador PID são executados em software, utilizando linguagem

Python, no processador ARM do Zynq7000. Observa-se que os resultados sem o PC fuzzy são iguais ao apresentado na seção 4.2. Com a inclusão do PC fuzzy em hardware observa-se um aumento do RMSE em cerca de 20%. Esse fato é esperado, pois, temos entradas e saídas limitadas a 10 bits. A quarta coluna da Tabela apresenta o erro numérico entre a implementação inteiramente em software (com pré-compensador na linguagem C), a qual usa uma representação numérica de ponto flutuante de 32 bits, e a implementação utilizando co-projeto hardware-software.

Tabela 25 – Tabela de análises da arquitetura sem e com PC fuzzy (segunda e terceira colunas) e o erro numérico entre hardware/software (quarta coluna).

| RMSE | Sem PC | Com PC | Erro numérico software/hardware |
|---|---|---|---|
| $\Theta_1$ | 0.096 | 0.059 | 0.014 |
| $\Theta_2$ | 0.257 | 0.060 | 0.015 |

A Tabela 26 apresenta os tempos de processamento do pré-compensador em software e hardware implementados no SoC FPGA. O tempo da implementação em hardware (considerando apenas o PC fuzzy) é calculado a partir do tempo de processamento da Figura 37 e do tempo de leitura e escrita do *AXI4-Lite* que são 10 ciclos de *clock*, portanto, o tempo de processamento em hardware é 20 ciclos de *clock* para a escrita das duas entradas, 7 ciclos de *clock* para o processamento do PC fuzzy, e 10 ciclos de *clock* para a leitura da saída, totalizando 37 ciclos de *clock*. A partir da frequência de operação de 73.923 MHz, temos o tempo aproximado de 500 ns. Observe que o fator de aceleração em hardware é de cerca de 98 vezes em relação ao software.

Tabela 26 – Tempo de processamento do PC fuzzy em software e hardware.

| Tempo médio | Total (ns) |
|---|---|
| Software | 49000 |
| Hardware | 500 |

As Figuras 38 e 39 apresentam as curvas de trajetória e do torque dos atuadores do manipulador robótico, respectivamente, para o co-projeto hardware/software. Observe que a proposta mostrou-se satisfatória com a melhoria da trajetória e com a suavização do torque semelhante aos resultados da seção 4.2.

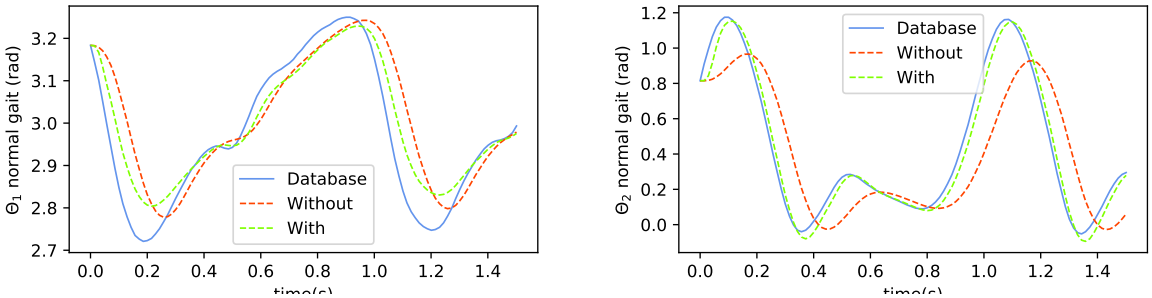


Figura 38 – Trajetória do manipulador com e sem PC fuzzy para a marcha humana em condição normal para o co-projeto hardware/software. Esquerda: $\theta_1$; Direita: $\theta_2$.

