DISSERTAÇÃO DE MESTRADO

SISTEMA EMBARCADO BASEADO EM ARQUITETURAS RECONFIGURÁVEIS DO CONTROLE DINÂMICO DE UMA MÃO ROBÓTICA SINTONIZADO COM ALGORITMOS BIOINSPIRADOS

Sergio Andres Pertuz Mendez

Brasília, Junho de 2017

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

SISTEMA EMBARCADO BASEADO EM ARQUITETURAS RECONFIGURÁVEIS DO CONTROLE DINÂMICO DE UMA MÃO ROBÓTICA SINTONIZADO COM ALGORITMOS BIOINSPIRADOS

Sergio Andres Pertuz Mendez

DISSERTAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA MECÂNICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO REQUISITO NECESSÁRIO PARA A OBTENÇÃO DO GRAU DE MESTRE EM SISTEMAS MECATRÔNICOS

APROVADA POR:

Prof. Daniel Maurício Muñoz Arboleda, FGA/UnB,PPMEC/UnB
Orientador

Prof. Carlos Humberto Llanos Quintero, PPMEC/UnB
Membro Interno

Profa. Suélia Rodrigues Fleury Rosa, FGA/UnB
Membro Externo

BRASÍLIA/DF, 23 DE JUNHO DE 2016

Pertuz, Sergio
SISTEMA EMBARCADO BASEADO EM ARQUITETURAS RECONFIGURÁVEIS DO CONTROLE DINÂMICO DE UMA MÃO ROBÓTICA SINTONIZADO COM ALGORITMOS BIOINSPIRADOS / Sergio Andres Pertuz Mendez. –Brasil, 2017.
100 p.

Orientador: Daniel Maurício Muñoz Arboleda

Dissertação (Mestrado) – Universidade de Brasília
Faculdade de Tecnologia – FT
Programa de Pós-Graduação em Sistemas Mecatrônicos – PPMEC, 2016.

1. Mão Robótica. 2. Controle de Impedância. 3. Arquiteturas Reconfiguráveis. 4. FPGA. 5. Algoritmos Bioinspirados. 6. Sistemas Embarcados. I. Daniel Muñoz, orientador. II. Universidade de Brasília. III. Faculdade de Tecnologia.

Dedicatória

Para mis amados padres Belmer y Lucila, este logro es de los tres, sin ustedes nada de esto hubiese sido posible. Gracias

Sergio Andres Pertuz Mendez

Agradecimentos

Primero que todo, quiero agradecer a mis maravillosos padres Belmer y Lucila quienes desde que tengo memoria han apoyado en todas mis metas, ellos que aun en la distancia, nunca dejaron que me sintiera solo por sus constantes bendiciones y gracias enviadas hacia mí, esas personas bondadosas que sin importar los sacrificios necesarios se han convertido en los impulsadores de mis sueños y la principal razón por la que hoy estoy culminando uno de mis objetivos profesionales y el cual es obtener el título de magister en una institución de gran reputación.

Quiero agradecer en segundo lugar al amor de mi vida Suamy quien se ha convertido en un pilar importantísimo en mi vida, por su cariño y comprensión incondicional.

Seguidamente pero no menos importante agradezco a mis hermanos Belmer, Luciani y Samanta quienes me llenan de orgullo al poder llamarlos hermanos, agradezco también a mi tan amada y gran familia quienes siempre han creído en mi y en las cosas que tengo el potencial de realizar. En especial a mi tía y madrina Nidia, a mi tía Yomalis, a mi tio Neil, a mis tíos Abelardo y Nailis y a todos aquellos quienes su acompañamiento y apoyo de alguna forma ha sido vital en mi proceso de formación.

Agradezco a mi orientador, Profesor Daniel Muñoz quien se convirtió en mucho mas que un mentor durante en este proceso, sino que también se ha convertido en un importante amigo y colega. Por su paciencia y tiempo dedicado durante todo este tiempo, le doy las gracias Daniel.

Al Profesor Carlos Llanos quien también se ha convertido en un mentor para este proyecto y quien me brindo un enorme apoyo durante mi carrera.

A mi primo y amigo Efraín, quien su apoyo también ha sido vital para conseguir este gran logro. A los amigos que conseguí en esta ciudad que al principio era extraña para mí y que sin embargo se ha convertido en un lugar mas familiar para mi y son Vanessa, Sergio$\times$2, Jõao, Renato, Carlos, Luiz, Ramsay, Sebastian, Marlon, Mayra, Reurison, Fabian y Oscar e todas las personas que tuve el gusto de conocer durante este tiempo.

A la CAPES (Cordenação de Aperfeiçoamento de Pessoal de Nível Superior) y el DPP-UnB (Departamento de pos-graduação da UnB) por su soporte financiero. Al IFB-Taguatinga (Instituto Federal de Brasilia) por prestar su laboratorio y equipo de prototipado electrónico e finalmente al Grupo de Automação e Controle (GRACO/UnB).

Sergio Andres Pertuz Mendez

RESUMO

Nos últimos anos grandes avanços tecnológicos formam feitos no campo da computação e áreas correlatas, o que permitiu o desenvolvimento de sistemas robóticos sofisticados como robôs biomiméticos. Esses robôs imitam sistemas biológicos que decorrem robustez e eficiência maiores se comparados com robôs convencionais quando usados em ambientes não estruturados. Por exemplo, uma mão robótica biomimética tem uma destreza e agilidade maior para executar tarefas de manipulação e agarres do que pinças convencionais.

Desde os anos oitenta, o desenvolvimento de mãos robóticas biomiméticas é o foco de pesquisa de várias equipes de investigação no mundo. Na atualidade há um número vasto de trabalhos encaminhados à construção e controle dos mesmos, os quais tem o intuito de melhorar a destreza e o desempenho das mãos e incluem tópicos como projeto da mão, mecanismos de movimento para as juntas, plataformas embarcadas e estratégias de controle. Existem várias abordagens a nível computacional que ainda não têm sido exploradas neste tipo de robôs, por exemplo o uso de um chip FPGA para o aumento de desempenho das estratégias de controle dinâmico usadas nos mesmos.

O presente trabalho descreve o desenvolvimento de uma arquitetura em hardware baseada em FPGA do controlador dinâmico de uma mão robótica, o qual é sintonizado usando algoritmos de otimização bioinspirada visando para atingir estabilidade de agarre.

O projeto da mão robótica realizado neste trabalho inclui o uso de mecanismos para emular os movimentos de flexão-extensão dos dedos. Os mecanismos foram otimizados visando minimizar o erro de trajetória usando a mão humana como referência. Os algoritmos bioinspirados PSO, DE e GA foram implementados para otimizar o mecanismo. 32 experimentos foram realizados para cada algoritmo a fim de realizar uma análise estatística para determinar o mecanismo com o melhor resultado, o qual é implementado no projeto final do mecanismo do dedo incluído no CAD da mão completa, o qual é descrito junto com o projeto eletrônico da plataforma. O projeto final da mão é avaliado com análise cinemática e adaptações do teste de Kapandji. Em seguida o protótipo é fabricado e montado usando diversos processos de fabricação e prototipagem, tais como corte a jato de água, torneamento e impressão 3D.

Logo após, foi projetado na plataforma *Matlab/Simulink* em alto nível o esquema de controle de impedância dos dedos o qual foi validado usando um simulador numérico do dedo para estudar o efeito do

controlador no sistema sem colocar em risco a plataforma física. A sintonização do controlador é realizada usando o algoritmo de otimização bioinspirado PSO visando reduzir o tempo de estabilidade, o sobreimpulso e o tempo de subida. Seguidamente, esses resultados foram implementados em plataformas embarcadas nas linguagens de programação C e na linguagem de descrição de hardware HDL. Após ser avaliado, o esquema de controle foi implementado em C na plataforma *Arduino* e mapeado em FPGA na placa de desenvolvimento *ZedBoard*. Uma comparação numérica e analítica foi realizada em termos do desempenho e precisão das duas abordagens.

O resultado da otimização do mecanismo de flexão-extensão produziu um erro de 0.2660% e o uso deste mecanismo permitiu fabricar um protótipo com dimensões e peso similar a uma mão humana real. Além disso, o protótipo atingiu os dez níveis da adaptação do teste de Kapandji. Adicionalmente, a sintonização da estratégia de controle resultou no comportamento desejado, o qual é subamortecido e com um tempo de estabilização $355ms$. Similarmente, os resultados da implementação em FPGA foram satisfatórios no sentido do desempenho do tempo de execução da estratégia de controle, o qual melhorou os resultados da implementação em *Arduino* e outros trabalhos correlatos no estado da arte.

ABSTRACT

In recent years, huge technological advances have been made in the field of computing sciences and related areas, which has allowed the development of sophisticated robotic systems such as biomimetic robots. These robots mimic biological systems that result in greater robustness and efficiency compared to conventional robots when used in unstructured environments. For instance, a biomimetic robotic hand has greater dexterity and agility to perform manipulation tasks and grasp than conventional grippers.

Since the 1980s, the development of biomimetic robotic hands has been the focus of many research teams all over the world. Nowadays several contributions regarding the construction and control of said systems, which aim to improve the dexterity and performance of the hands and include topics such as hand design, joint mechanisms, embedded platforms and control strategies.

However, even with the great available knowledge, there are still no perfect robotic hands, therefore, there is still knowledge to be contributed in the scientific community. There are several approaches at the computational level that have not yet been explored in this type of robots, for example, the use of a single FPGA chip to increase the performance of the dynamic control schemes used.

This work describes the development of an FPGA-based hardware architecture of the dynamic controller in a robotic hand, which is tuned using bioinspired optimization algorithms applied to achieve stability of grasps.

The robotic hand design performed in this work includes the use of mechanisms to emulate the flexion-extension movements of the fingers. The mechanisms were optimized to minimize the trajectory error using the human hand as reference. The bioinspired algorithms PSO, DE and GA were implemented to optimize the mechanism. 32 experiments are performed for each algorithm to perform a statistical analysis to determine the best result. This optimized mechanism was implemented in the final design of the finger mechanism included in the CAD of the complete hand, which is described together with the electronic design of the platform. The final hand design is evaluated with kinematic analysis and Kapandji clinical test adaptations. The prototype was manufactured and assembled using various manufacturing and prototyping processes, such as water-jet cutting, turning and 3D printing.

Afterwards, the finger impedance control scheme was designed on a high level platform using *Matlab/Simulink*, in addition to a numerical simulator of the finger for the study of the controller effect on

the system avoiding physical damage to the system. The controller was tuned using the PSO optimization algorithm aiming to reduce the stability time, the overshoot and the rise time. These results are then implemented on embedded platforms in both C and VHDL languages. After being evaluated, the control scheme is implemented in C on the Arduino platform and was manually mapped to FPGA on the ZedBoard development board. A numerical comparison between the two approaches was done in terms of performance and accuracy.

The result of the optimization of the flexion-extension mechanism produced an error of 0.2660% and the use of this mechanism allowed for manufacturing the prototype with dimensions and weight similar to a real human hand. The prototype reached the ten levels of the Kapandji test fitting. In addition, the tuning of the control strategy resulted in the desired behavior, which is underdamped and with a stabilization time of $355ms$. Similarly, the FPGA implementation results were satisfactory in the sense of the execution-time performance of the control strategy, which improved the implementation results in Arduino and other related work in the state of the art.

SUMÁRIO

RESUMO ..... **iii**

ABSTRACT ..... **v**

LISTA DE FIGURAS ..... **vii**

LISTA DE TABELAS ..... **ix**

LISTA DE ABREVIATURAS E ACROGRAMAS ..... **xiii**

1 Introdução **1**
1.1 Contextualização 1
1.2 Descrição do Problema 2
1.3 Justificativa 4
1.4 Objetivos 4
1.4.1 Objetivo Geral 4
1.4.2 Objetivos Específicos 4
1.5 Aspectos Metodológicos 5
1.6 Contribuição do Trabalho 6
1.7 Organização do Trabalho 6

2 Fundamentação Teórica **7**
2.1 Cinemática e Controle Cinemático de um Robô 7
2.1.1 Controle Cinemático 8
2.2 Dinâmica e Controle de Força de um Robô 8
2.2.1 Controle de Impedância 9
2.3 Agarres e Tipos de Agarre 9
2.3.1 Restrição Completa 10
2.3.2 Caracterização de Agarres 10
2.4 Hardware Reconfigurável 12
2.4.1 FPGAs (*Field Programmable Gate Arrays*) 12
2.5 Algoritmos Bioinspirados 13
2.5.1 PSO - Otimização por enxame de partículas 13
2.5.2 DE - Otimização por evolução diferencial 14

2.6 Estado da Arte sobre Mão Robóticas Biomiméticas 16
2.7 Conclusões do Capitulo 18

3 Projeto da Mão Robótica 19
3.1 Análise Cinemática do Modelo Simplificado de Mão - 7GDAs/16GDLs 21
3.2 Modelamento e Otimização do Mecanismo de Dedo Robótico 23
3.2.1 Modelamento Cinemático do Mecanismo do Dedo 24
3.2.2 Definição do Problema de Otimização 28
3.2.3 Resultados 29
3.2.4 Análise Estatística e Discussões 31
3.3 Projeto e Fabricação do Protótipo de Mão Robótica 34
3.3.1 Dedos Subatuados 34
3.3.2 Projeto CAD da Mão Robótica 35
3.3.3 Projeto Eletrônico 38
3.3.4 Fabricação do Protótipo de Mão Robótica 40
3.4 Conclusões de Capitulo 42

4 Projeto da Estratégia de Controle 45
4.1 Controlador de Impedância 46
4.1.1 Simulador do Dedo Robótico 48
4.2 Sintonização dos coeficientes do Controlador de Impedâncias 49
4.2.1 Definição do Problema 50
4.2.2 Resultados 51
4.3 Conclusões 53

5 Implementação da Estratégia de Controle 55
5.1 Implementação em C-Arduino 55
5.1.1 Pseudocódigo da Implementação 56
5.1.2 Resultados de Implementação 61
5.2 Implementação em VHDL-FPGA 64
5.2.1 Descrição de Hardware 64
5.2.2 Resultado e Análise de Síntese 68
5.3 Conclusões 73

6 Conclusões 75
6.1 Trabalhos Futuros 76

REFERÊNCIAS BIBLIOGRÁFICAS 77

LISTA DE FIGURAS

1.1 Resumo das técnicas computacionais usadas para os sinais tácteis. 2
1.2 Foto do protótipo de mão robótica biomimética *ManUPA*. 3

2.1 Relação entre cinemática inversa e direta. 7
2.2 Sistema massa-mola-amortecedor que representa o comportamento do controle de impedância. 9
2.3 Tipos de oposição para o agarre. 10
2.4 Taxionomia do agarre. 11
2.5 Estrutura interna de um FPGA. 12

3.1 GDLs de uma mão humana. 19
3.2 Estrutura cinemática da mão robótica simplificada de 7 GDAs. 22
3.3 Área de trabalho da mão robótica. 23
3.4 Mecanismo de 4 elos. 24
3.5 Mecanismo de 4 elos acoplado que simula os movimentos do dedo. 25
3.6 Trajetória alvo em contraste com a trajetória gerada pelo mecanismo com elos de comprimento aleatória. 26
3.7 Representação gráfica do modelo cinemático do mecanismo de 4 barras. 27
3.8 Trajetórias otimizadas (melhor valor dos 32 experimentos) visando atingir a trajetória alvo. . 30
3.9 Convergência do MSE do melhor resultado de cada algoritmo utilizado visando atingir a trajetória alvo. 30
3.10 Trajetórias otimizadas (melhor valor dos 32 experimentos) visando atingir o comportamento requerido dos ângulos. 31
3.11 Convergência do MSE do melhor resultado de cada algoritmo utilizado visando atingir o comportamento requerido dos ângulos. 31
3.12 Melhores trajetórias otimizadas 33
3.13 Mecanismo do dedo desmontado 34
3.14 Montagem do dedo robótico. 35
3.15 Dedo Indicador com eixos de flexão/extensão e adução/abdução. 36
3.16 Dedo polegar com eixos de flexão/extensão e adução/abdução. 36
3.17 Projeto CAD da palma integrando todos os dedos da mão robótica. 37
3.18 Teste de Kapandji. 37
3.19 Esquemático da placa eletrônica da mão robótica. 39
3.20 Projeto da placa eletrônica da mão robotica. 40

3.21 Peças fabricadas em aluminio. 40
3.22 Placa eletrônica da mão robótica. 41
3.23 Montagem Mão Robótica. 41
3.24 Mão robótica Montada. 42

4.1 Esquema de controle proposto. 47
4.2 Esquema de controle proposto. 47
4.3 Simulador do dedo robótico modelado em Matlab/Simulink Simmechanics. 49
4.4 Esquema de controle desenvolvido em Matlab/Simulink. 51
4.5 Convergência do MSE do processo de otimização do controlador de impedância. 52
4.6 Comportamento do controlador de impedância otimizado. 53

5.1 Dados do sensor de corrente com curvas de *fitting*. 59
5.2 Mão robótica com bola para agarrar 62
5.3 Comportamento do controlador de impedância implementado em Arduino. 63
5.4 Representação da FSM do esquema de controle em hardware. 64
5.5 Diagrama do fluxo de dados do esquema de controle implementado em hardware. 65
5.6 Nomenclatura de símbolos dos diagramas de fluxo de dados. 65
5.7 Diagrama de fluxo de dados do filtro de Kalman simples. 66
5.9 Diagrama de fluxo de dados do observador de torque. 66
5.8 Diagrama de fluxo de dados do filtro de Kalman para posição. 67
5.10 Diagrama de fluxo de dados do controlador de impedância. 68
5.11 Diagrama de fluxo de dados do controlador PID. 68
5.12 Simulação comportamental dos módulos de filtragem de corrente e posição e estimação de torque. 69
5.13 MSE da precisão dos cálculos realizados em FPGA vs Matlab 69
5.14 Comportamento do controlador de impedância implementado em FPGA. 70
5.15 Mapeamento dos módulos das arquiteturas em na FPGA Zynq XC7Z020-1CLG484 72

LISTA DE TABELAS

2.1 Comparações dos trabalhos correlatos de desenvolvimento de mãos robóticas 17

3.1 10 GDLs de maior importância para efetuar agarres. 20
3.2 Relações intrafâlangicas para mão simplificada de 7 GDLs. 21
3.3 Parâmetros cinemáticos da mão robótica. 22
3.4 Condições experimentais dos algoritmo PSO, DE e GA para a otimização do mecanismo do dedo. 29
3.5 Dados estatísticos da otimização 32
3.6 Teste *Wilcoxon ranksum* que compara o algoritmo DE usando função custo da trajetoria dos dedos com os otros. 33
3.7 CIs da placa eletrônica da mão robótica 38

4.1 Condições experimentais do algoritmo PSO para a otimização dos coeficientes do controlador de impedância. 52

5.1 Recursos da Arduino Mega utilizados na compilação do algoritmo de controle. Fonte: o autor 62
5.2 Recursos utilizados da FPGA para o controle de um dedo 71
5.3 Recursos utilizados da FPGA mapeando a arquitetura para a mão completa. 71
5.4 Comparação de desempenho das implementações (Arduino vs FPGA) 72

LISTA DE ABREVIATURAS E ACROGRAMAS

A	Amperes
aa	adução/abdução.
ADC	Conversor Análogo/Digital
B	Coeficiente de Atrito Viscoso do Amortecedor
CAD	Computer-Aided Design/Projeto Assistido por Computadoras
DE	Diferential Evolution/Evolução Diferencial
DIP	Articulação interfalângica distal
fe	flexão/extensão.
FPGA	Field Programmable Gate Array/Matriz de Portas Programáveis de Campo
FSM	Finite State Machine/Maquina de Estados Finitos
GA	Genetic Algorithm/Algoritmos Genéticos
GDA	Grau de Atuação
GDAs	Graus de Atuação
GDL	Grau de Liberdade
GDLs	Graus de Liberdade
HDL	Hardware Description Language/Linguagem de Descrição de Hardware
Hz	Hertz
K	Coeficiente da Mola
Kg	Quilograma
M	Massa
mA	miliAmperes
MCP	Articulação metacarpofalângea.

mm	milímetros
ms	milisegundos
MSE	Medium Square Error/Erro Quadrático Médio
N	Dimensionalidade/Número de variáveis de decisão
PC	Personal Computer/Computadora
PIP	Articulação interfalângica proximal.
PSO	Particle Swarm Optimization/Otimização por Enxame de Partículas
s	segundos
S	População/Tamanho do enxame
TCM	Articulação carpometacarpal do polegar.
V	Volts
VHDL	VHSIC Hardware Description Language/Linguagem de Descrição de Hardware VHSIC
ζ	Coeficiente de Amortecimento

Capítulo 1

Introdução

1.1 Contextualização

Nos últimos anos grandes avanços formam feitos no campo da inteligência artificial, controle de sistemas, robótica e outros campos do conhecimento relacionados. Esse progresso tem permitido o desenvolvimento de sistemas robóticos sofisticados, tais como os sistemas robóticos biomiméticos. A biomimética refere-se ao processo de imitar sistemas biológicos com robustez e eficiência maiores se comparado com robôs convencionais quando usados em ambientes não estruturados [1].

Desde os anos oitenta, o desenvolvimento de mãos robóticas biomiméticas tem sido o foco de pesquisa por várias equipes de investigação no mundo. Na atualidade há um número vasto de trabalhos encaminhados à construção de mãos biomiméticas multidedos, todos esses trabalhos foram feitos no intuito de melhorar a destreza e o desempenho das mãos, incluindo tópicos como projeto da mão, mecanismos de movimento para as juntas, destreza da mão, sensoriamento táctil. Contudo, atualmente não há sistemas de mãos robóticas com autonomia e destreza perfeita, porém, metodologias das abordagens atuais têm fornecido graus de agilidade e destreza maior nestes dispositivos [2].

Uma dessas metodologias é a realimentação táctil, a qual usa sinais táteis para a identificação das etapas na manipulação de um objeto: (a) contato/sem contato; (b) rotação e (c) deslizamento. Em aplicações com mãos robóticas, os sinais táteis têm sido usados para reconhecimento de objetos, controle de força e agarre. Diferentes tipos de sensores táteis medem diferentes tipos de grandezas, incluindo vetores de força, vibrações, e ações de contato, etc. Esses valores são submetidos então a vários tipos de técnicas computacionais no intuito de melhorar as estimativas e estimar outras grandezas (vide Fig. 1.1) [3].