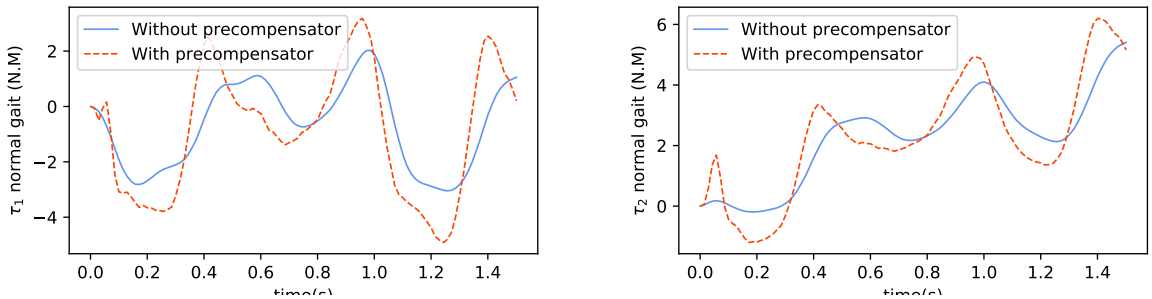


Figura 39 – Curva de torque dos atuadores do manipulador com e sem PC fuzzy para a marcha humana subindo escadas para o co-projeto hardware/software. Esquerda: $\tau_1$; Direita: $\tau_2$.

# 5 Conclusões

Os resultados da Seção 4.1 para o algoritmo mono-objetivo PSO mostra que a adição do PC fuzzy melhora o RMSE de ambas as juntas em mais de 50%, do manipulador robótico para as duas marchas humanas. No entanto, apresenta oscilações consideráveis nos torques dos atuadores. Com a introdução dos multiobjetivos NSGA-II e MODE nas Seções 4.2 e 4.3 o problema mostrado é atenuado para valores aceitáveis para a integridade dos atuadores, e os outros resultados são satisfatórios para ambos algoritmos e marcha humana. Com o NSGA-II, a melhoria chega até 83.750% para a condição de subir escadas.

Outro ponto importante a ressaltar é que nos algoritmos multiobjetivos, a melhoria do RMSE é maior para $\theta_2$, como apresentado no centro e à direita da Tabela 20. Já para o PSO, a melhoria é em $\theta_1$, mostrado à esquerda, também na Tabela 20. Esse fato pode ser explicado pela adição da função custo F2 para a suavização do torque, mostrando que a junta 2 possui grande influência na dinâmica do torque do manipulador robótico.

No entanto, outros estudos têm obtido melhores resultados. Por exemplo, em (HULTMANN AYALA; DOS SANTOS COELHO, 2012), o autor aplicou o algoritmo multiobjetivo NSGA-II sem pré-compensação para ajustar os parâmetros do mesmo manipulador utilizado neste trabalho e obteve resultados mais satisfatórios em termos de aproximação da trajetória do manipulador àquela originalmente desejada. Em (KUMAR, A.; KUMAR, V., 2017), o autor também utilizou o mesmo manipulador, mas ajustou os parâmetros com o algoritmo híbrido ABC-GA com pré-compensação e obteve melhores resultados para a trajetória, mas com oscilações no torque dos atuadores. Ambos os estudos modelaram as trajetórias desejadas como funções polinomiais com comportamento previsível. Até onde sabemos, nenhum outro estudo usou dados reais de marcha humana para ajustar um controlador PID fuzzy com pré-compensação para um manipulador robótico 2-DOF. É importante destacar que a abordagem de pré-compensação proposta compensa a falta de generalização comum nos controladores PID para manipuladores robóticos.

A solução da arquitetura de hardware do PC fuzzy, sem restrições de *timing*, utiliza menos de 2% dos recursos do FPGA e executa cerca de 10.5 MOPs. O tempo de operação de aproximadamente 10,58 horas para uma bateria de 4000 mAh mostra-se viável para um projeto de exoesqueleto. Além disso, se apenas o bloco do pré-compensador for utilizado, esse tempo aumenta consideravelmente, uma vez que o Zynq7000 consome mais de 96% da energia.