Da Fig. 1.1 pode-se observar que o "controle de força de agarre"é um dos objetivos das técnicas computacionais. O agarre é um processo complexo para mãos robóticas, mesmo sabendo a forma, dimensões, textura e localização do objeto que vai ser manipulado. Em ambientes estruturados, essas e outras propriedades são conhecidas e, portanto, abordagens analíticas podem ser incluídas para efetuar a tarefa. Por outro lado, em ambientes não estruturados, os parâmetros do objeto não são conhecidos o que resulta em um incremento da dificuldade da tarefa de agarre, e representa um desafio grande na hora de resolver o problema de estabilidade do agarre [4].

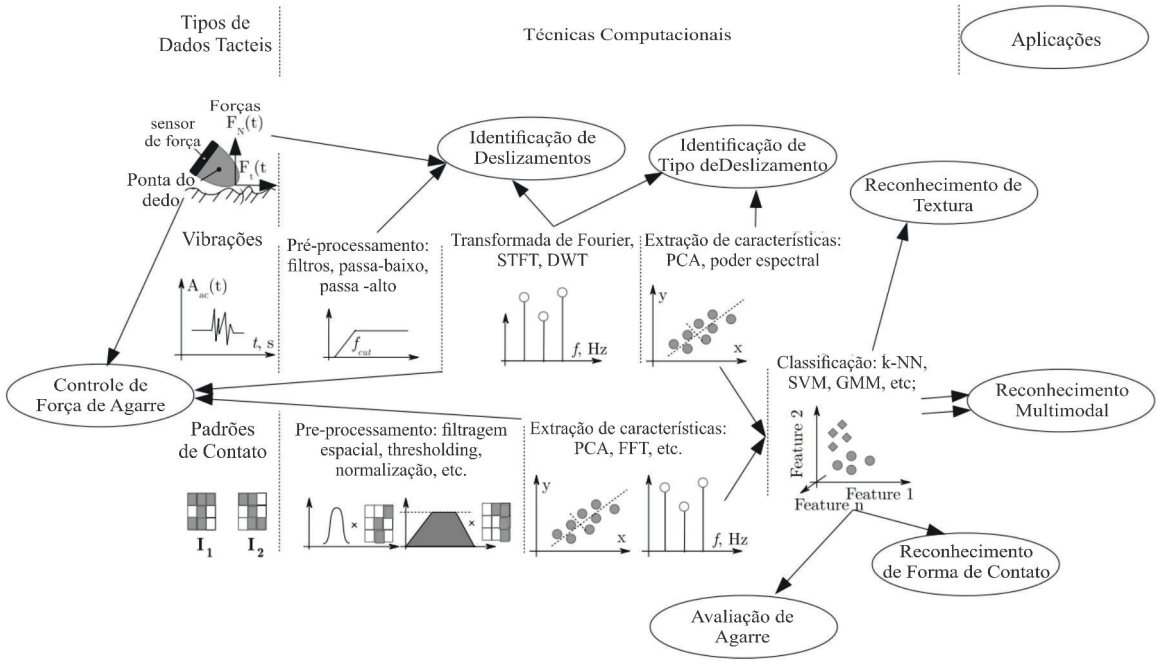


Figura 1.1: Resumo das técnicas computacionais usadas para os sinais tácteis. Fonte: adaptado de Kappassov et al.[3].

Devido ao grande número de elementos presentes em um sistema robótico biomimético (sensores, atuadores, etc.), os algoritmos de controle necessários para realizar diferentes tipos de agarre em mãos robóticas exigem requerimentos computacionais desafiantes tais como, alta velocidade para transferência de dados, manipulação de algoritmos de alta complexidade, realização de contas aritméticas e trigonométricas com alto desempenho, precisão, comunicação dependente entre processadores, altos requisitos de memória de dados e de programa, tamanho e peso reduzido do sistema computacional, entre outros.

Os FPGAs (do inglês *Field Programmable Gate Arrays*) são adequadas para estruturas lógicas flexíveis e complexas que visam resolver processamento de dados com alto desempenho. A velocidade de processamento pode ser melhorada com arquiteturas de processamento paralelas. Alguns pesquisadores têm embarcado algoritmos de controle em FPGA para melhorar o desempenho de um sistema servo-controlado. Seok et al. [5] trata o controle dinâmico de um sistema robótico multi-GDLs (*Graus de Liberdade*) onde foi alcançado uma frequência de controle de 4KHz com 50 processos diferentes em paralelo, para movimentos que podem ser comparados com uma locomoção real animal. Li e Huang [6] aborda o problema do controle de força em duas mãos robóticas multi-GDLs para tocar um piano, nessa abordagem é muito importante que o controle seja paralelo e que os robôs estejam sincronizados, isso foi alcançado com o uso de FPGAs, fechando malhas de controle paralelos para cada dedo.

1.2 Descrição do Problema

Como já foi descrito anteriormente, o agarre é uma tarefa especialmente difícil de atingir com mãos robóticas. Com ajuda de sensores o processo pode ser melhorado, porém, uma velocidade de processamento

maior é necessária no intuito de processar mais elementos na malha de controle. A meta a ser satisfeita nesse projeto é conseguir diferentes tipos de agarres estáveis com uma mão robótica.

Neste trabalho será desenvolvido uma plataforma de mão robótica de 7 graus de liberdade. Salienta-se que o protótipo desenvolvido neste trabalho é uma segunda versão do protótipo ManUPA (vide Fig. 1.2) desenvolvido anteriormente pelo autor desta dissertação [7]. O protótipo detém 7 atuadores elétricos (motores de CC), o quais são controlados de maneira desacoplada. A mão controla-se como uma célula de cinco robôs trabalhando em conjunto, portanto, resolve cinco malhas de controle diferentes que devem ser executados em paralelo.

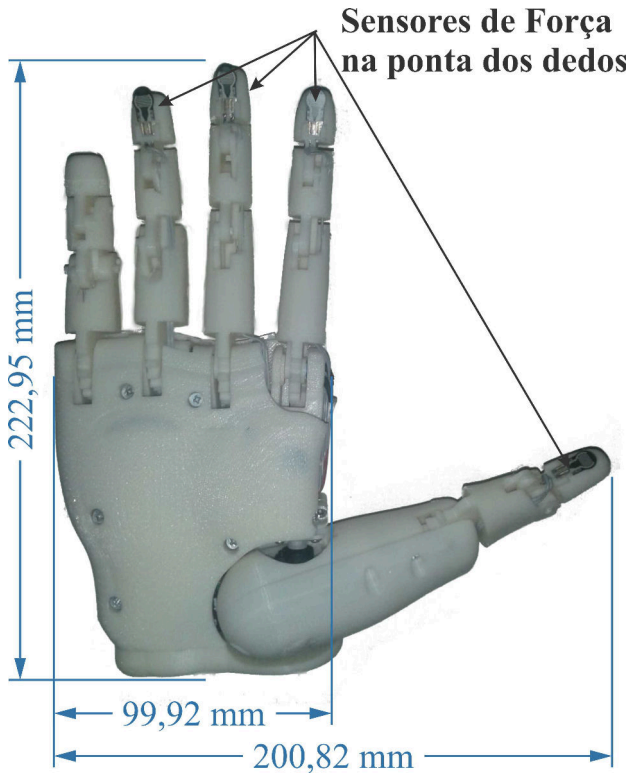


Figura 1.2: Foto do protótipo de mão robótica biomimética *ManUPA*. Fonte: o autor.

Por outro lado, o protótipo conta com 7 sensores de posição angular para cada atuador e 6 sensores de corrente. Os sensores de corrente serão usados no presente trabalho para estimar o torque gerado pelo motor para detectar contato com objetos.

Com o intuito de atingir tarefas de agarre e manipulação com a mão robótica, se propõe implementar um esquema de controle dinâmico, o qual usa-se como uma metodologia para resolver o problema de estabilidade do agarre. No intuito de ter um comportamento correto, o qual é uma resposta subamortecida e com um tempo de subida rápido, o controlador deve cumprir requisitos mínimos de frequência de atualização.

1.3 Justificativa

O controle dinâmico no protótipo de mão robótica permite atingir agarres mais estáveis e potencialmente detectar deslizamentos do objeto no momento do agarre. Para a detecção de deslizamentos é mister da frequência de processamento dos controladores seja alta, assim a detecção de mudanças abruptas nos sensores poderia determinar esses deslizamentos.

O problema ao tentar atingir alto performance para algoritmos de controle, é a complexidade computacional dos mesmos. Essa complexidade é derivada do grande número de operações aritméticas requeridas para cada malha de controle. Desta forma, a exploração de dispositivos reconfigurareis permite paralelizar as operações e potencialmente atingir uma velocidade de processamento alta.

Adicionalmente, durante o processamento dos sinais são usados diversos tipos de filtros e métodos estocásticos de estimação, tais como filtragem Kalman, que possuem um paralelismo intrínseco que pode ser explorado por arquiteturas paralelas usando FPGAs.

No protótipo de mão robótica biomimética a inclusão de mecanismos de transmissão de 4 elos para emular os movimentos dos dedos acarreta possíveis singularidades, o qual dificulta o controle do mesmo. Neste sentido, um algoritmo de otimização pode explorar soluções que minimizem a ocorrência dessas singularidades.

Em resumo, a mão robótica conta com um total de 20 elementos entre sensores e atuadores, os quais devem ser processados paralelamente para realizar controle cinemático e dinâmico da mão robótica. Neste contexto, dispositivos FPGAs permitem mapear em hardware os algoritmos envolvidos explorando o paralelismo intrínseco. Essa abordagem visa alcançar os requisitos da frequência de atualização do controlador na ordem de ms. Adicionalmente, FPGAs são uma solução factível para o processamento digital dos sensores de força e corrente permitindo explorar arquiteturas paralelas que melhorem a estimação dos valores dos sensores.

1.4 Objetivos

1.4.1 Objetivo Geral

Como objetivo geral, esse projeto de pesquisa apresenta o desenvolvimento de um sistema embarcado baseado em arquiteturas reconfiguráveis do controle dinâmico de uma mão robótica sintonizado com algoritmos bioinspirados aplicada ao problema de estabilizar agarres.

1.4.2 Objetivos Específicos

Para cumprir o objetivo geral deste trabalho, as seguintes metas devem ser alcançadas.

- Otimizar o projeto mecânico do dedo robótico no intuito de minimizar o erro de trajetória com relação à mão humana.

- Desenvolver um simulador que permita validar as estratégias de controle levando em consideração aspectos dinâmicos do sistema do dedo robótico.
- Implementar uma estrategia de controle de impedância baseada na estimativa de torque a partir de medições de corrente dos atuadores.
- Sintonizar o controlador de impedância no intuito de melhorar a resposta dinâmica do sistema.
- Mapear os algoritmos de controle e estimação de variáveis em um dispositivo FPGA no intuito de maximizar a frequência de atualização do controlador.

1.5 Aspectos Metodológicos

Neste trabalho a revisão da literatura é realizada de maneira sistemática com o intuito de adquirir conhecimentos sobre os tópicos a tratar, além de obter familiaridade com as novas ferramentas que serão usadas no desenvolvimento do projeto, como o uso de estratégias de controle de impedância aplicado para tarefas de agarre. A técnica utilizada nessa revisão é de tipo bibliográfica, se fundamentado em fontes bibliográficas de artigos científicos publicados em jornais relevantes na área correlata ao trabalho, a qual é engenharias em tecnologias de robótica, controle e computação.

A pesquisa desenvolvida neste trabalho quanto à natureza é uma pesquisa aplicada porque gera conhecimentos para aplicação pratica de algoritmos de otimização bioinspirada, dirigido à solução de otimizar a trajetória de mecanismos que emulam os movimentos de flexão/extensão de um dedo para serem implementados em uma mão robótica.

O resultado físico desta pesquisa é o protótipo da mão robótica projetada, fabricada e montada usando diferentes métodos de manufatura, como corte a jato de água, torneamento e impressão 3D. Dito protótipo detêm a sua estrutura validada através de análise cinemático e uma adaptação do teste clínico de Kapandji. Os mecanismos dos dedos são otimizados com diferentes algoritmos bioinspirados (PSO, DE e GA), e os resultados são testados com análise estatística para provar o melhor. Essa análise é quantitativa e está enquadrada na escala ordinal porque usa relações definidoras de equivalência e maior do que, além de utilizar os testes estatísticos de mediana, Kolmogorov-Smirnov, Krustal-Wallis e *Rank-sum*.

Quanto ao procedimento, este trabalho também é uma pesquisa de tipo experimental porque determina a manutenção de agarres estáveis em mãos robóticas como objeto de estudo, seleciona o uso e implementação da estratégia de controle de impedância em sistemas embarcados como variável capaz de supervisionar este fenômeno e observa os efeitos que a variável produz na execução do objeto de estudo.

A estratégia de controle de impedância proposta é validada com simulação numérica e sintonizada usando algoritmos de otimização bioinspirada com o objetivo para atingir o comportamento requerido, o qual é subamortecido com tempo de resposta rápido e o erro em estado estável pequeno. Adicionalmente, a estratégia é implementada em dispositivos embarcados em C para avaliar o comportamento do controlador na vida real. Logo após, a mesma é mapeada em FPGA para simulação numérica, visando analisar a precisão do resultado e o tempo de execução com uma implementação em PC (Matlab) e com um Arduino (solução embarcada em C). A precisão é analisada quantitativamente na escala nominal, usitando relações

definidoras de equivalência e apenas estatísticas de porcentagens.

Finalmente, os todos os resultados obtidos neste trabalho são analisados de maneira qualitativa de conteúdo. Comparando de forma discursiva as caraterísticas físicas e comportamentais presentes em trabalhos correlatos desenvolvidos por outros autores.

1.6 Contribuição do Trabalho

As contribuições relevantes do trabalho atual são: (a) a redução das singularidades mecânicas usando algoritmos bioinspirados de otimização; (b) A sintonização do controlador de impedância usando algoritmos bioinspirados, a qual é uma abordagem que não tem sido publicada na literatura científica; (c) Outra contribuição do trabalho atual foi o desenvolvimento de bibliotecas para Arduino de controle de impedância e filtragem de Kalman para estimação de velocidades, as quais não existiam na web. Os repositórios destas bibliotecas foram disponibilizadas na web para uso do público geral. Finalmente, está a proposta de uma arquitetura paralela em FPGA para o controle de baixo nível de uma mão robótica, o qual não foi reportado na literatura cientifica.

1.7 Organização do Trabalho

O presente trabalho contém 5 capítulos e está organizado da seguinte maneira: (a) O capitulo 2 apresenta o baseamento teórico específico necessário para o entendimento deste trabalho junto com algumas definições; (b) O capítulo 3 descreve o começo deste trabalho de pesquisa o qual foi a definição cinemática da mão robótica, a otimização bioinspirada dos mecanismos dos dedos e a fabricação e montagem do protótipo incluindo o projeto mecânico e eletrônico.; (c) O capitulo 4 detalha o projeto do controlador de impedância desenvolvido neste trabalho, alem disso, apresenta uma técnica de sintonização do controlador usando PSO e um simulador numérico do projeto mecânico; (d) O capitulo 5 descreve a implementação do controlador desenvolvido no capítulo anterior em arquiteturas embarcadas (*Arduino* e FPGA) e detalha uma análise de precisão numérica nos cálculos e de performance de execução dos algoritmos de controle embarcados; (e) Finalmente as conclusões do trabalho são apresentadas no capitulo 6.

Capítulo 2

Fundamentação Teórica

Neste capitulo, uma abordagem teórica sobre os tópicos a serem tratados neste trabalho será apresentada. No intuito de fazer um estudo sobre o estado da arte serão abordados os diferentes métodos de controle de mão biomimética, tipos de agarre, conceitos sobre hardware reconfigurável e aspectos gerais sobre algoritmos de otimização. Finalmente, uma tabela comparativa sobre os trabalhos correlatos encontrados na literatura científica será apresentada.

2.1 Cinemática e Controle Cinemático de um Robô

A cinemática de um robô é o estudo do movimento em relação a um sistema de referência, sem considerar as forças nele envolvidas. Geralmente, esse estudo é apresentado através de uma relação entre o órgão terminal do robô e os valores das juntas. A *cinemática direta* determina a posição e atitude do órgão terminal do robô com respeito a um sistema de coordenadas de referência, os valores conhecidos das juntas e os parâmetros geométricos dos elementos do robô. A *cinemática inversa* calcula a configuração que deve ser adotada para atingir uma posição e atitude conhecida do órgão terminal [8].

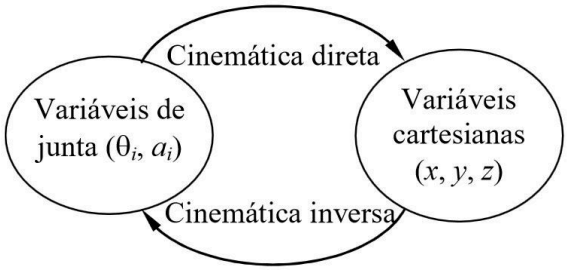


Figura 2.1: Relação entre cinemática inversa e direta. Fonte: o autor

Existem vários métodos para calcular o modelo cinemático de um robô: a) o método geométrico, o qual não é sistemático, precisa de considerações geométricas e é valido para robôs com poucos GDLs, e b) o método baseado em mudança da base, o qual é sistematizado e adequados para qualquer cadeia cinemática e números de GDLs.

No trabalho atual, o modelamento cinemático do mecanismo do dedo é realizado com o método geométrico, no entanto para a análise cinemática da mão robótica é utilizado o método baseado em mudança de base. Adicionalmente, parte deste trabalho também visa controlar os movimentos da mão robótica, com isto em mente é apresentado o controle cinemático e sua função a seguir.

2.1.1 Controle Cinemático

O objetivo do controle cinemático é estabelecer as trajetórias que deve seguir cada junta do robô ao longo do tempo, isso para atingir os objetivos fixados pelo usuário, tais como a posição alvo (*set point*), o tipo de trajetória do órgão terminal, o tempo investido no movimento, etc.

O robô pode ser programado para seguir dois tipos de trajetórias: a) trajetória no espaço da tarefa (com posições cartesianas) e, b) trajetória no espaço do movimento da junta (das variáveis de junta). É importante salientar que não é possível seguir qualquer trajetória dado que os robôs têm limitações mecânicas.

O controle cinemático funciona de acordo com o seguinte procedimento:

1. Converter a especificação do movimento em uma trajetória analítica no espaço cartesiano.
2. Realizar uma amostragem da trajetória cartesiana obtendo um número finito de pontos. Cada um desses pontos vem dado por uma conjunto de valores, tipicamente $(x, y, z, \phi, \theta, \psi)$.
3. Usar a transformação de cinemática inversa para representar cada ponto com valores de variáveis de junta.
4. Interpolação dos valores das variáveis de junta obtidas, gerando para cada valor uma função $q_i(t)$ que represente de forma aproximada (*fitting*) os valores das variáveis de junta.
5. Amostrar a trajetória da junta para gerar uma referência para o controle dinâmico.

O controle cinemático neste trabalho é desenvolvido aplicando um controlador PID com realimentação de posição.

2.2 Dinâmica e Controle de Força de um Robô

A dinâmica de um robô é a relação entre a atuação e as forças de contato, e a aceleração com a trajetória do movimento resultante. As equações dinâmicas são a base para vários algoritmos computacionais úteis no projeto mecânico, controle e simulação.

O controle dinâmico visa moderar o contato físico do robô com o ambiente, isto é um requerimento fundamental para realizar tarefas de manipulação. Uma das vantagens deste tipo de controle é a sua característica de efetuar comportamentos robustos e flexíveis interatuando em ambientes pouco estruturados. As estratégias de controle ativo dessas interações podem-se dividir em dois tipos: a) quando se realiza controle indireto de força e, b) quando se realiza controle direto de força. O primeiro atinge o controle dinâmico através do controle cinemático sem realimentação de força ou torque. Em contraste, o segundo tipo de

abordagem propõe o controle das forças de contato usando a realimentação de força ou torque do sistema [9]. A primeira pertence ao controle de impedância descrito na seguinte subseção.

2.2.1 Controle de Impedância

O controle de impedância (*ou controle de admitância*) é descrito como um sistema equivalente de massa-mola-amortecedor com parâmetros ajustáveis. Nesta relação, a impedância é reação passiva que o robô realiza quando é perturbado por forças externas atuantes enquanto a admitância corresponde à reação ativa do robô a essas forças [9].

A Eq. 2.1 apresenta o modelo de equação diferencial de segunda ordem em domínio do tempo do sistema na Fig. 2.2 que representa a ação do controle de impedância.

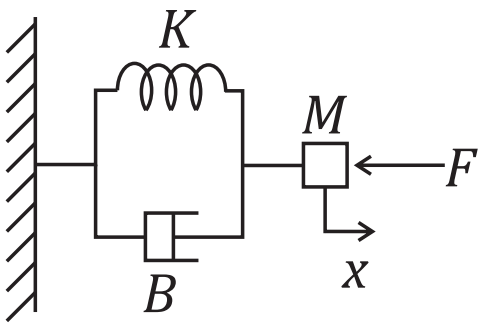


Figura 2.2: Sistema massa-mola-amortecedor que representa o comportamento do controle de impedância. Fonte: o autor.

$$M\ddot{x}(t) + B\dot{x}(t) + Kx(t) = F(t) \tag{2.1}$$

onde x é a posição do robô, $\dot{x}$ e $\ddot{x}$ são as derivadas de x e M, B e K são valores definidos pelo usuário que representam as grandezas de massa, o coeficiente de atrito viscoso do amortecedor e a constante da mola respectivamente.

2.3 Agarres e Tipos de Agarre

Para a análise deste trabalho a seguinte definição de agarre vai ser abordada. *"O agarre é toda aquela postura estática da mão que permite agarrar um objeto independentemente da orientação da mesma"* [10].

As mãos mecânicas foram desenvolvidas para dar aos robôs a habilidade de realizar manuseio ou agarre de objetos de diferentes geometrias y propriedades físicas. Um dos comportamentos mais desejáveis no agarre é a manutenção do mesmo na presença de perturbações, causadas por forças e momentos desconhecidos aplicados ao objeto. A manutenção do agarre significa manter as forças de contato no intuito de prevenir a separação com o objeto e deslizamentos indesejados [4].

2.3.1 Restrição Completa

A característica mais importante do agarre é a restrição completa. Um agarre com restrição completa consegue uma manutenção do agarre segura. As duas principais propriedades da restrição completa são o fechamento de forma e fechamento de força.

Assuma uma mão agarrando um objeto com todos os ângulos das suas juntas fixados e a palma estática; logo, o agarre tem fechamento de forma se é impossível realizar movimentos no objeto (vide figura 2.3(b)). Consequentemente, fechamento de forma ocorre quando a palma e os dedos da mão envolvem o objeto formando uma gaiola sem deixar espaço para o objeto se movimentar. Um dos requerimentos para o fechamento de forma é que, para um objeto rígido com n GDLs, são necessários pelo menos $n + 1$ pontos de contato. É importante salientar que essa condição não é suficiente, pois mesmo com $n + 1$ pontos de contato, devem-se considerar as suas posições.

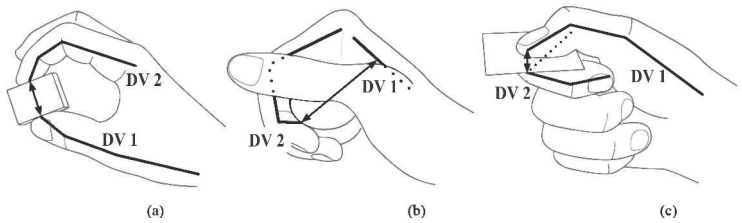


Figura 2.3: Tipos de oposição para o agarre. Oposições (a) apoiada (b) da palma e (c) de lado. Fonte: adaptado de Feix *et al.* [11]

Sob as mesmas condições, um objeto tem fechamento de força quando o objeto é agarrado deixando livre alguns movimentos do objeto. Portanto, um fechamento de força pode ser feito com menos contatos do que o fechamento de forma. Contudo, o fechamento de força precisa habilidade para controlar as forças internas da mão. O fechamento de força utiliza as forças de atrito para manter o equilíbrio do objeto. Uma vantagem de incluir o atrito no analise é a redução de pontos de contato para atingir o fechamento do objeto. Para atingir um fechamento de força são necessários apenas dois pontos de contato com o objeto se os dedos são modelados como dedos suaves (aqueles que consideram deformações geradas por forças) ou três contatos quando são modelados como dedos rígidos (aqueles que não consideram deformações). Note-se que todos os fechamentos de forma são fechamentos de força também, pelo uso do atrito para manter o agarre estável. Existem agarres com fechamento parcial nos quais só alguns dos possíveis GDLs são restringidos pelo fechamento de forma [4].

2.3.2 Caracterização de Agarres

Feix *et al.* [11] apresenta uma análise para definir os agarres feitos pela mão humana. Nesse trabalho, apresenta-se uma taxionomia de 33 tipos distintos de agarres tendo como base só os agarres estáticos, onde o objeto é fixado com relação à mão (sem movimento dentro da mão). Além disso, três conceitos principais foram utilizados para a abordagem desta taxionomia.

- **Agarres de potência, intermediários e de precisão**. Cada tipo de agarre pode ser classificado pela

sua necessidade de precisão ou potência. Existem agarres de: a) potência ou fechamento de forma; b) manipulação de precisão ou fechamento de força (no presente trabalho os movimentos intrínsecos no objeto não serão levados em consideração); e c) agarre intermediário ou fechamento parcial.

- **Tipos de Oposição**. Existem três direções relativas às quais a mão exerce forças para atingir agarres seguros, estas são: a) oposição apoiada (vide Fig. 2.3(a)); b) oposição da palma (vide Fig. 2.3(b)); e c) oposição de lado (vide Fig. 2.3(c)).

- **Dedos virtuais**. Para várias tarefas, diferentes dedos funcionam como uma unidade funcional aplicando forças na mesma direção e atuando de forma sincronizada (vide Fig. 2.3). Essa unidade funcional também é conhecida dedo virtual (DV).

A taxionomia resultante de Feix *et al.* [11] é a apresentada na Fig. 2.4. As colunas estão dispostas de acordo aos requerimentos de agarres de potência/precisão. No segundo nível (Op), a classificação depende do tipo de oposição (de palma, apoio e lado). No terceiro nível (DV), classificam-se os dedos virtuais. A posição do polegar é usada para diferenciar as duas linhas da Fig. 2.4, onde o polegar pode estar aduzido ou abduzido. Uma analise mais detalhado sobre a taxionomia apresentada pode ser encontrada em [11].

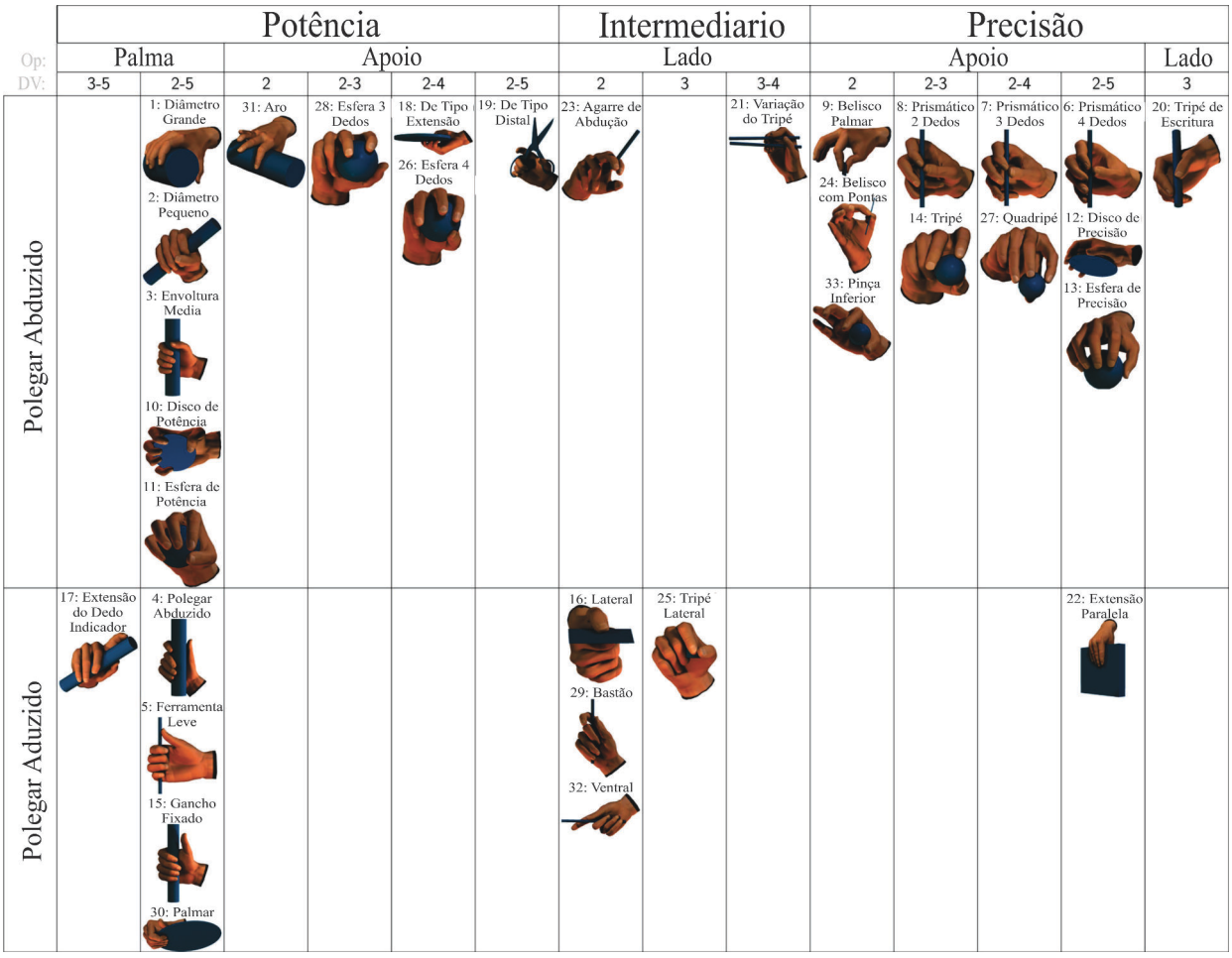


Figura 2.4: Taxionomia do agarre. Fonte: adaptado de Feix *et al.* [11]

2.4 Hardware Reconfigurável

O hardware reconfigurável mistura a programabilidade após a fabricação com o estilo de computação espacial (paralelo) de circuitos de aplicação especifica “ASICs” (do inglês *Application Specific Integrated Cicuits*), o qual é mais eficiente se comparado com o estilo de computação temporal (sequencial) dos processadores de fluxo de instruções. O hardware reconfigurável oferece um excelente balance entre eficiência e flexibilidade na hora de implementar um sistema. Devido aos acrescentes requerimentos de flexibilidade (e.g. a necessidade de um dispositivo conseguir se adaptar a ambientes e condições de operação), os dispositivos precisam ser adaptáveis às aplicações que abordam. Por outro lado, a solução também deve ser eficiente em termos de consumo de potencia e tempo de execução. Os ASICs não são adequados para o tipo de problema onde é necessário certo graus de flexibilidade, porem o hardware reconfigurável apresenta uma opção interessante para esses casos [12].

2.4.1 FPGAs (*Field Programmable Gate Arrays*)

Os FPGAs são dispositivos de hardware reconfigurável. A sua arquitetura interna esta definida pela sua característica de *granularidade*, refletindo sob o bloco mais pequeno que compõe o dispositivo, estes blocos são chamados Blocos Lógicos Configuráveis (CLB). Nos dispositivos com arquiteturas de*granularidade fina* os CLBs estão compostos por memórias baseadas em LUTs (*Look-up Tables*) que implementam funções combinatórias e alguns flip-flops, os quais podem ser programados para lógica sequencial para funções simples (e.g. um somador de 2 bits). Esses CLBs são conectados através de uma rede de interconexão programável. Operações mais complexas podem ser feitas reconfigurando essa rede [12].

Um FPGA pode ter milhares de CLBs e operam a frequências de até 500 MHz. Quatro componentes principais fazem parte geralmente em uma FPGA (vide Fig. 2.5): (a) Módulo de lógica ou bloco lógico; (b) elemento de configuração ou de roteamento; (c) blocos de entrada e saídas; (d) blocos de conexão e (e)circuitos especiais (DSPs, RAM, PLLs, ADC, etc.) .

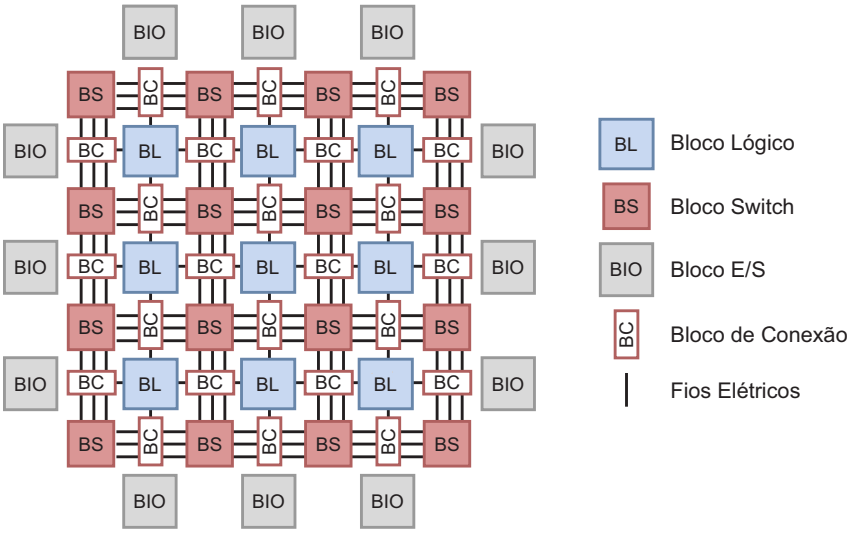


Figura 2.5: Estrutura interna de um FPGA. Fonte: o autor.

Entre as vantagens de usar um FPGA com relação a ASICs podem-se mencionar[13]:

- O sistema é reconfigurável.
- Custo de projeto em FPGAs é bem menor.
- *Time to market* é menor para implementações em FPGAs.
- Facilidade e tempo de desenvolvimento

Porém, o uso de FPGA acarreta as seguintes desvantagens:

- Área em silício maior.
- Consumo de potência maior.
- Custo de produção em massa maior.
- Desempenho (frequência de operação) restrito à tecnologia disponível pelo fabricante.

2.5 Algoritmos Bioinspirados

A otimização é um princípio que sempre há estado na natureza, os organismos sempre estão procurando estar em um estado ótimo para sobreviver, desde as partículas menores como os átomos até a humanidade são regulados por esse princípio. A natureza sempre está procurando um estado ótimo com o menor esforço possível. Os algoritmos bioinspirados baseiam-se no comportamento da natureza; como por exemplo, colônias de formigas, colônias de abelhas, enxames de peixes e aves, entre outros, os quais estão baseados nos princípios de compartilhamento de informação para a sobrevivência das espécies naturais.

Esses algoritmos são usados no campo da otimização global (entre outros), o seu objetivo é encontrar os melhores resultados x^* em um espaço de busca χ que cumpram com os objetivos $F = f_1, f_2, ..., f_n$. Esses eventos são expressados como funções matemáticas [14].

Neste trabalho serão usados algoritmos bioinspirados para resolver problemas de otimização global multimodal mono-objetivo (com apenas uma função custo) no intuito de reduzir o erro na trajetória do projeto mecânico do dedo e de sintonizar os parâmetros do controlador de impedância. Em particular, no presente trabalho foram explorados os algoritmos PSO e DE, os quais são apresentados nas seguintes subseções.

2.5.1 PSO - Otimização por enxame de partículas

O PSO (do inglês *Particle Swarm Optimization*) é um algoritmo que simula a inteligência coletiva de um enxame; como por exemplo, o comportamento de um sistema biológico social, tais como, bandos de aves ou cardumes de peixes quando estão explorando uma fonte de comida.

A posição de uma partícula representa uma possível solução ao problema de otimização. Cada partícula mensura a sua aptidão mediante a avaliação da função objetivo f e conserva seu conhecimento do melhor valor de aptidão por meio de uma memória individual $\overrightarrow{y_i}$ e uma memória coletiva $\overrightarrow{y_s}$. O movimento pelo espaço de busca é realizado através de uma velocidade aleatória $\overrightarrow{v_i}$ associada, calculada pela Eq. 2.2. A posição de cada partícula $\overrightarrow{x_i}$ é calculada com a Eq. 2.3 [14].

$$v_{ij}^{(t+1)} = wv_{ij}^{(t)} + c_1 \overbrace{U_{1j}[0,1](y_{ij}^{(t)} - x_{ij}^{(t)})}^{\text{componente cognitivo ind.}} + c_2 \overbrace{U_{2j}[0,1](y_{sj}^{(t)} - x_{ij}^{(t)})}^{\text{componente social}} \tag{2.2}$$

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + v_{ij}^{(t+1)} \tag{2.3}$$

onde, w é o fator de inercia, o qual diminui linearmente durante a execução do algoritmo, c_1 é o coeficiente cognitivo individual ou coeficiente de autoconfiança, c_2 é o coeficiente social ou coeficiente de confiança no conhecimento do enxame, U_1 e U_2 são números aleatórios uniformemente distribuídos na faixa [0,1].

O Algoritmo 1 apresenta o pseudocódigo para o algoritmo de otimização PSO.

Algorithm 1 Algoritmo de otimização PSO

função PSO-BASICO(S,N,c_1,c_2,x_{max},x_{max},Max_{iter},umbral)
  inicializa enxame;
  repeat
    para i $\leftarrow$ 1 **até** S **faça**
      se $f(x_k) \leq f(y_{ik})$ **então**
        $y_{ik} \leftarrow x_k$;
      fim se
    fim para
    } Avaliação e detecção da melhor posição individual
    calcule y_s usando os S valores de aptidão $f(y_{ik})$ } Melhor Global
    para i $\leftarrow$ 1 **até** S **faça**
      para j $\leftarrow$ 1 **até** N **faça**
        $v_{ij}^{(t+1)} \leftarrow wv_{ij}^{(t)} + c_1 U_{1j}[0,1](y_{ij}^{(t)} - x_{ij}^{(t)}) + c_2 U_{2j}[0,1](y_{sj}^{(t)} - x_{ij}^{(t)})$
        $x_{ij}^{(t+1)} \leftarrow x_{ij}^{(t)} + v_{ij}^{(t+1)}$
      fim para
    fim para
    } Atualização
  until $f(y_{y_s}) < threshold$
  devolve posição da melhor partícula x e a sua aptidão $f(x)$
fim função

2.5.2 DE - Otimização por evolução diferencial

O algoritmo DE (do inglês *Diferential Evolution*) é um método de busca direta paralela, que utiliza vetores de parâmetros $x_{i,G}$ e N-dimensional para uma população de tamanho NP em cada geração G. NP não se altera durante o processo e o vetor de população inicial é escolhido aleatoriamente. Como regra geral, supõe-se uma distribuição de probabilidade uniforme para todas as variáveis aleatórias.

No algoritmo DE os vetores atuais $\overrightarrow{x}$ são chamados de vetores alvo os quais são usados para gerar novos vetores, também chamados de vetores ruidosos $\overrightarrow{v}$, mediante a adição da diferença ponderada entre dois vetores alvo a um terceiro vetor (operação de *mutação*, vide Eq. 2.4).

$$v_{i,G+1} = x_{r_{1,G}} + F(r_{3,G} - r_{2,G}) \tag{2.4}$$

onde, r_1, r_2, $r_3 \in 1, 2, \ldots, NP$ são índices aleatórios diferentes uns dos outros e diferentes de i e F é uma constante real $\in [0, 2]$ chamada de peso diferencial.

Os vetores ruidosos são então misturados com os parâmetros de um outro vetor predeterminado, compondo os vetores de teste $\overrightarrow{u}$ a partir de uma operação de *crossover* (vide Eq. 2.5). Se o vetor de teste produz um valor melhor da função de custo do que o vetor alvo, então o vetor de teste substitui o vetor alvo na próxima geração (operação de seleção) [15].

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & se\ \ r_j \leq CR \text{ ou } j = l_i \\ x_{ji,G} & se\ \ r_j > CR \text{ ou } j \neq l_i \end{cases} \tag{2.5}$$

onde, $j = 1, \ldots, N$, $r_j \sim U(0, 1)$, CR é a probabilidade de crossover e l_i é um índice aleatoriamente escolhido.

O Algoritmo 2 apresenta o pseudocódigo para o algoritmo de otimização DE.

Algorithm 2 Algoritmo de otimização DE

```
1:  função DE-BÁSICO(NP,CR,F,range,f)
2:      x ← random(range,NP);
3:      fit_x ← f(x);
4:      enquanto critério de parada não for satisfeito faça
5:          para i ← 1 até NP faça
6:              v_{i,G+1} ← Mutação (x_{i,G},F);
7:              u_{i,G+1} ← Crossover (x_{i,G},v_{i,G+1},CR);
8:          fim para
9:          fit_u ← f(u);
10:         para i ← 1 até NP faça                          }
11:             se fit_u(i) < fit_x(i) então                }
12:                 x_{i,G} ← u_{i,G+1};                    }
13:             senão                                       }  Seleção
14:                 x_{i,G} ← x_{i,G};                      }
15:             fim se                                      }
16:         fim para                                        }
17:     fim enquanto
18:     devolve posição da melhor partícula x e a sua aptidão f(x)
19: fim função
```

Os algoritmos anteriores são utilizados para otimizar o mecanismo de flexão/extensão dos dedos imple-

mentados do projeto de mão robótica do trabalho atual. Cabe salientar que o algoritmo de otimização GA foi utilizado da mesma maneira para otimizar o mecanismo em um trabalho anterior [7]. Adicionalmente o algoritmo PSO também é utilizado para sintonizar o controlador de impedância. O procedimento destas tarefas são descritos em capítulos conseguintes.

2.6 Estado da Arte sobre Mão Robóticas Biomiméticas

Nesta subseção vão ser apresentados vários trabalhos correlatos sobre mãos robóticas biomiméticas apresentando uma comparação tendo em conta várias características das diferentes abordagens.

No intuito de descrever a configuração mecânica de cada mão é utilizada uma variação da nomenclatura descrita em Tavakoli *et al.* [10]. Nessa nomenclatura um número e símbolo é assinado para cada dedo e o movimento que eles exercem. Cada dedo é identificado por um número exceto para o dedo polegar o qual é representado pela letra maiúscula 'P'. Os dedos indicador, médio, anular e mínimo são representados pelos números 2, 3, 4 e 5, respetivamente. Cada dedo contém movimentos de adução/abdução e flexão/extensão, representados pelos símbolos 'A' e 'F' respetivamente, os quais são dispostos na frente de cada dedo.

Nesta nomenclatura [+] quer dizer que um novo atuador tem sido adicionado e [,] significa que a atuação das juntas estão acopladas e são realizadas com um único atuador. Existem várias combinações de possíveis movimentos de adução/abdução e flexão/extensão para cada dedo, neste trabalho as seguintes combinações serão usadas.

- O dedo só tem movimentos de flexão/extensão (e.g. para o dedo indicador seria representado da seguinte forma: "F1").
- O dedo contem movimentos de flexão/extensão e os movimentos de adução/abdução são feitos manualmente (e.g. para o dedo polegar seria representado da seguinte forma: "FP Manual-AP").
- Os movimentos de um dedo estão acoplados aos movimentos do outro (e.g. os movimentos de flexão/extensão dos dedos indicador, médio, anular e mínimo estão acoplados, isto é representado por "F2,3,4,5"). Uma variação desta combinação é quando os movimentos de flexão/extensão e adução/abdução estão acoplados (e.g. para o dedo polegar isto seria representado por: "AP,FP").
- Ambos os movimentos de flexão/extensão e adução/abdução são efetuados de maneira independente (e.g. para o dedo indicador isto seria representado por "A2+F2").

A comparação do estado da arte é apresentada na Tabela 2.1, onde as linhas apresentam os diferentes trabalhos e as colunas descrevem as características de cada um.