A implementação do co-projeto hardware/sofware mostrou ser viável a implementação do PC fuzzy em um exosqueleto real, com aquisição e filtragem de dados, pois além dos resultados satisfatórios da trajetória e torque dos atuadores, o tempo de resposta é bem

abaixo, e quase não afeta, do que é requisitado para o *loop* de controle integrado (10ms).

## 5.1 Trabalhos futuros

- Desenvolvimento de um controle clássico para a planta do manipulador robótico, como o *sliding mode control* (SMC) para efeitos de comparação com o ajuste dos parâmetros pelos algoritmos bioinspirados;
- Uso de outros algoritmos bioinspirados e *machine learning* para o ajuste dos parâmetros dos controladores;
- Desenvolvimento de outras arquiteturas de PC fuzzy em hardware, como por exemplo, a inferência Takagi-Sugeno, e/ou implementações de outros métodos de fuzzificação e defuzzificação, como mostra o trabalho (CARVALHO FARIA, 2021);
- Desenvolvimento de uma arquitetura em hardware usando uma representação numérica de ponto flutuante;
- Substituir o processador ARM pelo MicroBlaze ou similar, a fim de consumir menos recursos e potência. O MicroBlaze seria responsável pelo processamento dos dados de entrada, cálculos de pré-processamento, filtragem e condicionamento de dados;
- Integrar o PC fuzzy desenvolvido em hardware no controle de um exosqueleto real;
- Criar uma estimativa de consumo de potência com alta confiabilidade usando um modelo de ativação e realizar uma análise de eficiência energética. Por exemplo, verificar a quantidade de operações que o modelo pode executar por segundo em relação ao consumo de energia necessário para realizar essas operações (MOPS/W);
- Publicação dos resultados em revistas de sistemas inteligentes e/ou sistemas embarcados.

# Referências

AXI Reference Guide. 2012. Xilinx Inc., San Jose, California, USA. Citado na p. 47.

AZAR, A. T.; SERRANO, F. E. Fractional Order Two Degree of Freedom PID Controller for a Robotic Manipulator with a Fuzzy Type-2 Compensator. In: HASSANIEN, A. E.; TOLBA, M. F.; SHAALAN, K.; AZAR, A. T. (Ed.). **Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2018**. Cham: Springer International Publishing, 2019. Citado na p. 34.

BINGÜL, Z.; KARAHAN, O. A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control. **Expert Systems with Applications**, 2011. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417410007591>. Citado nas pp. 17, 19, 34, 35, 42.

CARVALHO FARIA, P. **Arquitetura de hardware para cálculo do centroide aplicado a defuzzificação de sistemas nebulosos com inferência do tipo Mamdani**. 2021. Citado nas pp. 19, 36, 67.

CHEN, G.; PHAM, T. **Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems**. CRC Press, 2019. ISBN 9780367397883. Disponível em: <https://books.google.com.br/books?id=lIySyAEACAAJ>. Citado na p. 24.

CHIU, Z.-K.; LEE, P.-J. A fuzzy control for obstacle avoidance implemented in the wheel robot with FPGA. In: 2017 International Conference on System Science and Engineering (ICSSE). 2017. Citado na p. 18.

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, 2002. Citado na p. 28.

ELAWADY, W.; LEBDA, S.; SARHAN, A. An optimized fuzzy continuous sliding mode controller combined with an adaptive proportional-integral-derivative control for uncertain systems. **Optimal Control Applications and Methods**, 2020. Citado na p. 17.

FUZZY LOGIC DESIGN TOOLS. 2018. Instituto de Microelectrónica de Sevilla (IMSE-CNM). Disponível em: <http://www2.imse-cnm.csic.es/Xfuzzy/>. Citado na p. 45.

GARCIA, P.; COMPTON, K.; SCHULTE, M.; BLEM, E.; FU, W. An Overview of Reconfigurable Hardware in Embedded Systems. **EURASIP Journal on Embedded Systems**, 2006. Citado nas pp. 30, 32.