A coluna *Ano* apresenta o ano no qual o tipo de controle apresentado foi realizado. A característica de *Tipo de Controle* apresenta qual método de controle é realizado no trabalho para atingir os movimentos da mão e, portanto, para realizar o agarre. A coluna de *Transmissão dos dedos* apresenta o mecanismo utilizado para que os dedos cumpram com os movimentos de flexão/extensão, e.g. mecanismo rígido de quatro elos. A coluna *Atuadores* descreve o tipo de acionamento com que a mão efetua os seus diferentes movimentos, e.g. motor de CC, motor brushless, pneumático, etc. A característica de *Sensores* expõe

os sensores que contem a mão robótica, e.g. de posição, torque, etc. A coluna *Plataforma* refere-se ao hardware onde é realizado o controle do sistema. A coluna *Frequência de Atualização* descreve o tempo em que um ciclo de controle é realizado. A característica *Configuração* descreve a configuração mecânica de cada mão, e.g. algumas tem mais atuadores, menos dedos, etc. Finalmente, a coluna *GDLs* apresenta o número de graus de liberdade do robô. É importante destacar que o número de graus de atuação (número de motores) pode diferir do número e graus de liberdade dependendo do tipo de mecanismo de transmissão usado.

Tabela 2.1: Comparações dos trabalhos correlatos de desenvolvimento de mãos robóticas

Autor	**Ano**	**Tipo de Controle**	**Transmissão dos Dedos**	**Atuadores**	**Sensores**	**Plataforma**	**Frequência de Atualização (Hz)**	**Configuração**	**GDLs**
KITECH-Hand [16]	2017	Controle PID de corrente	Redução de engrenagens	16 Motores de CC	16 Sensores de posição (potenciômetros); 16 Sensores de corrente.	Sistema multiprocessador baseado Micro-controladores (16 MCU)	333	AP + FP + A2 + F2 + A3 + F3 + A4 + F4	16
Calderon *et al.* [17]	2017	Controle PID de Posição	Mecanismo de 4 elos	5 Motores de CC	5 Sensores de força	Computador	N/A	FP + F2 + F3 + F4 + F5 + AP-Manual	5
Jeong *et al.* [18]	2017	Redes neurais para a estimação de posição e controle de força com PID	Mecanismo de 4 elos	6 Motores de CC	5 Sensores de posição	Computador	N/A	AP + FP + A2 + F2 + F3 + F4 + F5	6
Wang *et al.* [19]	2010	Controle de Impedância com loop interno de controle PID da posição	Mecanismo de 4 elos	5 Motores de CC	5 Encoders magnéticos incrementais; 5 sensores de posição de efeito de Hall; 5 Sensores de torque.	Sistema multiprocessador baseado em DSP (2 DSP)	10 - 40	AP,FP + F2 + F3 + F4 + F5	5
LMS Hand [20],[21]	2012	Controle de Força com estimador de força baseado em redes neurais e loop interno PID de posição	Transmissão por fios	16 Motores de CC	16 Encoders incrementais para os motores, 16 encoders absolutos para as juntas	Computador	50	AP + FP + A2 + F2 + A3 + F3 + A4 + F4	16
Lee *et al.* [22]	2009	Controle PD hibrido de posição/força	Mecanismo de 4 elos	9 Motores de CC	9 Encoders incrementais, 4 sensores de força resistivos	Computador	Simulação	AP + FP + A2 + F2 + A3 + F3 + A4 + F4 + Palma da mão	9
HIT/DLR Prosthetic Hand [23],[24]	2006	Controle de Impedância com loop interno de controle PD da posição	Mecanismo de 4 elos	3 Motores de CC	3 encoders de quadratura, 3 sensores de posição de efeito de hall, 3 sensores de torque, 3 sensores de força na ponta dos dedos	Sistema multiprocessador baseado em DSP (2 DSP)	1000	AP,FP + F2 + F3,4,5	3
HIT/DLR Prosthetic Hand II [25],[26]	2009	Controle de Impedância com loop interno de controle PD da posição	Transmissão por fios metálicos	15 Motores Brushless de CC	15 sensores de posição, 15 sensores de torque, 5 sensores de força na ponta dos dedos, 15 encoders absolutos de efeito hall, 10 sensores de temperatura, sensores táteis.	Sistema multi-processador baseado em DSP&FPGA (6 DSP e 6 FPGA)	-	AP + FP + A2 + F2 + A3 + F3 + A4 + F4 + A5 + F5	20
ISR-SoftHand [27]	2014	Controle PD de posição	Transmissão por fios	3 Motores de CC Dynamixel AX-12	Vide manual AX-12	Computador	N/A	FP + F2 + F3,4,5 + AP-Manual	10
KNU Hand [28],[29]	2009	Controle de força realimentado com detector de deslizamentos	Transmissão por mola espiral e mecanismo de 4 elos	2 Motores de CC	2 sensores de posição resitivos	DSP	-	FP+F2,3,4,5	16

Com base na Tabela 2.1, pode-se observar que apenas um trabalho correlato usou FPGAs para implementar o controlador [26]. Note-se que nesse trabalho utiliza-se uma mão robótica de alta complexidade

(20 GDLs e 15 motores), assim como uma instrumentação elaborada que justifica o uso de 6 DSPs e 6 FPGAs. Por outro lado, os trabalhos mais similares com a presente proposta são os trabalhos de Wang *et al.* [19], Lee *et al.* [22], os quais usam o mesmo mecanismo de transmissão de 4 elos e uma configuração e instrumentação similar. A KITECH-Hand [18], comercialmente conhecida como Alegro-Hand, mesmo não sendo um trabalho plenamente igual, tem similitudes grandes com o projeto atual, como o uso de sensores de corrente nos motores para estimar força. Desta forma, estes últimos três trabalhos serão usados para fins de comparação.

2.7 Conclusões do Capitulo

Neste capitulo foram apresentados os tópicos referentes ao projeto de pesquisa de mão robótica biomimética no intuito de expor o baseamento teórico pertencente ao trabalho. Esses tópicos são necessários para a compreensão do desenvolvimento do trabalho apresentados nos seguintes capítulos.

Por fim, foi apresentado um estado da arte apresentando trabalhos correlatos para possíveis comparações entre os resultados finais do projeto e os resultados reportados na literatura científica.

Capítulo 3

Projeto da Mão Robótica

Uma mão robótica biomimética, como indica o nome, é um robô que pretende mimetizar uma mão humana real. Uma mão humana dispõe de 31 músculos, que fornecem o movimento de 19 articulações e pelo menos 25 GDLs [30]: 4 em cada dedo, 3 para flexão-extensão e um para abdução-adução; o polegar é o dedo mais complicado com 4 GDLs incluindo a sua caraterística de oponibilidade, restando 3 GDLs para a rotação e translação do pulso (Vide Fig. 3.1).

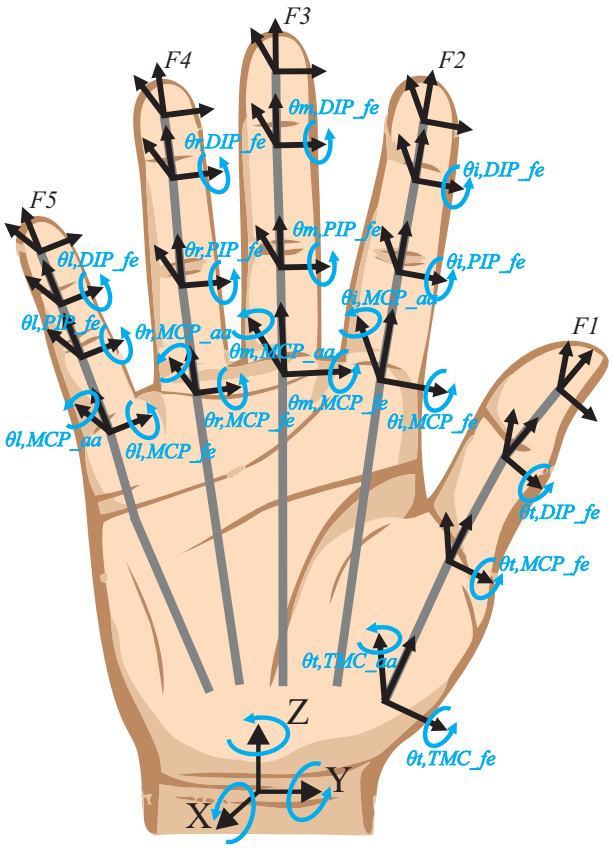


Figura 3.1: GDLs de uma mão humana. onde l - mínimo, r - anelar, m - médio, i - indicador e t - polegar. Fonte: o autor

Naturalmente, a grande quantidade de GDLs presente na mão representa um problema complexo ao reproduzir uma mão robótica biomimética, devido a que existe pouco espaço físico entre as juntas para um modelo cinemático tão complexo, além de gerar um alto custo computacional para o controle do mesmo.

Essa complexidade se vê quando pretende-se interpretar cada GDLs como independente um do outro, no entanto, a faixa de movimentos que a mão pode exercer nas suas juntas são limitados [30]. Essas limitações podem-se expressar matematicamente através de equações. Cobos [31] especifica essas limitações que tem alguns GDLs da mão com outros, mas também, elenca o grau de importância que tem cada grau de liberdade ao reproduzir os movimentos necessários para realizar agarres. Essa classificação foi feita através de uma análise estatístisca de componentes principais (ACP) de 200 amostras, o qual resultou na Tabela 3.1, onde se especifica os 10 GDLs mais importantes para agarres de precisão e potência (Vide Seção 2.3).

Tabela 3.1: 10 GDLs de maior importância para efetuar agarres de precisão e potência conforme ao analise de componentes principais. Na coluna *Uso no projeto* se indica as articulaões que serão usadas neste projeto com *. A Fig. 3.1 apresenta uma representação gráfica das articulações.

No.	Dedo	Articulação	Uso no projeto
1	Polegar	θ_{t,TMC_aa}	*
2	Polegar	θ_{t,TMC_fe}	*
3	Indicador	θ_{i,MCP_aa}	*
4	Indicador	θ_{i,PIP_fe}	*
5	Médio	θ_{m,MCP_fe}	*
6	Médio	θ_{m,PIP_aa}	
7	Anelar	θ_{r,MCP_fe}	*
8	Anelar	θ_{r,PIP_aa}	
9	Mínimo	θ_{l,MCP_fe}	*
10	Mínimo	θ_{l,PIP_aa}	

Fonte: Cobos [31].

A Tabela 3.1 pode ser usada para o desenvolvimento de modelos simplificados de mãos. Nesse sentido, no intuito de reproduzir a cinemática de uma mão, a independência de alguns GDLs poderiam ser dispensadas e o número de graus de liberdade desejado determinaria quais GDLs seriam conservados segundo o grau de relevância. Desta forma, os GDLs dispensados seriam matematicamente relacionados com outros. Desta maneira, obtém-se uma mão com menos graus de atuação (ou GDAs que inferem GDLs que são independentes) conservando a mesma quantidade de GDLs. Cobos estimou o erro que produz essas simplificações ao reproduzir os movimentos de uma mão real e observou que uma simplificação de uma mão com 7 GDLs produz um erro menor ao 10%, conseguindo atingir a maioria dos agarres. As relações necessárias para a simplificação de 7 GDLs estão indicadas na Tabela 3.2.

Essas relações são apresentadas na Tabela 3.2

Tabela 3.2: Relações intrafâlangicas para mão simplificada de 7 GDLs.

Polegar	Indicador	Médio	Anelar	Minimo
θ_{t,TMC_aa}				
	θ_{i,MCP_aa}			
$\theta_{t,MCP\_fe} \approx \frac{4}{5}\theta_{t,DIP\_fe}$	$\theta_{i,MCP\_fe} \approx \frac{4}{3}\theta_{i,PIP}$	$\theta_{m,MCP\_fe} \approx \frac{4}{3}\theta_{m,PIP}$	$\theta_{r,MCP\_fe} \approx \frac{4}{3}\theta_{r,PIP}$	$\theta_{l,MCP\_fe} \approx \frac{4}{3}\theta_{l,PIP}$
$\theta_{t,DIP}$	$\theta_{i,PIP} \approx \frac{3}{2}\theta_{i,DIP}$	$\theta_{m,PIP} \approx \frac{3}{2}\theta_{m,DIP}$	$\theta_{r,PIP} \approx \frac{3}{2}\theta_{r,DIP}$	$\theta_{l,PIP} \approx \frac{3}{2}\theta_{l,DIP}$
	$\theta_{i,DIP}$	$\theta_{m,DIP}$	$\theta_{r,DIP}$	$\theta_{l,DIP}$

Fonte: Cobos [31].

Neste capitulo apresenta-se a análise cinemática do modelo simplificado de 7 GDAs o qual é a base do projeto da mão robótica deste trabalho, o mecanismo de transmissão que sub-atua os GDLs relacionados conforme as restrições apresentadas, a otimização do mecanismo usando algoritmos bioinspirados, a análise estatística que demonstra a superioridade obtidas pelos algoritmos PSO e DE, atingindo um erro de 0.00266%. Apresenta-se, também, a inclusão do dedo robótico no sistema completo de mão robótica de 7 GDLs proposto, exibindo os métodos de manufatura e prototipagem que resultam na plataforma de mão robótica biomimética.

3.1 Análise Cinemática do Modelo Simplificado de Mão - 7GDAs/16GDLs

A Fig. 3.7 apresenta a abordagem proposta para a estrutura cinemátoca da mão robótica. Totalizando, a mão robótica proposta contem 16 juntas ou GDLs, no entanto apenas 7 dessas juntas são atuadas (GDAs). As juntas escolhidas para serem atuadas aprensentam-se com estrela asterisco (*) na Tabela 3.1 (destacadas em vermelho na Fig. 3.7). A sub-atuação é realizada mediante um mecanismo que simula as restrições vistas na Tabela 3.2, onde, dependendo do estado da junta θ_{MCP_fe} de cada dedo, as juntas continuas do mesmo dedo realizaram um movimento proporcional.

Com o intuito de fabricar um protótipo visando ser o mais similar possível a uma mão humana, decidiu-se usar 5 dedos. Cabe salientar, que os movimentos de adução-abdução (juntas θ_{i,MCP_aa} e θ_{t,MCP_aa}) proporcionam mais agilidade à mão robótica. Por exemplo, a junta θ_{t,MCP_aa} propicia a caraterística de oponibilidade do dedo polegar o qual determina é importante para realizar tarefas de agarres.

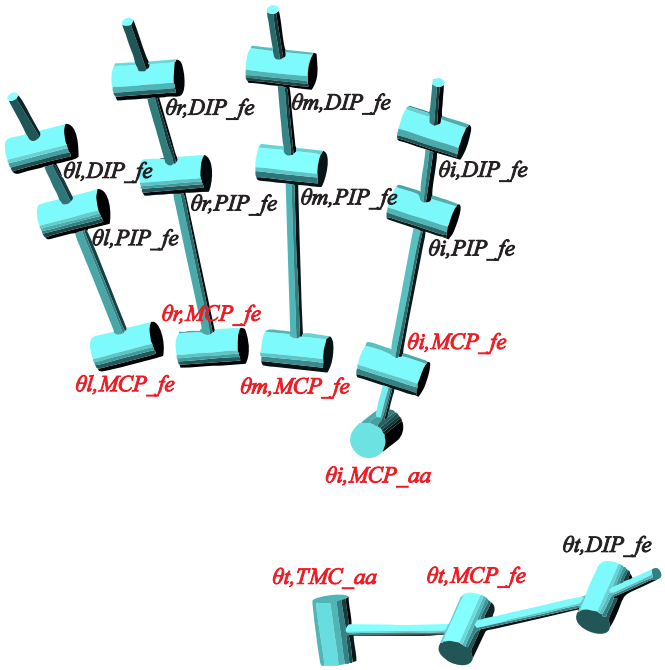


Figura 3.2: Estrutura Cinemática da Mão Robótica Simplificada de 7 GDAs. A estrutura tem 7 GDAs e 16 GDLs. Os componentes identificados com letras vermelhas são os GDAs. Fonte: o autor.

O conjunto de movimento e os comprimentos das falanges da mão estão descritas na Tabela 3.3. Esses valores cinemáticos foram determinados, tomando como base, as medidas típicas da estrutura de uma mão.

Tabela 3.3: Parâmetros cinemáticos da mão robótica.

Junta	Comprimento das falanges de cada dedo [mm]					Conjunto do Mov. [graus]
	Polegar	Indicador	Médio	Anelar	Mínimo	
MCP_aa	-	17	-	-	-	$0-15$
TMC_aa	38	-	-	-	-	$0-90$
MCP_fe	37.6772	45.7788	51.3602	47.6086	37.6772	$0-90$
PIP_fe	-	25.7549	30.3006	29.5180	20.8410	$0-67.5$
DIP_fe	20.8410	18.2057	20.0239	19.9088	18.3668	$0-45$

Fonte: o autor

As geometrias e limitações das juntas repassadas no quadro anterior determinam a área de trabalho da mão robótica. A área de trabalho de um manipulador é o volume total varrido pelo órgão terminal do mesmo (no nosso casso, a ponta dos dedos), quando executam-se todas as possíveis posições angulares das juntas [32]. Com o intuito de esclarecer os alcances da mão robótica proposta nesta abordagem, uma análise qualitativa da área de trabalho foi realizada. A área de trabalho é facilmente ilustrada como os lugares geométricos dos pontos onde as pontas dos cinco dedos atingem movimentos validos, vide Fig 3.3. Nessa figura, pode-se observar que todos os dedos têm vários pontos comum com o polegar, validando a sua caraterística oponível.

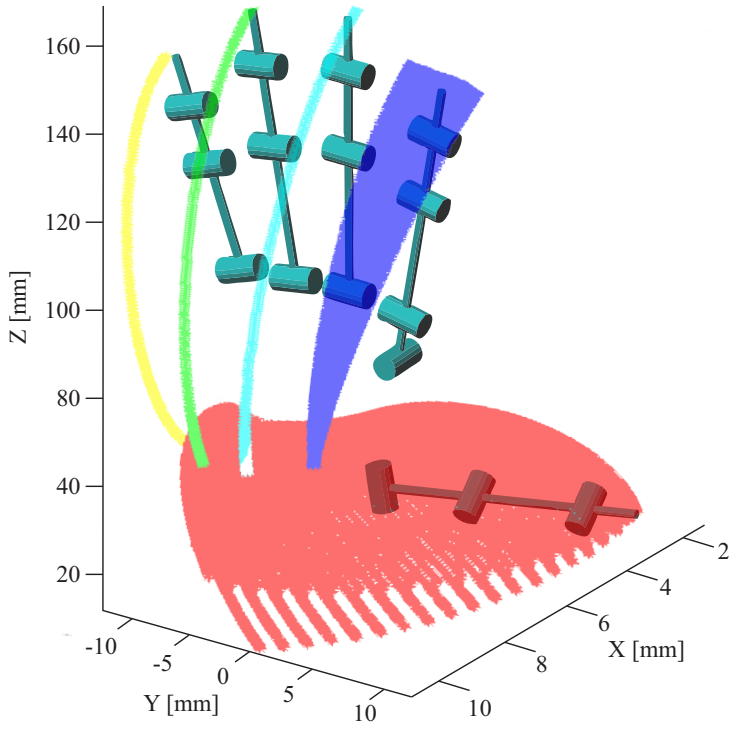


Figura 3.3: Área de trabalho da mão robótica. Fonte: o autor.

Na figura anterior, também pode se observar que a área de trabalho de alguns dedos, são simples trajetórias, isto é, porque o dedo tem juntas que são sub-atuadas. A seguinte seção aprofunda o projeto, a modelagem e a otimização do mecanismo que realiza essa tarefa de sub-atuação.

3.2 Modelamento e Otimização do Mecanismo de Dedo Robótico

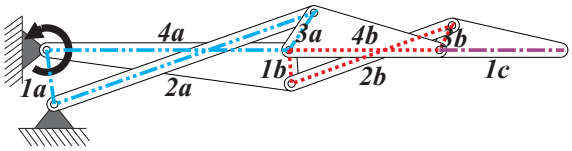
Para cumprir com as relações que devem ter uns ângulos com os outros podem ser empregados diferentes tipos de mecanismos de transmissão de movimento. O tipo de transmissão que deve ser usado depende da abordagem e do posicionamento dos atuadores no robô. Basicamente existem duas possibilidades de posicionamento dos atuadores ao desenvolver uma mão robótica: a) posicionar os atuadores diretamente nas juntas; isto implica o uso de atuadores com tamanho reduzido instalando-os na palma da mão ou nas mesmas falanges [17, 18, 16, 33, 34, 22, 26, 35, 25, 23, 24] e, b) posicionar o atuadores fora da palma, o que permite o uso de atuadores maiores e com um torque maior [20, 27, 36].

Nesse trabalho a primeira abordagem é usada através de mecanismos rígidos de quatro barras acoplados emulando os movimentos de flexão/extensão dos dedos com 1 GDA. O restante desta seção descreve o projeto e modelamento cinemático do mecanismo, a exploração de técnicas bioinspiradas para otimização dos elos do mecanismo proposto, que imita a trajetória que um dedo executa ao se flexionar (fechar) e ao se estender (abrir), a análise estatística e uma comparação do resultado da otimização desses algoritmos. Finalmente o projeto CAD da mão robótica e a sua validação.

3.2.1 Modelamento Cinemático do Mecanismo do Dedo

Os movimentos de flexão/extensão do dedo são atingidos através de dois mecanismos de 4 elos acoplados. A Fig. 3.4 apresenta o duplo mecanismo acoplado de 4 elos onde o conjunto de elos: $\{1a, 2a, 3a, 4a\}$ representam o primeiro mecanismo e o conjunto: $\{1b, 2b, 3b, 4b\}$ representam o segundo mecanismo de 4 elos acoplado ao primeiro.

(a)



(b)

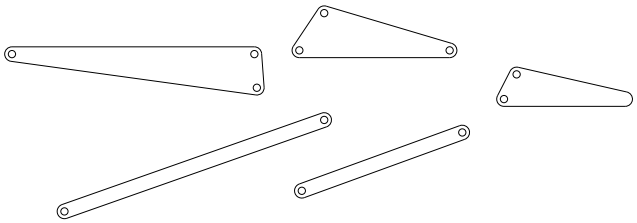


Figura 3.4: Mecanismo de 4 elos. (a) Mecanismo acoplado, pode-se observar os dois conjuntos que conformam o dedo completo. (b) Vista explodida do mecanismo. Fonte: o autor.

Na Fig. 3.5 pode-se observar a estrutura do mecanismo utilizado, a trajetória gerada pela ponta do dedo ao executar o movimento de flexão e o estado do dedo quando está completamente estendido e flexionado. O mecanismo proposto gerada uma trajetória única, contudo, essa trajetória deve ser similar à trajetória gerada por um dedo real ao realizar os mesmos movimentos fe.

Cobos [31] apresenta também as relações interfalângica mais comuns ao executar vários tipos de agarres, as quais foram obtidas a partir de vários experimentos de agarres, usando luvas com sensores inerciais. Essas relações estão listadas na Tabela 3.2, onde se observa que os movimentos de flexão/extensão de cada dedo podem ser simplificados atuando uma única junta. As juntas estão elencadas na descrição da Fig 3.1, por simplicidade, no resto dessa seção a seguinte nomenclatura será usada para citar as juntas metacarpofalângea, interfalângica proximal e interfalângica distal: $\theta_1, \theta_2, \theta_3$ respetivamente (vide 3.5.b). Da Tabela 3.2 pode se inferir as seguintes equações

$$\begin{aligned} \theta_2 &= \frac{3}{4}\theta_1 \\ \theta_3 &= \frac{1}{2}\theta_1 \end{aligned} \tag{3.1}$$

das Eq. 3.1 decorre a trajetória que o mecanismo deve percorrer. Contudo, atingir a trajetória alvo com o mecanismo resulta num trabalho não trivial dado que depende do ajuste dos comprimentos dos elos.

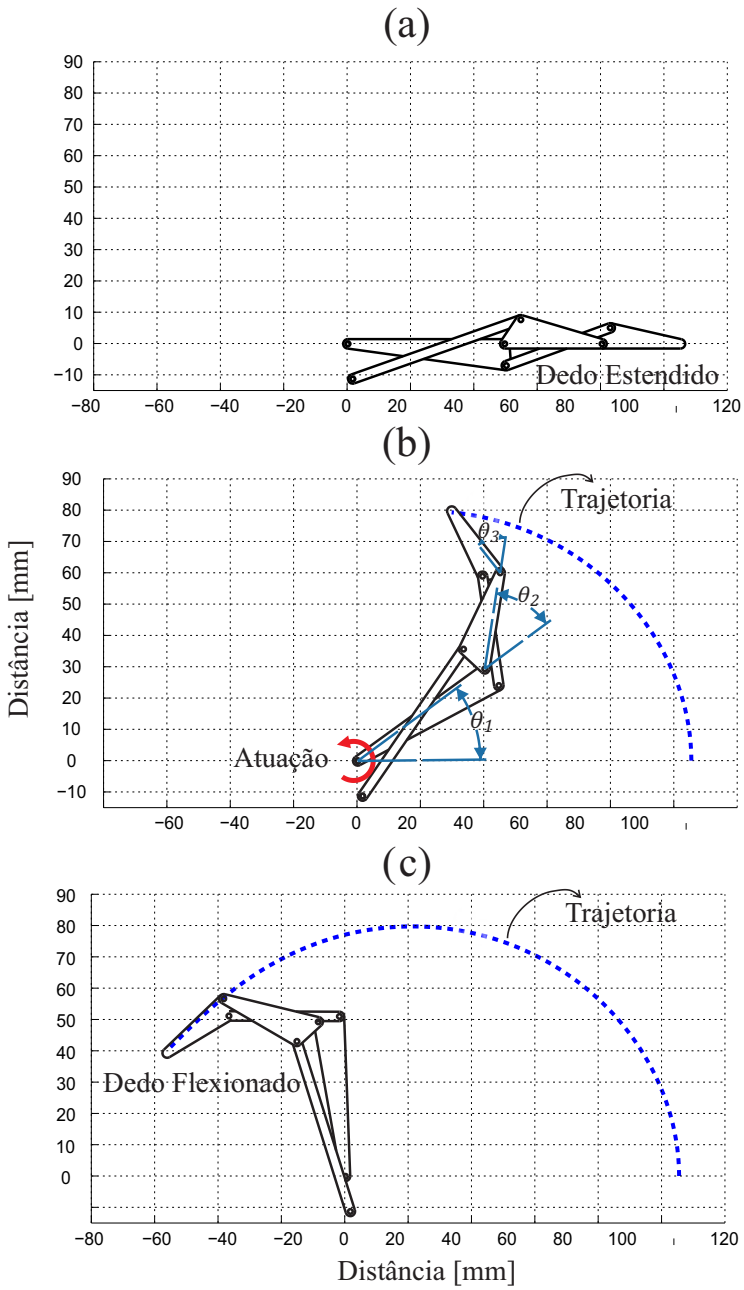


Figura 3.5: Mecanismo de 4 elos acoplado que emula os movimentos do dedo. (a) Posição do dedo completamente estendido, nesta posição o mecanismo está em repouso; devido à natureza do mecanismo; nessa posição o mesmo está auto travado e só permite movimentos em sentido anti-horário. (b) O dedo está em movimento, a única junta que é atuada no mecanismo é a que está sinalizada com a seta vermelha. (c) Posição do dedo completamente flexionado; nesta posição entra a função de auto travamento do mecanismo, onde não é mais permitido o movimento anti-horário. Fonte: o autor.

A Fig. 3.6 apresenta a trajetória alvo calculada com a Eq. 3.2 e a trajetória gerada pelo mecanismo proposto.

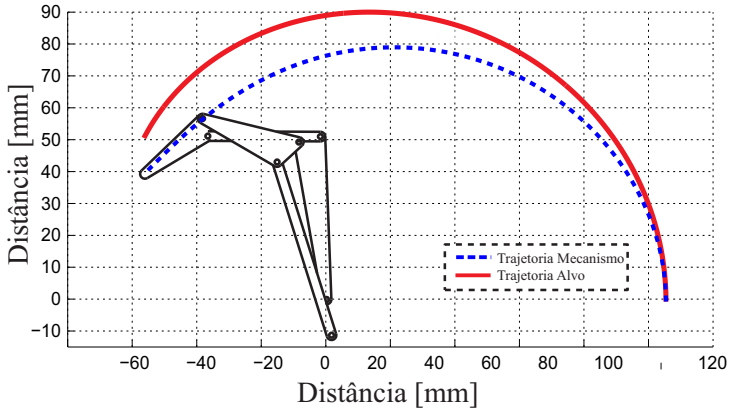


Figura 3.6: Trajetória alvo em contraste com a trajetória gerada pelo mecanismo de elos com comprimento aleatório. Saliente-se que o comprimento dos elos que representam as falanges proximal, medial e distal decorrem da média do comprimento do dedo indicador em homens adultos.

Fonte: o autor.

$$Traj\_alvo = \langle (\ elo_{4a}\cos\theta_1 + elo_{4b}\cos(\theta_1+\theta_2) + elo_{1c}\cos(\theta_1+\theta_2+\theta_3)\ )\hat{i}, \\ (\ elo_{4a}\sin\theta_1 + elo_{4b}\sin(\theta_1+\theta_2) + elo_{1a}\sin(\theta_1+\theta_2+\theta_3)\ )\hat{j}\rangle \tag{3.2}$$

Pode se ver na Fig. 3.6 que a trajetória atingida pelo mecanismo está longe de se se aproximar à trajetória alvo. Por outro lado, os elos que representam as falanges do dedo: $\{4a, 4b, 1c\}$ são fixos, portanto, o comprimento dos elos restantes definem a trajetória efetuada. Portanto, a fim de ajustar a trajetória do mecanismo à trajetória alvo é necessário calcular o modelo cinemático do mecanismo.

No intuito de calcular o modelo cinemático do mecanismo, o mesmo é desacoplado, dessa forma, é calculado somente o primeiro conjunto de 4 elos $\{1a, 2a, 3a, 4a\}$, o que simplifica o modelamento. Após ter o modelo do primeiro conjunto do mecanismo, para obter o mecanismo completo, só é necessário calcular o modelo do segundo mecanismo desacoplado e transladar o modelo calculado. A Fig. 3.7 apresenta o mecanismo desacoplado para o modelamento, assim, sabe-se que a entrada do mecanismo é o ângulo θ_1 e os valores que são de interesse calcular são o ponto P_{f1} e o ângulo θ_2. Cabe salientar para fato que, embora o elo $4b$ não fazer parte do mecanismo em questão, a sua consideração é mister para calcular θ_2. Os valores necessários para resolver o mecanismo são L_{f1}, $P_{a2}\hat{i}$, $P_{a2}\hat{j}$, L_{a1} e L_{b1}.

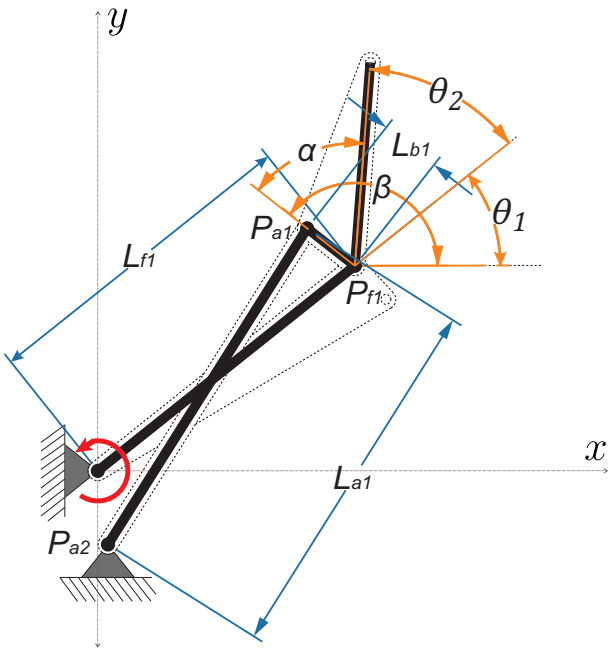


Figura 3.7: Representação gráfica do modelo cinemático do mecanismo de 4 barras. Fonte: o autor.

Num primeiro momento, o calculo de P_{f1} é realizado da seguinte maneira

$$P_{f1} = L_{f1}\langle \cos\theta_1 \hat{i}, \sin\theta_1 \hat{j} \rangle \tag{3.3}$$

Com o intuito de determinar θ_2, calcula-se primeiro a interseção entre os elos $2a$ e $3a$ (ponto P_{a1})

$$\begin{aligned}\left(\frac{(D_a - D_b)^2}{(E_a - E_b)^2} + 1\right) x^2 + \left(D_a - \frac{E_a\,(D_a - D_b)}{E_a - E_b} + \frac{2\,(D_a - D_b)\,(F_a - F_b)}{(E_a - E_b)^2}\right) x + \\ \left(F_a + \frac{(F_a - F_b)^2}{(E_a - E_b)^2} - \frac{E_a\,(F_a - F_b)}{(E_a - E_b)}\right) = 0\end{aligned} \tag{3.4}$$

$$y = \frac{(D_b - D_a)x + F_b - F_a}{E_a - E_b} \tag{3.5}$$

onde, x é a posição do ponto P_{a1} no eixo $\hat{i}$ e y na Eq. 3.5 é a posição no eixo $\hat{j}$, as variáveis $D_a = -2P_{a2}\hat{i}$, $E_a = -2P_{a2}\hat{j}$, $F_a = (P_{a2}\hat{i})^2 + (P_{a2}\hat{j})^2 - {L_{a1}}^2$, $D_b = -2P_{f1}\hat{i}$, $E_b = -2P_{f1}\hat{j}$ e $F_b = (P_{f1}\hat{i})^2 + (P_{f1}\hat{j})^2 - {L_{b1}}^2$.

Pode-se observar que na Eq. 3.4 é uma equação de quadratica. Da Eq. 3.4 infere-se em dois possíveis soluções para o ponto P_{a1}, por restrições do mecanismo, a solução usada sempre é aquela com o valor maior de y. Resultados complexos também são um problema, isto ocorre quando os valores no modelo não atingem uma solução real do mecanismo.

As Eq. 3.4 e 3.5 calculam o ponto P_{a1}, logo, o ângulo θ_2 é calculado da seguinte maneira

$$\theta_2 = \beta - \theta_1 - \alpha \tag{3.6}$$

onde, $\alpha = \tan^{-1}\left(\frac{(P_{a1}\vec{-}P_{f1})\hat{j}}{(P_{a1}\vec{-}P_{f1})\hat{i}}\right)$ e $\beta \leftarrow \alpha$ quando $\theta_1 = 0$.

Consequentemente, após conhecer P_{f1} e θ_2, o cálculo do segundo mecanismo é determinado usando θ_2 como entrada, usando as mesmas equações anteriores e mudando os comprimentos dos elos com os comprimentos do seguinte mecanismo desacoplado. Depois de calcular este segundo mecanismo, o mecanismo completo é acoplado. Deste modo, obtém-se o ponto P_{f2}, P_{f3} (P_{f3} é o posicionamento cartesiano da ponta do dedo) e o ângulo θ_3, e, por conseguinte, inferem-se as relações

$$\forall\theta_1 \in [0, \pi/2] : \{\ P_{f3}(\theta_1)\ ,\ \theta_2(\theta_1)\ ,\ \theta_3(\theta_1)\ \} \tag{3.7}$$

3.2.2 Definição do Problema de Otimização

Nesta subseção, os algoritmos bioinspirados DE, PSO e GA são implementados para otimizar o mecanismo, de modo que consiga mimetizar os movimentos que realiza um dedo real. Como foi descrito na Seção 2.5, o objetivo dos algoritmos bioinspirados é explorar os valores das variáveis de decisão que consigam minimizar uma função custo.

Com isso em mente, o primeiro passo é escolher as variáveis de decisão, no presente caso, estas seriam os valores necessários para resolver o mecanismo L_{f1}, $P_{a2}\hat{i}$, $P_{a2}\hat{j}$, L_{a1} e L_{b1}. Dado que, os comprimentos dos elos que representam as falanges dos dedos são fixos, as variáveis que devem ser otimizadas são $P_{a2}\hat{i}$, $P_{a2}\hat{j}$, L_{a1} e L_{b1}. Adicionalmente, dado que o mecanismo completo contém dois mecanismos de 4 elos acoplados (Vide Fig. 3.4), são necessárias 8 variáveis de decisão..

O segundo passo para abordar o problema de otimização é definir a função custo. Do modelamento revisto na subseção anterior, obtem-se as equações que permitem criar uma trajetória de pontos P_{f3} no espaço do trabalho e o comportamento dos ângulos θ_2 e θ_3 no espaço da tarefa. O mecanismo deve atingir a trajetória alvo apresentada na Fig. 3.6 e os ângulos θ_2 e θ_3 devem ter o mesmo comportamento manifestado nas Eq. 3.1. De tal forma que, no presente caso, propõe-se dois enfoques diferentes para a função custo: a) comparar a trajetória de pontos produzida pelo modelo cinemático com a trajetória alvo (vide Eq. 3.2) e, b) comparar os comportamentos dos ângulos computados pelo modelo cinemático com o comportamento descrito na Eq. 3.1.

Neste trabalho utilizou-se o Erro Médio Quadrático (MSE) para quantificar ponto a ponto o grau de semelhança entre as trajetórias. O MSE é calculado de acordo à seguinte equação

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{3.8}$$

onde $\hat{Y}_i$ é a trajetória ou comportamento alvo, Y_i é a trajetória ou comportamento gerado pelo mecanismo e n é o número de pontos da trajetória.

Saliente-se que, dada a natureza do mecanismo e o modelamento cinemático, nem todas a combinações de comprimento dos elos resultam numa solução real do mecanismo. Nas funções custo, para casos onde o

mecanismo não tem solução real, uma penalidade alta é atribuída ao erro, com o intuito de que esses casos não sejam mantidos em consideração.

3.2.3 Resultados

Com o intuito de realizar análise estatística dos resultados obtidos vários experimentos foram realizados para todos os algoritmos de otimização bioinspirada usados (PSO, DE e GA). As condições experimentais para cada algoritmo estão listadas na Tabela 3.4.

Tabela 3.4: Condições experimentais dos algoritmo PSO, DE e GA para a otimização do mecanismo do dedo. *$[P_{a2,1}\hat{i},\ P_{a2,2}\hat{i},\ P_{a2,1}\hat{j},\ P_{a2,2}\hat{j},\ L_{a1,1},\ L_{a1,2},\ L_{b1,1},\ L_{b1,2}]$

Parâmetro	Valor
Numero de Partículas	30
Dimensionalidade	8
Limite inferior de busca*	$[0, 0, -9, -10, 10, 10, 2, 2]$
Limite superior de busca*	$[9, 10, 0, 0, 70, 60, 20, 20]$
Iteração Limite	300
Coeficiente Cognitivo c_1 (PSO)	2.05
Coeficiente Social c_2 (PSO)	2.05
ω (PSO, decrescimento linear)	$[1.0, 0.001]$
Fator de Mutação F (DE)	1.1
Taxa Crossover C (DE)	0.95
Taxa Crossover C (GA)	0.5

Fonte: o Autor

Um ambiente de avaliação foi desenvolvido em Matlab R2017a (9.2.0.538062), permitindo que as trajetórias obtidas pelos algoritmos de otimização possam ser visualmente comparadas. Cada algoritmo foi executado 32 vezes com cada uma das duas abordagens propostas (da trajetória no espaço do trabalho e da trajetória no espaço da tarefa), e, para cada execução foram usado os mesmos valores iniciais (valores onde o mecanismo era resolvível).

3.2.3.1 Primeira Abordagem: Comparação das Trajetórias no Espaço do Trabalho

Como já foi discutido, nessa abordagem foi otimizado o erro entre a trajetória da ponta do dedo gerada pelo mecanismo e trajetória alvo. A Fig. 3.8 retrata a trajetória atingida do melhor resultado dos 32 experimentos realizados para cada algoritmo (PSO, DE e GA). No requadro pode-se observar de maneira mais clara a diferença em cada resultado de cada algoritmo.

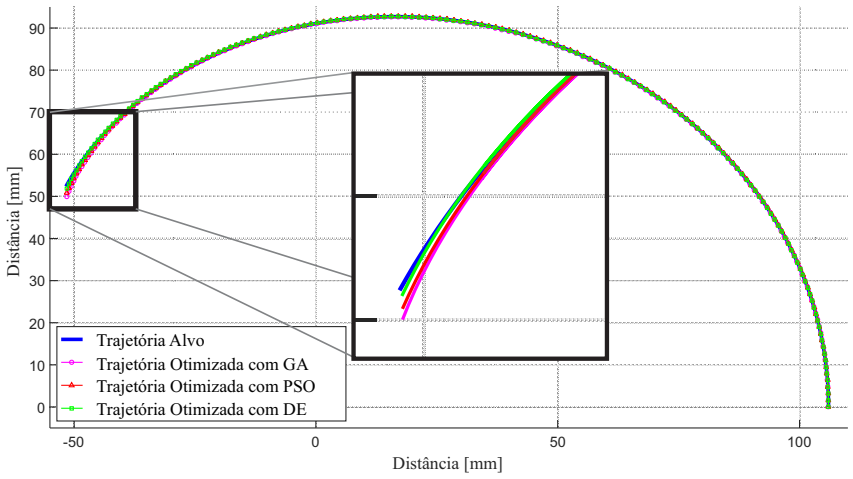


Figura 3.8: Trajetórias otimizadas (melhor valor dos 32 experimentos) visando atingir a trajetória alvo. Fonte: o autor.

A Fig. 3.9 apresenta a convergência do MSE dos três algoritmos usados para otimizar a trajetória do mecanismo, está claro que todos os algoritmos atingiram valores menores que 10^{-2}, e que o algoritmo com o erro menor é o DE.

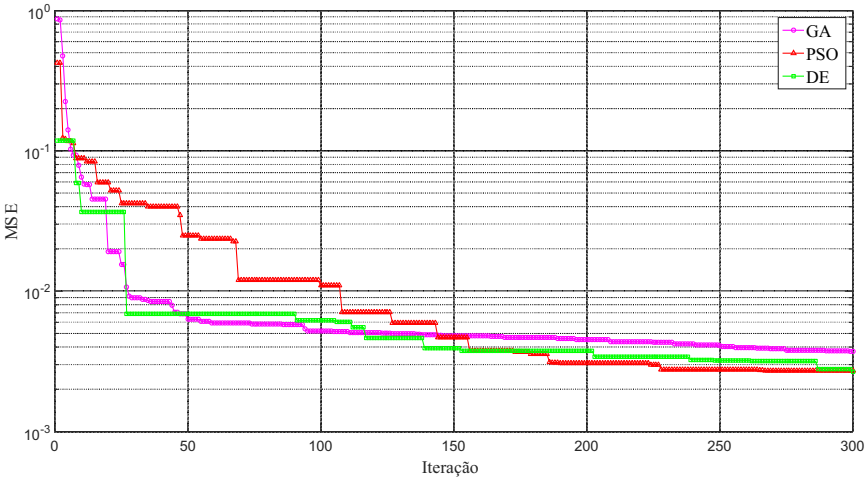


Figura 3.9: Convergência do MSE do melhor resultado de cada algoritmo utilizado visando atingir a trajetória alvo. Fonte: o autor.

3.2.3.2 Segunda abordagem: Comparação do Comportamento no Espaço da Tarefa

Nessa abordagem, usa-se o comportamento dos ângulos θ_2 e θ_3 (vide Fig. 3.5.b) calculados pelo modelo cinemático, usando os valores de decisão deliberado em cada iteração para comparar o seu comportamento com o descrito na Eq. 3.1. A Fig. 3.10 exibe o comportamento através do tempo do melhor resultado dos 32 experimentos

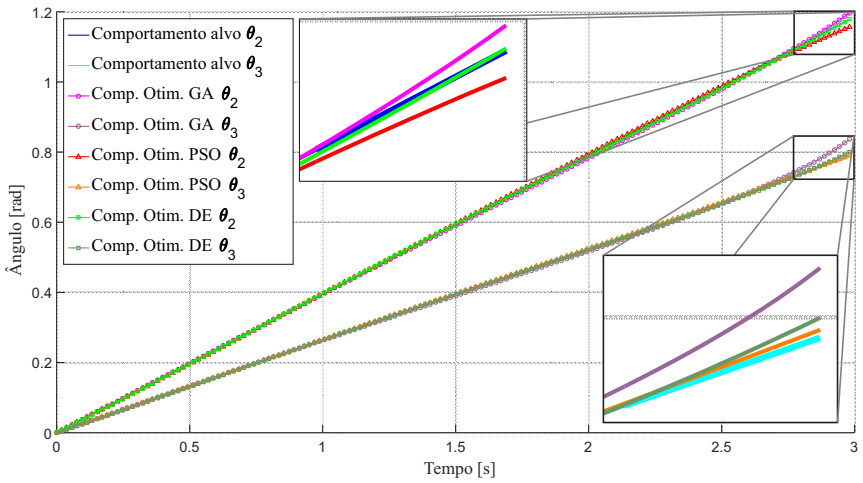


Figura 3.10: Trajetórias otimizadas (melhor valor dos 32 experimentos) visando atingir o comportamento requerido dos ângulos. Fonte: o autor.