GARCÍA-MARTÍNEZ, J. R.; CRUZ-MIGUEL, E. E.; CARRILLO-SERRANO, R. V.; MENDOZA-MONDRAGÓN, F.; TOLEDANO-AYALA, M.; RODRÍGUEZ-RESÉNDIZ, J. A PID-Type Fuzzy Logic Controller-Based Approach for Motion Control Applications. **Sensors**, 2020. Disponível em: <https://www.mdpi.com/1424-8220/20/18/5323>. Citado nas pp. 34, 36.

HULTMANN AYALA, H. V.; DOS SANTOS COELHO, L. Tuning of PID controller based on a multiobjective genetic algorithm applied to a robotic manipulator. **Expert Systems with Applications**, 2012. Citado nas pp. 18, 19, 22, 26, 34, 35, 44, 51, 54, 66.

INSTRUMENTS, N. **Fundamentos da tecnologia FPGA**. 2013. Disponível em: <http://www.ni.com/white-paper/6983/pt/>. Acesso em: 10 jan. 2023. Citado nas pp. 30, 31.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: PROCEEDINGS of ICNN'95 - International Conference on Neural Networks. 1995. Citado nas pp. 27, 28.

KIM, J.-H.; KIM, K.-C.; CHONG, E. Fuzzy precompensated PID controllers. **IEEE Transactions on Control Systems Technology**, 1994. Citado nas pp. 34, 36, 40.

KOENDJBIHARIE, M. W. **Control system design for exoskeleton of the right lower limb**. 2017. Master's Thesis – University of Brasilia. Citado na p. 17.

KOVACIC, Z.; BOGDAN, S. **Fuzzy Controller Design: Theory and Applications**. Taylor & Francis, 2005. (Automation and Control Engineering). Citado na p. 18.

KUMAR, A.; KUMAR, V. Hybridized ABC-GA optimized fractional order fuzzy pre-compensated FOPID control design for 2-DOF robot manipulator. **AEU - International Journal of Electronics and Communications**, 2017. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1434841117302327>. Citado nas pp. 17, 19, 26, 34, 35, 38, 40, 42, 50, 66.

LEE, K.; LEE, J.; WOO, B.; LEE, J.; LEE, Y.-J.; RA, S. Modeling and Control of a Articulated Robot Arm with Embedded Joint Actuators. In: 2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT). 2018. Citado na p. 17.

LI, L.; SCHNELLENBERGER, J.; NANDOR, M.; CHANG, S.; FOGLYANO, K.; REYES, R.-D.; KOBETIC, R.; TRIOLO, R.; QUINN, R. Embedded control system for stimulation-driven exoskeleton. In. Citado nas pp. 34–36.

LUO, Y.; COPPOLA, S.; DIXON, P.; LI, S.; DENNERLEIN, J.; HU, B. A database of human gait performance on irregular and uneven surfaces collected by wearable sensors. **Scientific Data**, 2020. Citado na p. 42.

MARAFA, N. A. **Mechanical design of a lower limb exoskeletonfor rehabilitation of paraplegic patient**. 2021. Master's Thesis – University of Brasilia. Citado na p. 17.

MASMOUDI, N.; HACHICHA, M.; KAMOUN, L. Hardware design of programmable fuzzy controller on FPGA. In: FUZZ-IEEE'99. 1999 IEEE International Fuzzy Systems. Conference Proceedings (Cat. No.99CH36315). 1999. Citado na p. 18.

MATLAB. **Fuzzy Logic Toolbox**. Natick, Massachusetts: The MathWorks Inc., 2018. Citado nas pp. 24–26, 39.

MEDEIROS, R. B. de; MUÑOZ, D. M. Tuning of FP-PID Controller based on PSO Algorithm Applied to a Human Gait. In: 2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE). 2022. Citado na p. 20.

MONTGOMERY, D.; RUNGER, G.; HUBLE, N. Engineering Statistics, 5ª Ed. John Wiley Sons, 2011. Citado na p. 60.