A evolução do MSE em cada iteração do melhor resultado dos 32 experimentos é exibida na Fig. 3.11. Pode-se observar que o algoritmo que obteve um melhor resultado foi o PSO.

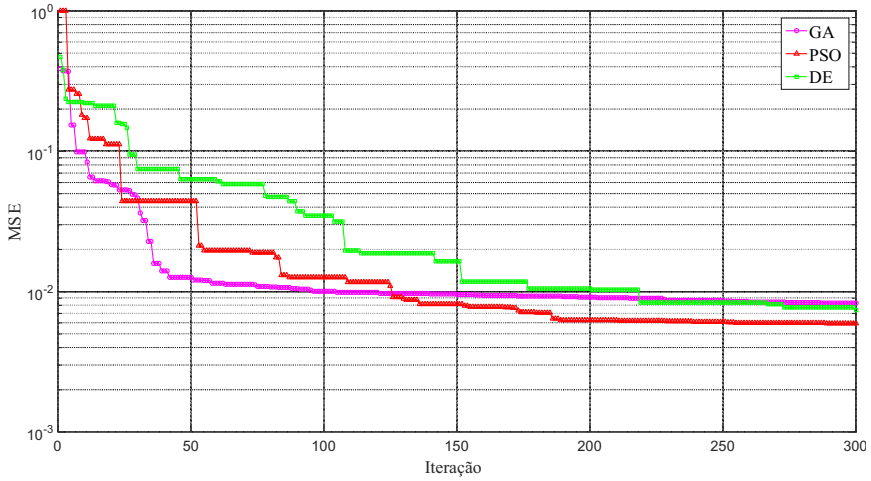


Figura 3.11: Convergência do MSE do melhor resultado de cada algoritmo utilizado visando atingir o comportamento requerido dos ângulos. Fonte: o autor.

3.2.4 Análise Estatística e Discussões

Uma análise estatística foi realizada para comparar a efetividade dos resultados atingidos, isto, com o intuito de efetuar um estudo objetivo. A Tabela 3.5 apresenta os valores estatísticos dos experimentos realizados, observando-se que a função custo da trajetória do dedo com o algoritmo DE tem o melhor (menor) valor de MSE.

Com os dados da tabela anterior realizou-se um teste de análise de significância estatística, isto, para

Tabela 3.5: Dados estatísticos da otimização, μ = média, $\tilde{x}$ = mediana e σ = desvio padrão.

Função Custo	**Algoritmo**	μ **(MSE)**	$\tilde{x}$ **(MSE)**	**min (MSE)**	σ **(MSE)**
Trajetoria do Dedo	GA	0.01520	0.01458	0.00374	0.00853
	PSO	0.00872	0.00528	0.00272	0.00791
	DE	0.00531	0.00482	0.00266	0.00175
Comportamento dos ângulos	GA	0,03495	0.03383	0.01029	0.01784
	PSO	0.02815	0.01093	0.00349	0.04057
	DE	0.01034	0.00727	0.00292	0.01427

Fonte: o autor.

definir qual algoritmo-função obteve o melhor resultado. A análise é simples: o algoritmo com o valor de mediana menor será o melhor resultado, no entanto, isto é certo só, quando os conjuntos de dados provêm de distribuições de probabilidade não normais e diferentes.

Para validar a expressão anterior, é necessário aplicar vários testes não paramétricos. O teste *Kilmogorov-Smirnov* [37] verifica se os conjuntos de dados são normalmente distribuídos. A hipótese nula (H_0) implica que os dados são, de fato, provenientes de distribuições normais, enquanto que, a hipótese alternativa indica que os dados não procedem de uma distribuição normal. Os possíveis resultados são $h = 1$ (que rejeita H_0 com um nível de significância de $\alpha = 0.05$) e $h = 0$ (que valida H_0 com um nível de significância de $\alpha = 0.05$). Neste teste, todos os algoritmos-funções custo obtiveram como resultado $h = 1$ negando a hipótese nula H_0.

Seguidamente, o teste *Krustal-Wallis* [38] é aplicado em todos os conjuntos, com o objetivo de determinar se há diferencias estatisticamente significativa entre dois o mais conjuntos. Neste caso, a hipótese nula H_0 considera todos os conjuntos provenientes de distribuições com de dados que advêm da mesma distribuição de probabilidade e a e a hipótese alternativa H_0 indica que pelo menos um conjunto tem uma distribuição de probabilidade diferente das outras. Se o $p - valor \approx 0$, então H_0 é rejeitada com um nível de significância de $\alpha = 0.05$. O resultado deste teste foi $p - valor = 4.1102e - 10$.

Por fim, o teste *Wilcoxon ranksum* é usado para verificar que o dois conjuntos de valores do algoritmo-função provem de probabilidade diferentes. Neste teste, a hipótese nula H_0 indica que os dois conjuntos de dados decorrem de distribuições diferente, enquanto que, a hipótese alternativa H_1 aponta que ambos conjuntos tem a mesma distribuição de probabilidade. Se o $p - valor \approx 0$ ou $h = 1$ então H_0 é rejeitada com um nível de significância de $\alpha = 0.05$. Os resultados dos testes realizados comparando o algoritmo DE usando a função custo da trajetória dos dedos com os outros conjuntos estão listados na seguinte tabela.

Tabela 3.6: Teste *Wilcoxon ranksum* que compara o algoritmo DE usando função custo da trajetoria dos dedos com os otros.

Algoritmo-Função Custo	$p-valor$	h
GA-Traj	8.74e-09	1
PSO-Traj	0.19	0
GA-Ang	7.15e-12	1
PSO-Ang	3.24e-07	1
DE-Ang	2.06e-03	1

Fonte: o autor.

Pode-se observar que a hipótese nula do algoritmo PSO com a função custo da trajetória não é rejeitada, em outras palavras, os dois resultados não são estatisticamente diferentes. Sendo assim, os dois algoritmos resolvem o problema com a mesma efetividade. Por outro lado, mesclar os melhores resultados desses algoritmos resulta numa abordagem interessante, isto é realizado calculando a média das variáveis de decisão resolutas por esses algoritmos. O resultado desta abordagem pode-se observar na Fig. 3.12, obtendo uma proximidade maior à trajetória alvo do que as trajetórias originais, concluindo na sua escolha como mecanismo para incluir no projeto do dedo.

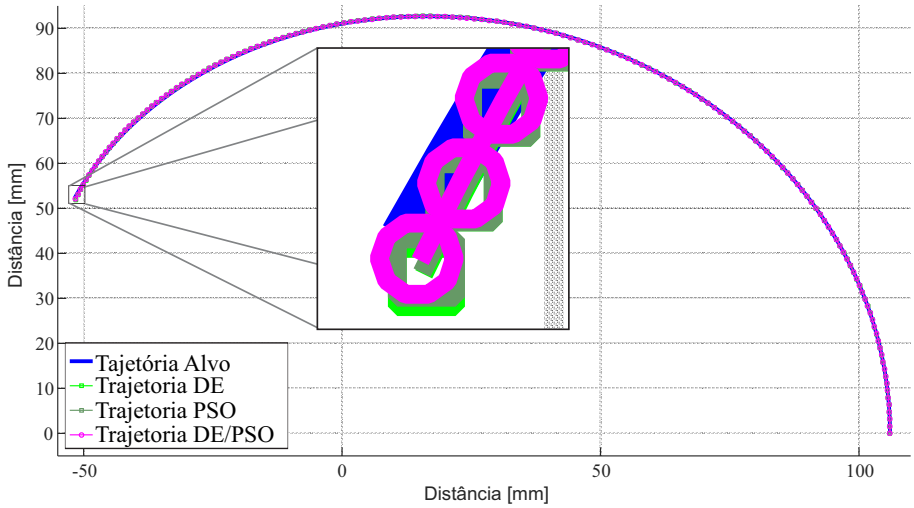


Figura 3.12: Melhores trajetórias otimizadas, o resultado escolhido para ser implementado é aquele computado das variáveis de decisão do PSO e o DE. Fonte: o autor.

Finalmente, o mecanismo otimizado é adicionado na estrutura cinemática proposta na seção anterior. A próxima seção integra os resultados obtidos nesta seção com a anterior para projetar o sistema de mão robótica completo.

3.3 Projeto e Fabricação do Protótipo de Mão Robótica

A seção atual descreve o projeto mecânico e eletrônico da mão robótica proposta neste trabalho. A mão robótica pretende ser uma abordagem de baixo custo de implementação permitindo diversos métodos de manufatura e prototipagem rápida sejam utilizados na fabricação da mão robótica. Portanto, escolheu-se o uso de matérias como primas como plástico e alumínio para o desenvolvimento do protótipo.

Inicialmente, o mecanismo otimizado na seção anterior é antropomorfizado, isto, com o intuito de atingir um dos objetivos do projeto, o qual é mimetizar uma mão humana real. A seguinte subseção ilustra o resultado do projeto dos dedos usando o mecanismo otimizado.

3.3.1 Dedos Subatuados

Cada dedo está composto pelas seguintes partes: a) o atuador, b) o sensor de posição, c) as falanges dos dedos, f) os elos de transmissão do mecanismo e g) o mecanismo de parafuso sem fim que transmite o movimento do motor.

O atuador utilizado nesta abordagem são micromotores de CC de alta potência. Estes motores atingem velocidades de $200RPM$ e um torque máximo de $0.28Nm$. Os motores vai atuar diretamente as juntas MCP_fede cada dedo, as quais conjuntamente têm um sensor de posição. O sensor de posição é um potenciômetro, o qual, também é utilizado para estimar a velocidade do dedo.

Os próximos componentes são os elos que constituem o mecanismo antropomorfizado de flexão/extensão. Na Fig. 3.13(a) apresenta-se os elos que representam as falanges dos dedos. Estas peças são intencionadas para ser fabricadas com impressora 3D em plástico ABS. Na Fig. 3.13.b estão ilustrados os elos de transmissão do mecanismo, sendo que estas peças são intencionadas para serem usinada em alumínio de 2mm.

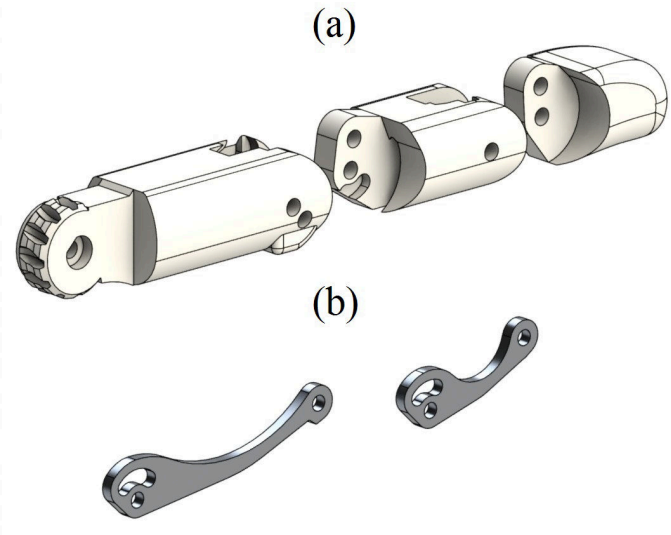


Figura 3.13: Mecanismo do dedo desmontado. a) Elos das falanges. b) Elos de transmissão do mecanismo. Fonte: o autor.

Finalmente o parafuso sem fim que transmite o movimento do motor à junta MCP_fe pode-se observar na Fig. 3.14. Nesta mesma figura, também se ilustra a montagem do mecanismo antropomorfizado. A transmissão de parafuso sem fim-coroa permite a transmissão do movimento axial a 90 graus, facilitando a implementação da atuação, além de reduzir o tamanho do robô. Por outro lado, este mecanismo acrescenta a redução do motor, o que permitirá gerar mais torque para agarres de potência, no entanto, a sua natureza auto-travante dificulta o controle dinâmico do robô.

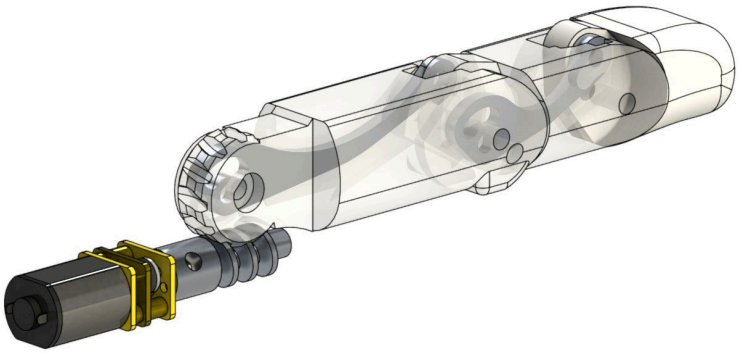


Figura 3.14: Montagem do dedo robótico. Fonte: o autor.

Assim, acaba a descrição do projeto dos dedos a serem incluídos no projeto do sistema de mão robótica completo, o qual é apresentado na seguinte subseção.

3.3.2 Projeto CAD da Mão Robótica

O projeto CAD da mão é feito usando como base o esquemático da cinemática apresentado na Seção 3.1 e o projeto dos dedos obtidos na subseção anterior. É mister ter em consideração que a configuração cinemática da mão proposta neste trabalho (vide Fig. 3.2), detém 3 tipos de dedos:

- O polegar, o qual tem 2 GDLs (adução/abdução e flexão extensão) e 3 GDLs (juntas θ_{t,TMC_aa}, θ_{t,MCP_fe} e θ_{t,DIP_fe});
- O indicador, que possui 2 GDLs (adução/abdução e flexão extensão) e 4 GDLs (juntas θ_{i,MCP_aa}, θ_{i,MCP_fe}, θ_{i,PIP_fe} e θ_{i,DIP_fe}) e
- Os dedos restantes (médio, anelar e mínimo), os quais tem 1 GDA (flexão extensão) e 3 GDLs (θ_{MCP_fe}, θ_{PIP_fe} e θ_{DIP_fe})

Visto que o último tipo de dedo inclui apenas os movimentos de flexão/extensão, pode-se implementar usando diretamente o modelo de dedo sub-atuado exibido na subseção anterior. Assim, os outros dois tipos de dedos são implementados adicionando uma junta a mais.

Adicionar a junta extra de modo que intercepte a junta θ_{i,MCP_fe} implica uma dificuldade alta na implementação. Nesta abordagem decidiu-se adicionar um espaço entre as duas juntas para facilitar a

implementação do movimento de adução/abdução. A Fig. 3.15 apresenta o dedo indicador, destacando-se o posicionamento dos eixos de movimento e o comprimento x é o espaço entre as juntas.

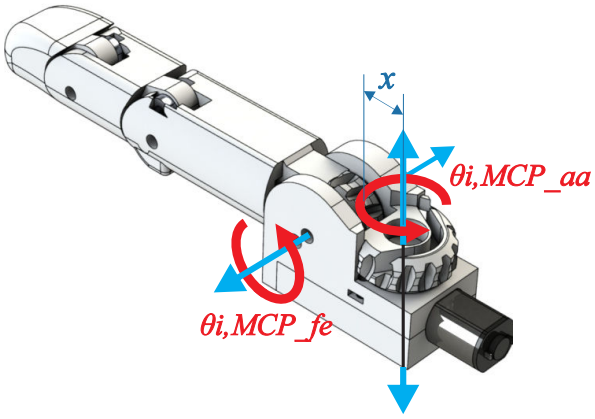


Figura 3.15: Dedo Indicador com eixos de flexão/extensão e adução/abdução. Fonte: o autor.

Uma abordagem similar é proposta para a implementação do polegar. Dada a natureza da oponibilidade do polegar, o eixo de adução/abdução não está defasado do flexão/extensão por $90°$; nesse trabalho, um ângulo de $55°$ foi usado (vide Fig. 3.16).

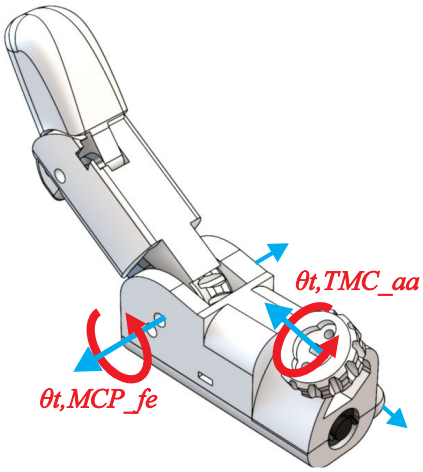


Figura 3.16: Dedo polegar com eixos de flexão/extensão e adução/abdução. Fonte: o autor.

Finalmente, a parte da mão robótica que integra todos os componentes é a palma. Por facilidade de projeto e de manufatura, a geometria da palma da mão robótica nesse trabalho é retangular, conservando as dimensões de uma mão real. A Fig. 3.17 elucida o resultado final do projeto da mão robótica, pode-se observar o posicionamento dos atuadores. Cabe salientar que o posicionamento dos atuadores dos movimentos de adução/abdução é muito importante, especialmente aquele do polegar devido a que a orientação da junta de adução/abdução deve-se escolher considerando a oponibilidade do polegar.

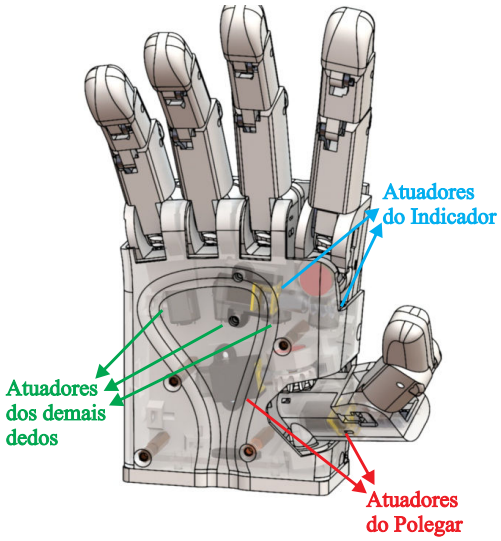


Figura 3.17: Projeto CAD da palma integrando todos os dedos da mão robótica. Fonte: o autor.

A oponibilidade do polegar da mão robótica deste trabalho pode-se quantificar usando uma adaptação do bem conhecido teste clinico de Kapandji [39]. O método avalia a capacidade da ponta do dedo polegar em atingir o contacto com uma parte especifica da mão. Totalizando, são 10 posições em ordem de dificuldade, onde, o posicionamento mais fácil é o primeiro (a falange proximal do dedo índice), e, o posicionamento mais difícil é o último (a distensão palmar distal). O posicionamento e orientação dos dedos foi feita empiricamente visando atingir todas as posições do teste de Kapandji. A Fig. 3.18 apresenta dois testes representativos: o teste número 6 e o 10, de modo que, a mão robótica desse trabalho consegue o posicionamento mais difícil.

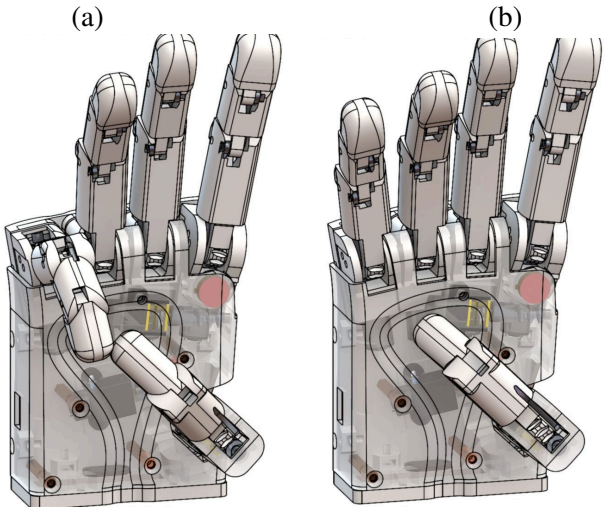


Figura 3.18: Teste de Kapandji. (a) Teste 6 (tocar a ponta do dedo mínimo). (b) Teste 10 (tocar a distensão palmar distal). Fonte: o autor.

A placa eletrônica que inclui a adequação dos sinais dos sensores de posição e a amplificação de potência para fornecer os motores deve caber no espaço disponibilizado na placa. A mesma, está detalhada na seguinte subseção, onde, se descrevem as componentes eletrônicas da mão robótica.

3.3.3 Projeto Eletrônico

O protótipo de mão robótica inclui os atuadores e o circuito de adaptação de sinais embarcado na palma. Isto facilita o seu uso em diversas aplicações como a sua montagem em manipuladores robóticos como órgão terminal. De tal forma que, a placa eletrônica deve ter um tamanho reduzido para poder ser utilizada dentro da palma.

Como foi dito anteriormente, a placa eletrônica deve conter os seguintes componentes: a) os drivers dos motores (estes fornecem a tensão elétrica ao motor através de PWM), b) os transdutores de corrente (um para cada motor, menos para o motor da junta θ_{i,MCP_aa}). c) os reguladores dos sinais a $1V1$ e $5V$ para alimentar os sensores de posição e transdutores de corrente. Esta escolha se justifica devido a que o conversor ADC que planeja ser usado usa níveis de tensão de até $1V1$, os transdutores de corrente funcionam a uma tensão de $5V$. Os motores são alimentados a $8V$, portanto, a placa é fornecida com $8V$, d) os "*sockets*" dos motores e sensores de posição, e) o "*breakout*" dos sinais de entrada: PWMs dos motores (dois por cada um) e finalmente, f) o "*breakout*" dos sinais de saída: os sinais analógicos dos sensores de posição dos dedos e os sinais analógicos dos transdutores de corrente dos motores.

Por fim, os CIs (Circuitos Integrados) usados no protótipo de mão robótica são listados a continuação.

Tabela 3.7: CIs da placa eletrônica da mão robótica

Função	Referencia	Requerimentos de Projeto	Caraterísticas do Componente
Drivers	Pololu DRV8833	Corrente continua de $1.6A$ Tensão dos motores de $8V$ Tensão lógica de $3.3V$ e $5V$	Corrente continua de $2.4A$ Tensão dos motores de $2.7V - 10.8V$ Tensão lógica de $3.3V$ e $5V$ Limitador de Corrente
Transdutores de corrente	ACS712 ELCTR-05B-T	Corrente Bidirecional Corrente máxima de $\pm 1.6V$ Frequência de operação $> 100Hz$	Corrente Bidirecional Corrente sensível de $\pm 5A$ Frequência de operação $\approx 80kHz$
Regulador de tensão step-down	LM317	Niveis de regulação: $5V$ e $1.1V$ Corrente máxima $100mA$	Rango de tensão de saída de $1.25V - 37V$ Corrente de saída de até $1.5A$

Fonte: o autor.

O esquemático da placa é ilustrado na Fig. 3.19. Além dos CIs, a placa também ocupa espaços nos breakout e os componentes passivos (resistências e capacitores). Por esse motivo, decidiu-se que a placa seria produzida com componentes de montagem superficial (SMD).

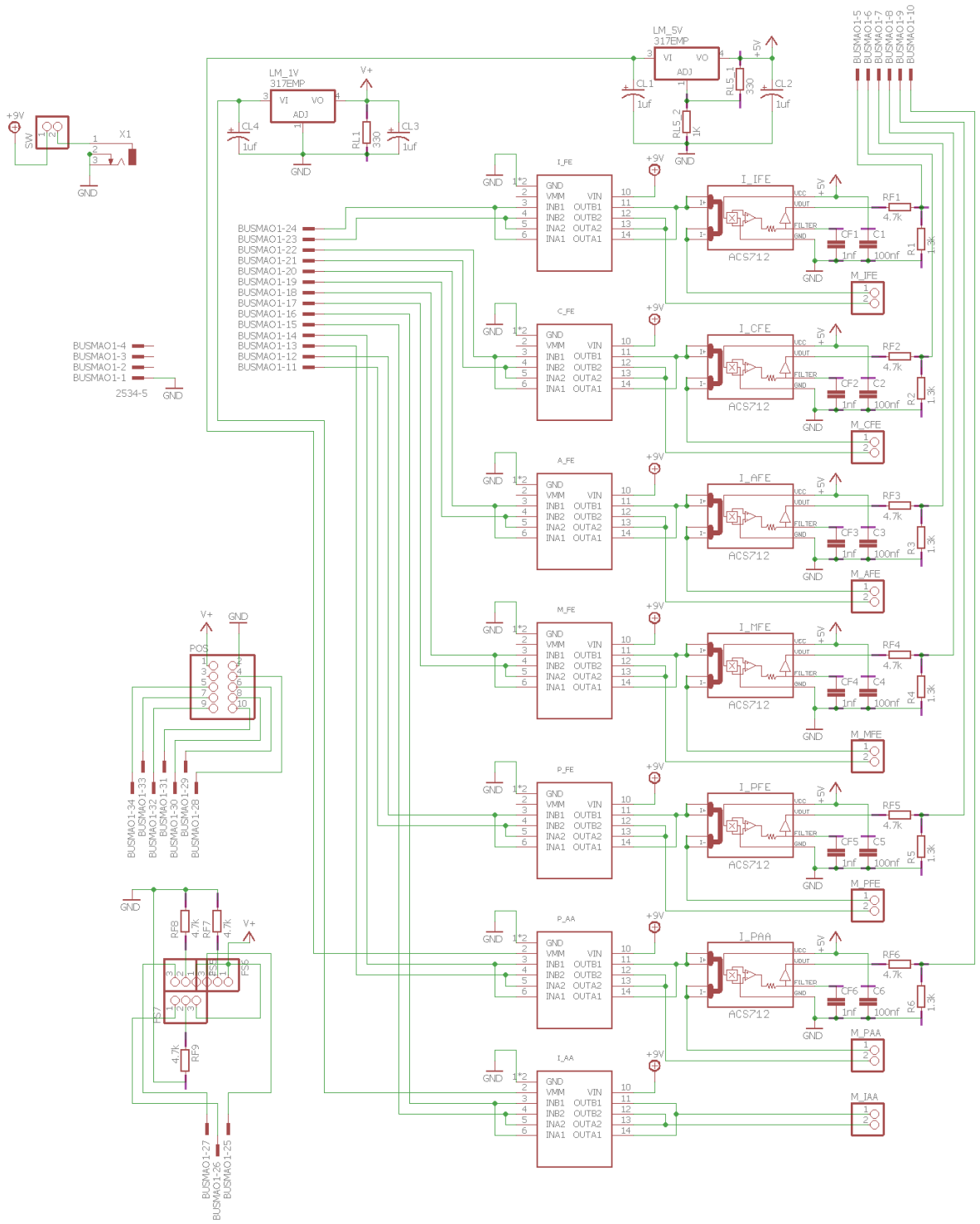


Figura 3.19: Esquemático da placa eletrônica da mão robótica. Fonte: o autor.

O layout dos componentes eletrônicos é utilizado para projetar a placa com o posicionamento físico dos mesmos. O resultado da placa projetada é o apresentado na Fig. 3.20. As dimensões da placa são visadas para encaixar na palma da mão apresentada na subseção anterior.

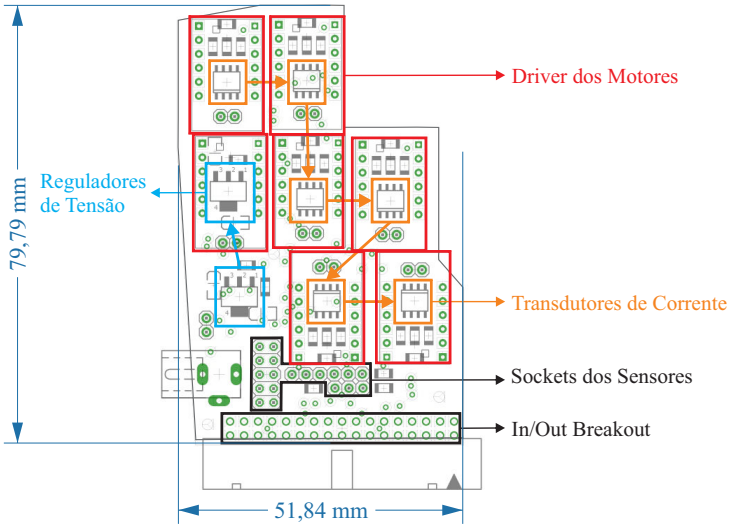


Figura 3.20: Projeto da placa eletrônica da mão robótica. A geometria da placa é para que permite o encaixe na palma da mão apresentada na seção anterior. Escala 1:1. Fonte: o autor.

Finalmente, a última parte desta seção está dedicada à manufatura e montagem dos componentes do sistema de mão robótica apresentado até agora.

3.3.4 Fabricação do Protótipo de Mão Robótica

Como já foi indicado, a manufatura do protótipo da mão robótica é realizado usando vários métodos de fabricação e materiais. Em primeiro lugar, estão os componentes fabricados em aluminio: os parafusos sem fim e os elos de transmissão do mecanismo de flexão/extensão. Os parafusos sem fim foram usinados em torno e os elos foram usinados em placas de 2 mm de alumínio com processo de corte com jato de água. Algumas das peças usinadas podem-se observar na Fig. 3.21.

(a) (b)

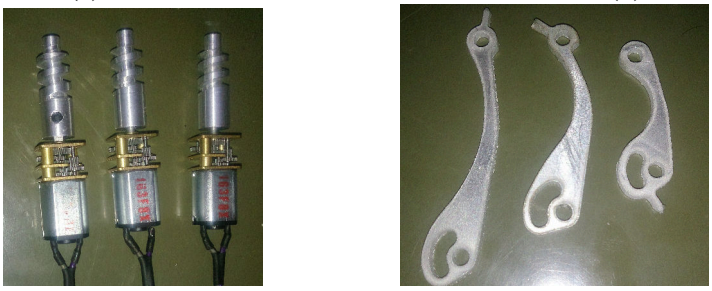


Figura 3.21: Peças fabricadas em alumínio. (a) Parafusos sem fim montados em micromotores. (b) Elos de transmissão do mecanismo. Fonte: o autor.

A placa eletrônica foi fabricada no Laboratório de Prototipagem de Eletrônica do IFB - Campus Taguatinga. O laboratório conta com equipamentos de prototipagem de placas de circuitos SMD multicamada,

tal como, a fresa CNC e Pick&Place semiautomático que foram usados para esse projeto. O resultado da placa é ilustrado na Fig. 3.22.

(a) (b)

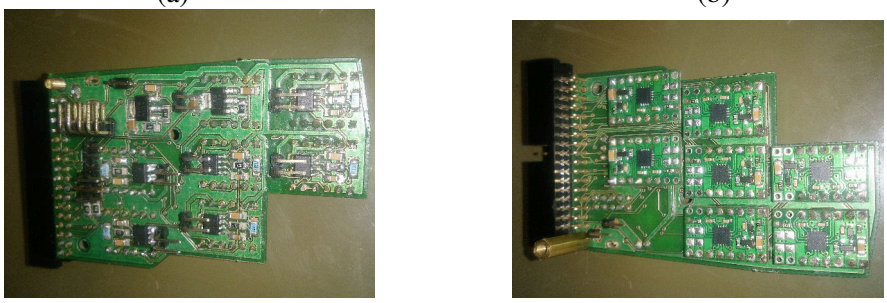


Figura 3.22: Placa eletrônica da mão robótica. (a) Vista do topo. (b) Vista de baixo. Fonte: o autor.

O restante dos componentes da mão robótica foram fabricados usando impressoras 3D disponibilizadas no Laboratório de Eletrônica do LEIA-GRACO na UnB. As impresoras utilizadas, são máquinas RepRap Prusa i3 com resolução de camada de até $100\mu m$, as peças foram impressas em material plastico ABS. O ensamble das peças é apresentado na Fig. 3.23.

(a) (b) (c)

(d) (e) (f)

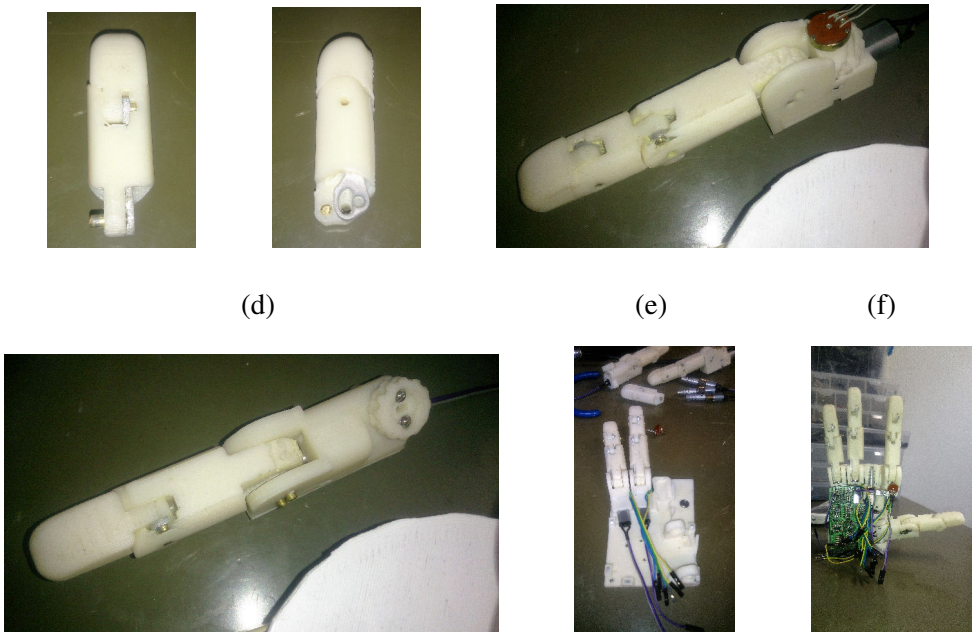


Figura 3.23: Montagem Mão Robótica. (a) e (b) Falanges proximal e distal. (c) Dedo indicador. (d) Dedo polegar. (e) Palma parcialmente montada com o dedo mínimo e anelar. (f) Palma parcialmente montada com a placa eletrônica. Fonte: o autor.

Finalmente, a montagem do protótipo pode-ser observado na Fig. 3.24. O protótipo de mão robótica tem um peso de $413.13Kg$ e as suas dimensões são: $210 \times 84 \times 38\ mm$.

Figura 3.24: Mão robótica Montada. Fonte: o autor.

3.4 Conclusões de Capitulo

Neste capitulo foi apresentado o projeto, otimização e manufatura de uma mão robótica biomimética. O maior diferencial do projeto apresentado neste trabalho é o uso de mecanismos rígidos para sub-atuar os dedos, a otimização do mecanismo usando algoritmos bioinspirados e o a fabricação de um projeto que de baixo custo que é próximo às dimensões de uma mão humana.

O protótipo de mão robótica nesse trabalho é um modelo simplificado dos 7 GDLs mais importantes na hora de realizar agarres. Através da análise cinemática, foi demonstrado a capacidade da área de trabalho do projeto, concluindo-se que é possível realizar várias posições na manipulação de objetos. A oponibilidade do dedo polegar, a qual tem um papel importante para a avaliação da destreza da mão, foi validada usando uma adaptação do teste clinico de Kapandji, conseguindo todas as posições que demanda o mesmo.

Desde uma perspectiva mecânica, a simplificação dos movimentos de flexão/extensão no projeto de mão robótica, permitiram a implementação de um projeto compacto, onde a maioria dos componentes foram incluídos na palma do protótipo. Isto facilita a adaptação do protótipo para ser usado com outros manipuladores.

Essa simplificação foi possível pelo uso de mecanismos acoplados usados em todos os dedos da mão robótica. Esses mecanismos passaram por um processo de otimização que visou que o seu comportamento mimetize aquele de uma mão humana. Os algoritmos usados para otimizar o mecanismo foram GA, PSO e DE. O algoritmo com o melhor resultado foi o DE com $MSE = 0.00266$. No entanto, a análise estatística dos resultados dos algoritmos revelaram que, estatisticamente, os algoritmos PSO e DE resolveram o problema de maneira similar. Consequentemente, decidiu-se usar as variáveis de decisão dos melhores resultados desses algoritmos mediante o cálculo da média das mesmas. O que resultou em um erro de MSE menor.

Finalmente, neste capitulo, a manufatura e montagem do protótipo de mão robótica é descrita e ilustrada.

Capítulo 4

Projeto da Estratégia de Controle

Realizar tarefas de agarres e manipulação é uma tarefa muito importante na área da robótica, isto reflete-se na grande quantidade de órgãos terminais e abordagens usados como ferramentas nesta ação. No entanto, essa variedade de abordagens para realizar tarefas de agarre concorre, do mesmo modo, da complexidade da mesma, porque, os objetos que são alvo de agarres nem sempre tem geometrias simples, ou tem superfícies que dificultam essa tarefa. As mãos robóticas biomiméticas são uma solução universal possível a essa problemática, dado a destreza e habilidade que esses sistemas poderiam atingir potencialmente. Por este motivo, o interesse da comunidade cientifica nesta área é grande, o que pode ser demonstrado pelo fato de que todo ano uma abordagem diferente é contribuída na literatura cientifica [16, 17, 40, 41, 42, 43, 44].

Em mãos robóticas, a habilidade de realizar agarres com destreza é uma característica fundamental. Basicamente, os métodos de controle para agarres podem-se agrupar em dois categorias principais: a) agarres com planejamento clássico de movimento com detecção de colisões e b) síntese do agarre através da construção direta de contatos baseados nas propriedades do movimento de fechamento. Bohg *et al.* [45] realiza uma revisão abrangente sobre a primeira categoria, além disso, destaca que essas abordagens classificam-se em três categorias, de acordo ao conhecimento disponível sobre os objetos alvo para agarre:

1. O objeto é bem conhecido. Nesse caso, assume-se que o agarres do objeto alvo já foi previamente realizado. Nesta abordagem o robô tem acesso a uma base de dados com características do objeto (geometria, superfície, etc.).
2. O objeto é medianamente conhecido. Neste caso, ao invés de conhecer perfeitamente o objeto, atribui-se ao objeto similitudes com um grupo de características geométricas predefinidas. O desafio nesta abordagem é definir essas similitudes.
3. O objeto é totalmente desconhecido. Nesta abordagem são desconhecidas qualquer tipo de característica do objeto.

Nesse trabalho a abordagem explorada é a última. Nesta categoria, as características do objeto alvo de agarre (geometria, superfície, etc...) são maiormente desconhecidas, daí que, o planejador do agarre calcula o esquema do agarre baseado na informação que é deduzida usando os sistemas sensoriais da mão. Com esse intuito, uma grandeza comum para ser medida ou estimada é a força/

e que a mão aplica ao fechamento, em outras palavras, a força/torque que é aplicada no objeto através da mão. Vários trabalhos correlatos usam força/torque para realizar agarres [16, 41, 46, 47].

A estimação de torque para controlar agarres implica a realização do controle dinâmico para realizar os agarres dos objetos. Como foi mencionado anteriormente, uma destas abordagens é o controle de impedâncias, o qual visa controlar a dinâmica com que o robô interatua com o ambiente [48, 49]. Nesta abordagem utiliza-se realimentação de torque para conseguir um comportamento desacoplado linear. O controle de impedâncias pode ser definido, basicamente, como um sistema dinâmico de segunda ordem com parâmetros ajustáveis (vide Seção 2.2).

Estes parâmetros podem ser sintonizados virtualmente para qualquer valor, no entanto, para obter um sistema estável, esses coeficientes dependem fortemente do sistema alvo de ser controlado.

Sintonizar esses coeficientes, pode ser uma tarefa difícil pois envolve cálculos complicados da dinâmica do robô, contudo, algoritmos de otimização bioinspirados têm sido explorados para a solução de sintonização de controlador, no entanto, a aplicação especifica de sintonizar um controlador de impedância ainda não foi abordado por outro autor na literatura. Esta, é um dos aportes deste trabalho à comunidade cientifica. Nessa abordagem o algoritmo de Particle Swarm Optimization (PSO) é utilizado com o intuito de otimizar os valores desses coeficientes.

A finalidade deste capitulo é propor e validar um esquema de controle dinâmico para o dedo apresentado da Seção 3.3.1. O esquema de controle proposto implementa um controlador de impedâncias para controlar a dinâmica do dedo. O restante deste capítulo descreve os seguintes tópicos: a) a proposta e validação do esquema de controle através de simulação numérica e, b) a otimização dos coeficientes usando PSO que resulta num sistema com um comportamento dinâmico estável ao fechar do dedo robótico.

4.1 Controlador de Impedância

Como já foi dito, a dinâmica de um robô é controlada com o modelo de controle de impedâncias da Eq. 2.1, para esse trabalho o método de integração discreta [50] é utilizado para implementar o controle de impedância, este método representa o esquema de controle através de 3 passos:

1. Discretizar a Eq. 2.1 do sistema representando-o da seguinte maneira:

$$\ddot{x}(k) = M^{-1}(\Delta T(k) - K\Delta x(k) - B\Delta\dot{x}(k)) \tag{4.1}$$

   onde a força F na Eq. 2.1 é substituída pelo erro do torque observado do sistema no momento k $\Delta T(k)$, $\Delta x(k)$ é o erro da movimento e $\Delta\dot{x}$ é o erro da derivada do movimento.

2. O resultado da equação anterior $\ddot{x}$ o qual representa a aceleração da junta é integrada para obter $\dot{x}$

$$\dot{x} = \int_{k-1}^{k} \ddot{x}dx \tag{4.2}$$

3. Repetir o processo para obter x que é a posição desejada do sistema

$$x = \int_{k-1}^{k} \dot{x} dx \tag{4.3}$$

x nesta abordagem é a posição ou deslocamento desejado do robô controlado, nesse caso, a junta atuada do dedo robótico.

A representação esquemática do método de integração discreta do controle de impedância é apresentado na Fig. 4.1.

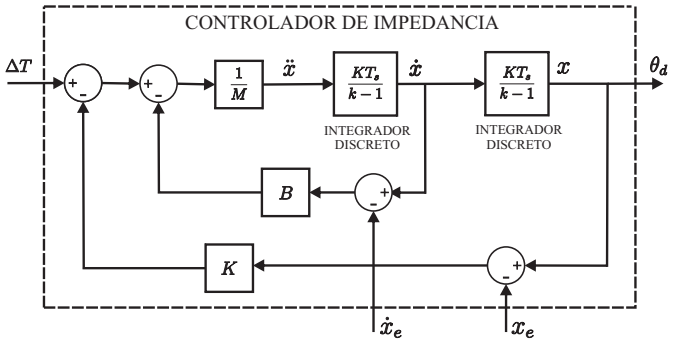


Figura 4.1: Esquema de controle proposto. T_s é o período de atualização do controlador e x_e e $\dot{x}_e$ são o movimento medido e a sua derivada respetivamente. Fonte: o autor.

O esquema apresentado na figura anterior é implementado no sistema completo de controle do dedo robótico, ilustrado na Fig. 4.2. O modelo de controle proposto, visa detectar os objetos que interferem no movimento de fechamento do dedo robótico. O bloco controlador de impedância tem como dados de entrada o erro ($\Delta T = T_d - T_e$) e os valores estimados de posição e velocidade (x e $\dot{x}$). Como dado de saída está posição desejada da junta (θ_d). O controlador PID destina-se para controlar a posição da junta calculando a tensão elétrica do motor. Essa tensão é transferida para o sistema do dedo robótico que conta com sensores de posição (θ) e corrente (i) (vide Seção 3.3.1).

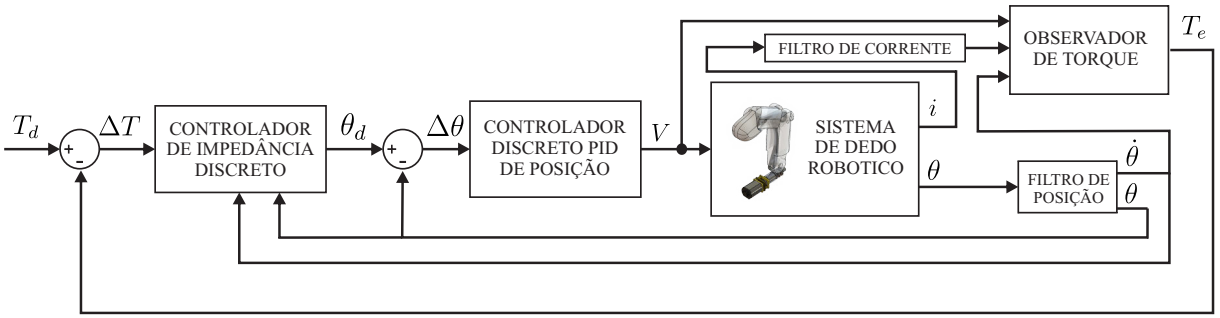


Figura 4.2: Esquema de controle proposto. Fonte: o autor.