OGATA, K. Engenharia de controle moderno, 5ª Ed. Pearson Universidades, 2010. Citado nas pp. 23, 42, 43.

OSYCZKA, A. Multicriteria optimization for engineering design, 1985. Citado na p. 45.

PAMUNGKAS, D. S.; CAESARENDRA, W.; SOEBAKTI, H.; ANALIA, R.; SUSANTO, S. Overview: Types of Lower Limb Exoskeletons. **Electronics**, 2019. Disponível em: <https://www.mdpi.com/2079-9292/8/11/1283>. Citado na p. 17.

PYNQ-Z2 Reference Manual v1.0. 2018. TUL Technology Unlimited. Citado na p. 33.

ROBIČ, T.; FILIPIČ, B. DEMO: Differential Evolution for Multiobjective Optimization. In: COELLO COELLO, C. A.; HERNÁNDEZ AGUIRRE, A.; ZITZLER, E. (Ed.). **Evolutionary Multi-Criterion Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005a. Citado na p. 29.

ROBIČ, T.; FILIPIČ, B. DEMO: Differential Evolution for Multiobjective Optimization. In: COELLO COELLO, C. A.; HERNÁNDEZ AGUIRRE, A.; ZITZLER, E. (Ed.). **Evolutionary Multi-Criterion Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005b. Citado na p. 45.

ROMEIRO DE JESUS, T. **Arquiteturas de hardware dedicadas de controladores nebulosos para auxílio à locomoção de deficientes visuais**. 2017. Master's Thesis – University of Brasilia. Citado nas pp. 19, 36.

SÁNCHEZ-SOLANO, S.; BROX, M. Hardware Implementation of Embedded Fuzzy Controllers on FPGAs and ASICs. In: **Fuzzy Modeling and Control: Theory and Applications**. Edição: Fernando Matía, G. Nicolás Marichal e Emilio Jiménez. Paris: Atlantis Press, 2014. Citado nas pp. 36, 37, 46.

SENTHIL KUMAR, J.; KARTHIGAI AMUTHA, E. Control and tracking of robotic manipulator using PID controller and hardware in Loop Simulation. In: 2014 International Conference on Communication and Network Technologies. 2014. Citado na p. 23.

SEVAK, M. M.; PAWAR, T. D. ECG Signal Compression Using Singular Value Decomposition and Implementation on PYNQ-Z2. In: TUBA, M.; AKASHE, S.; JOSHI, A. (Ed.). **ICT Systems and Sustainability**. Singapore: Springer Nature Singapore, 2022. Citado na p. 34.

STORN, R.; PRICE, K. Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. **Journal of Global Optimization**, 1997. Citado na p. 29.

SUN, Y.; TANG, Y.; ZHENG, J.; DONG, D.; CHEN, X.; BAI, L. From sensing to control of lower limb exoskeleton: a systematic review. **Annual Reviews in Control**, 2022. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1367578822000189>. Citado na p. 17.

VIVADO Design Suite User Guide. 2019. Xilinx Inc., San Jose, California, USA. Citado na p. 45.

WINTER, D. A. Appendix A: Kinematic, Kinetic, and Energy Data. In: BIOMECHANICS and Motor Control of Human Movement. John Wiley  Sons, Ltd, 2009. Citado nas pp. 23, 42.

XILINX. **7 Series FPGAs Configurable Logic Block**. California, USA, 2016. Citado nas pp. 30–32.

YANG, X.-S. **Nature-Inspired Metaheuristic Algorithms**. 2010. Citado na p. 27.

ZADEH, L. Fuzzy sets. **Information and Control**, 1965. Disponível em: <https://www.sciencedirect.com/science/article/pii/S001999586590241X>. Citado na p. 17.

ZYNQ-7000 SoC Data Sheet: Overview. 2018. Xilinx Inc., San Jose, California, USA. Citado na p. 33.