O bloco "Observador de Torque", estima o torque exercido pelo motor. A equação de torque deriva-se usando a corrente que passa pelo motor, a tensão fornecida e a velocidade estimada do eixo, tal como explicado a seguir:

Temos que $P_{in} = iv$, $P_{out} = T\omega$ onde P_{in} e P_{out} são as potencias de entrada e saída respetivamente, i é a corrente que passa pelo motor, v a tensão, ω a velocidade angular do eixo de saída e T é o torque gerado pelo eixo. Se a eficiência do motor vem dada pela proporção $E = P_{in}/P_{out}$ deriva-se a equação

$$T = \frac{ivE}{\omega} \tag{4.4}$$

Da equação anterior pode-se ver uma descontinuidade e é quando $\omega = 0$. A solução para esse problema é definir uma velocidade mínima para $\omega > 0$.

Adicionalmente, os blocos de filtragem dos sensores são revistados na Seção 5.1.1.1. O bloco que simula a ação do dedo robótico apresenta-se a seguir.

4.1.1 Simulador do Dedo Robótico

Com o intuito de observar o comportamento dinâmico do dedo robótico de forma segura, um simulador numérico foi desenvolvido neste trabalho. O simulador é desenvolvido em Matlab/Simulink-Simmechanics e pode ser visto na Fig. 4.3, o sistema do dedo é composto pelos seguintes blocos (vide Fig. 4.3(d)): (a) o driver amplificador de potência para o atuador, (b) o atuador com o sensor de corrente que mede a magnitude da corrente que passa pelos terminais do motor em um determinado instante, (c) o parafuso sem fim acoplado ao eixo do motor, (d) o mecanismo de flexão extensão do dedo, o qual inclui o sensor de posição na junta θ_{MCP_fe} e finalmente (e) um bloco que simula a colisão do dedo com um objeto para imitar um agarre. O simulador conta com um visualizador gráfico 3D com a posição e estado do mecanismo e o objeto alvo de agarre.

Na figura anterior pode-se observar em 4.3(b)-(d) o movimento de flexão/extensão do dedo e na Fig. 4.3(e)-(f) ilustra-se o comportamento do sistema de dedo robótico em função do tempo. Pode-se observar a perturbação que provoca a colisão com o objeto (bola vermelha). No grafo de corrente pode-se ver como a perturbação aumenta a corrente do motor, o qual também freia o eixo, e a falta de movimento pode-se ver no grafo da posição após o segundo 1.

Os coeficientes K_P, K_I e K_D do bloco controlador PID de posição ilustrado na Fig. 4.2 são derivados usando a metodologia de Ziegler–Nichols, o qual é um método heurístico manual de sintonização dos coeficientes que cria um controlador agressivo com um ganho proporcional e um sobre-impulso alto [51]. Em contraste, os coeficientes do controlador de impedância são sintonizados usando algoritmos de otimização bioinspirada *PSO*, esse processo é apresentado na seguinte seção.

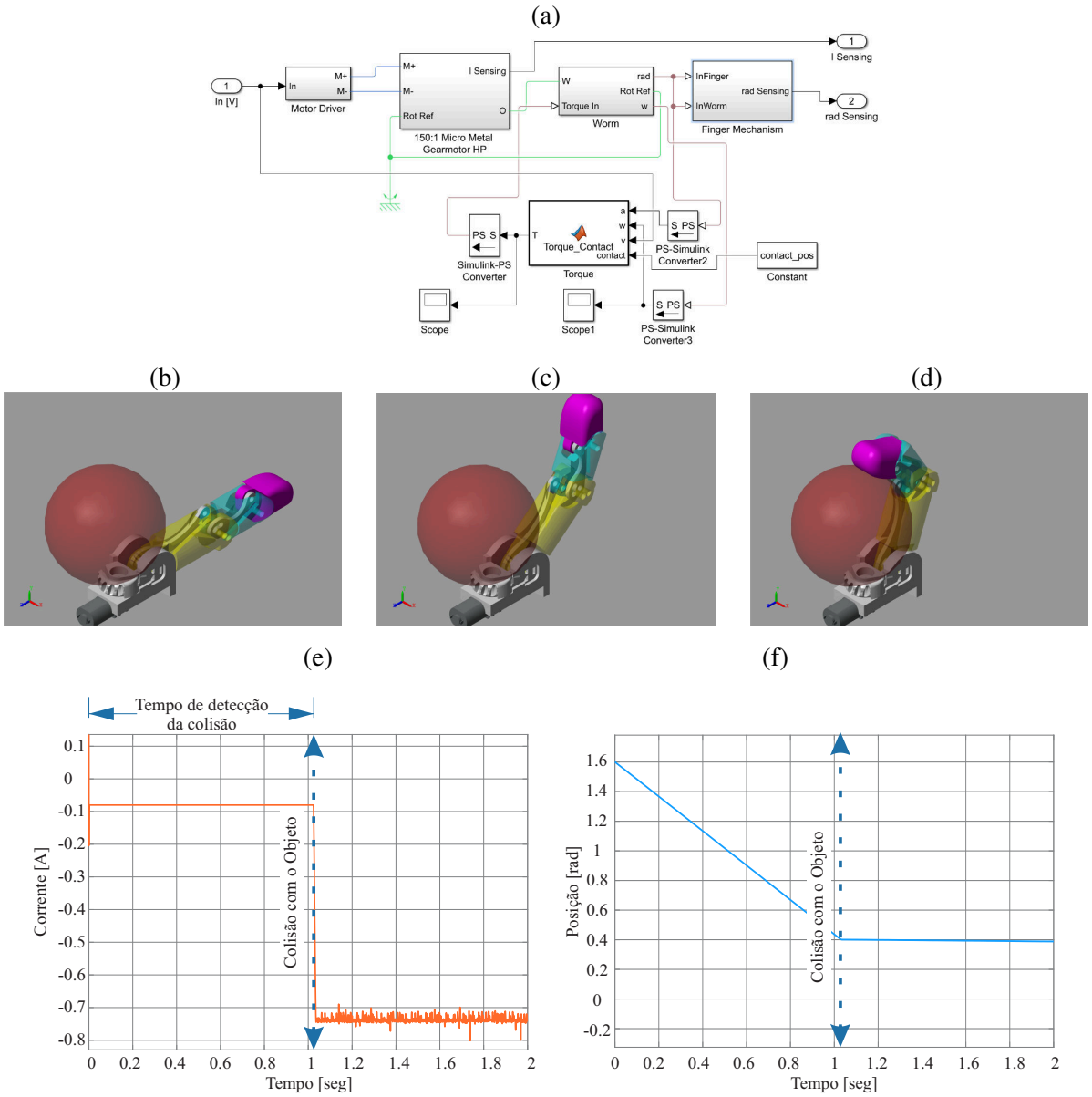


Figura 4.3: Simulador do dedo robótico modelado em Matlab/Simulink Simmechanics, (a) esquemático em Matlab/Simulink do simulador, (b)-(d) simulador executando o comportamento do dedo quando um degrau de $-6V$ é fornecido ao atuador e, (e)-(f) dados coletados dos sensores de corrente e posição respetivamente. Fonte: o autor.

4.2 Sintonização dos coeficientes do Controlador de Impedâncias

O algoritmo de otimização bioinspirado *PSO* (vide Seção 2.5.1) é um algoritmo de busca heurística que visa atingir um objetivo, com o menor esforço possível e imitando o comportamento que tem os enxames ao buscar alimento. O uso de algoritmos bioinspirados de otimização para sintonizar um controlador é uma aplicação comum, no entanto, este problema de otimização pode-se resolver com diferentes tipos de abordagens, especificamente, através de diferentes definições da função custo. A função custo geralmente

representa o principal problema para este tipo de aplicações, a seguinte subseção aborda como foi calculada a função custo e a determinação e limitações das variáveis de decisão.

4.2.1 Definição do Problema

O problema de otimização nesta seção é definir os valores dos coeficientes do controlador de impedância que determinam a correção correta do torque gerado. Da frase anterior, deriva-se que as variáveis de decisão são os coeficientes do controlador de impedância M, B e K, no entanto, esses valores são limitados pelo amortecimento desejado.

Para os experimentos de agarres no trabalho atual é desejado um comportamento sub-amortecido, para o qual é calculada a taxa é calculado a taxa de amortecimento ζ, que é uma medida que descreve como as oscilações em um sistema decaem após alguma perturbação. Essa taxa vem dada pela relação

$$\zeta = \frac{c}{c_c} \tag{4.5}$$

onde c é o coeficiente de amortecimento do sistema atual e c_c é o coeficiente de amortecimento quando o sistema está criticamente amortecido. Da equação do controlador de impedâncias dado na Eq. 2.1, deriva-se a taxa de amortecimento

$$\zeta = \frac{B}{2\sqrt{KM}} \tag{4.6}$$

O valor de ζ desejado para a aplicações onde se requer uma compensação entre o sobre-impulso máximo permitido e o tempo quando o mesmo ocorre (t_p). Uma taxa de amortecimento menor diminui t_p que é um fato positivo, no entanto, diminuir t_p aumenta o sobre-impulso. A taxa de amortecimento é de decisão subjetiva e, na maioria das aplicações está no intervalo $0.4 \leq \zeta \leq 0.7$ [52]. Logo, as variáveis de decisão são: K, B e ζ, onde M é calculada usando a Eq. 4.6.

A função custo do problema é calculada observando o torque gerado pelo sistema do dedo robótico, derivando várias caraterísticas da sua resposta ao degrau e, finalmente, ponderando esses valores. As caraterísticas quantificadas para a função custo são: a) a porcentagem de sobre-impulso. Esta tem uma penalidade quando não supera o valor do setpoint (sub-impulso), b) o tempo usado para ocorrer a colisão com o objeto (t_o, vide Fig. 4.3(e)-(f)), c) o tempo de estabilização do controlador após a colisão com o objeto (t_e) e d) o MSE da resposta do controlador comparando com o setpoint (vide Eq. 3.8). O erro total é calculado ponderando cada variável da seguinte maneira

$$f\_custo = 0.05(sobre - impulso) + 0.15(t_o) + 0.15(t_e) + 0.65(MSE) \tag{4.7}$$

onde os pesos de cada variável são determinados de maneira empírica.

Resumindo, o problema de otimização nessa seção é mono-objetivo e de dimensionalidade 3. Na seguinte subseção mostra-se os resultados do controlador de posição PID do dedo robótico e do procedimento de otimização do controlador de impedância usando os critérios apresentados anteriormente.

4.2.2 Resultados

O sistema de simulação do controlador e o sistema de dedo robótico implementado em Matlab/Simulink é ilustrado na Fig. 4.4.

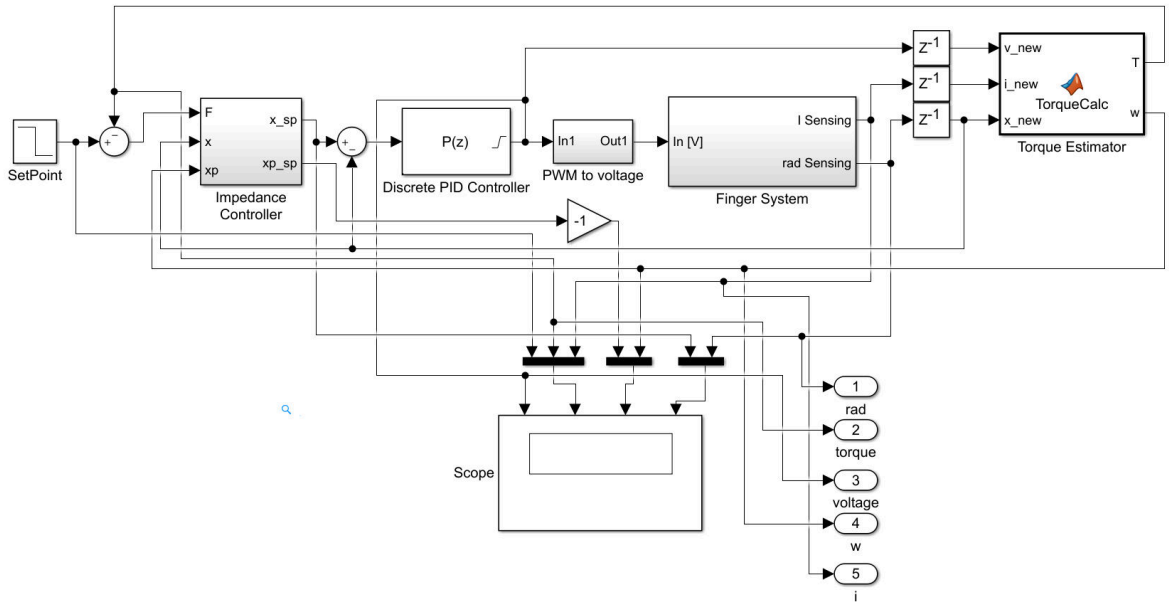


Figura 4.4: Esquema de controle desenvolvido em Matlab/Simulink. Fonte: o autor.

Na utilização do algoritmo de PSO para a otimização do controlador de impedância foi utilizada uma população do enxame de 12 partículas e 500 iterações. Devido ao pouco conhecimento sobre o sistema, a posição inicial das partículas é aleatória. A faixa de possíveis valores de decisão para ζ foi descrito na subseção anterior. Contudo, para as outras variáveis, pouca informação está disponível, no entanto, com o intuito de que o efeito do controlador seja maior, da Eq. 2.1 infere-se que o valor de M deve ser pequeno. Concluindo assim que $K > B$ da Eq. 4.6. A Tabela 4.1 descreve o condicionamento do algoritmo PSO usados nesta abordagem, nela, estão descrita os limites de busca do algoritmo de otimização.

Dada a natureza das variáveis de decisão e do sistema, existem vários mínimos locais, com o intuito de minimizar o efeito desta caraterística no algoritmo PSO, implementa-se uma modificação de diversificação no algoritmo clássico. A modificação usada é o método de inversão da posição, onde, si o mínimo do indivíduo passa do número de tentativas para melhorar o seu resultado, a posição da partícula é reiniciada aleatoriamente.

Tabela 4.1: Condições experimentais do algoritmo PSO para a otimização dos coeficientes do controlador de impedância. *$[K,\ B,\ \zeta]$

Parâmetro	Valor
Número de Partículas	12
Dimensionalidade	3
Limite inferior de busca*	$[10^{-16}, 10^{-16}, 0.4]$
Limite superior de busca*	$[30, 10, 0.7]$
Iteração Limite	100
Coeficiente Cognitivo c_1	2.05
Coeficiente Social c_2	2.05
ω (decrescimento linear)	$[1.0, 0.001]$

Fonte: o Autor

O erro mínimo atingido pelo algoritmo PSO após as 500 iterações é 0.0725, com as variáveis de decisão $K = 5.0774$, $B = 0.3450$, $M = 0.0052$ e $\zeta = 0.7000$. A convergência do erro mínimo de cada iteração está ilustrada na Fig. 4.5.

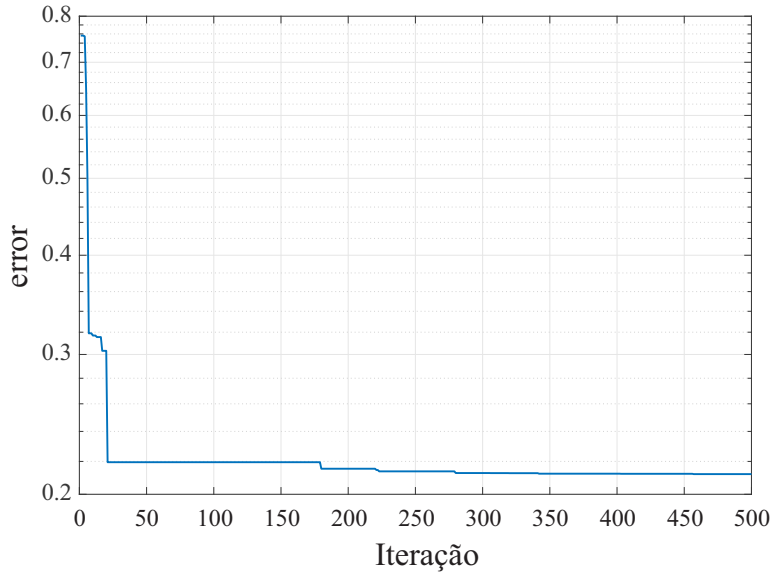


Figura 4.5: Convergência do MSE do processo de otimização do controlador de impedância. Fonte: o autor.

Finalmente, o comportamento do dedo está descrito na Fig. 4.6, nessa figura, estão inclusos: a) a correção do torque realizada pelo controlador de impedância, b) a correção da posição do sensor do dedo e a posição alvo calculada pelo controlador de impedância, c) a velocidade do dedo em função do tempo e a velocidade calculada pelo controlador de impedância e d) a tensão elétrica alimentada ao atuador.

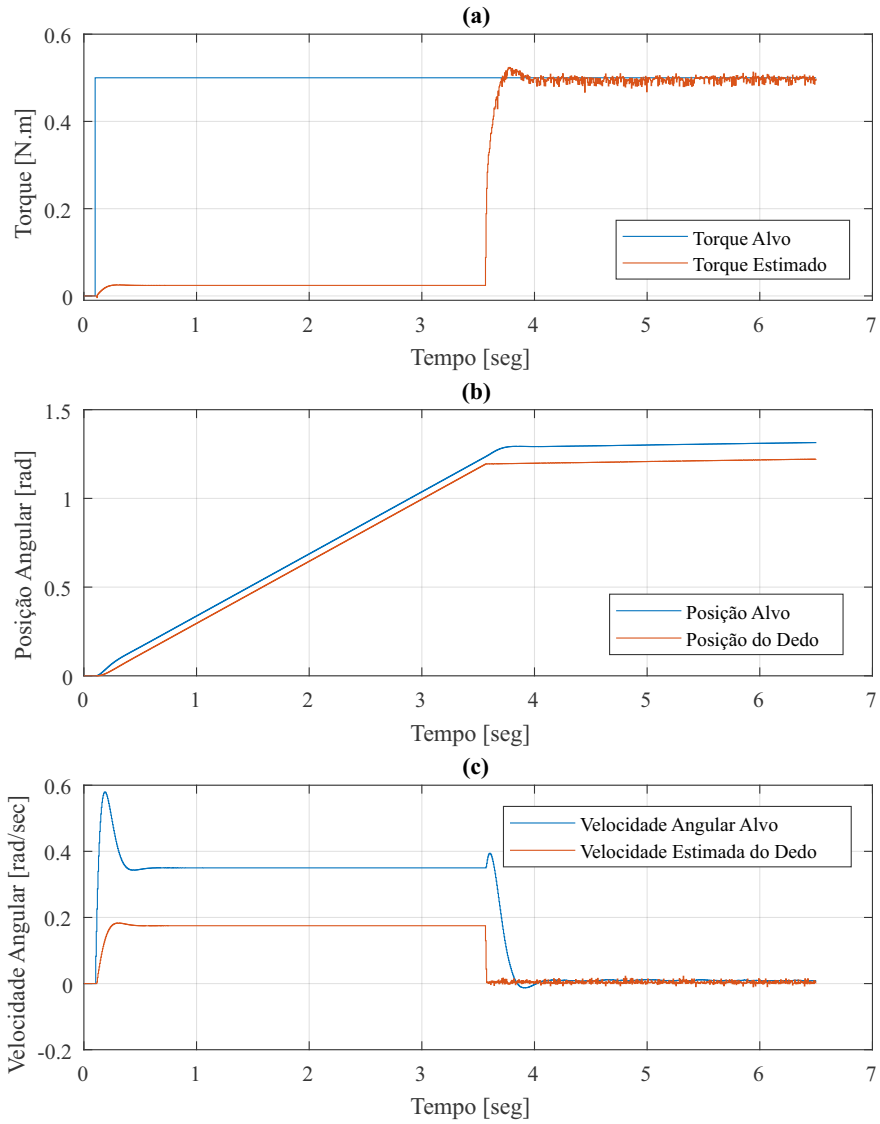


Figura 4.6: Comportamento do controlador de impedância otimizado, (a) Torque vs Tempo, (b) Posição vs Tempo, (a) Velocidade vs Tempo. Fonte: o autor.

Da figura anterior deriva-se várias observações importante como o tempo de colisão com o objeto de $3,565s$ e o tempo de estabelecimento do controlador ao detetar a colisão, o qual foi após $335ms$. Após o período de estabelecimento, o erro em estado estável calculado com MSE (vide Eq. 3.8) teve um valor de 0.0120.

4.3 Conclusões

O controlador de impedância que visa controlar a dinâmica da mão robótica deste trabalho foi projetado e sintonizado neste capítulo. O controlador de impedância é simulado em conjunto com o sistema de dedo

robótico com a finalidade de realizar vários experimentos sem comprometer a mão robótica desenvolvida no capítulo anterior. O anterior, possibilitou o uso do algoritmo de otimização *PSO* para sintonizar o controlador de impedância, os resultados da otimização foram aceitáveis obtendo um erro de apenas 0.0725 da função custo.

O comportamento do controlador otimizado é amostrado na Fig. 4.6(a) e pode-se ver, no grafo de *torque vs tempo* uma oscilação quando a colisão com objeto é detectada (vide Fig. 4.6(b) quando o movimento é bloqueado), essa oscilação é controlada pela ação do controlador de impedância e é estabilizada seguindo a referência do torque. A ação do controlador de impedância, pode ser visualizada em outro contexto na Fig 4.6(c), onde a velocidade exigida da junta é zerada após a detecção da colisão com o objeto.

Em resumo, este capitulo propôs uma nova metodologia de sintonização de um controlador de impedância para um sistema robótico onde se tem pouca informação sobre a dinâmica do mesmo e sem a necessidade de fazer modelamento dinâmico do robô. Os coeficientes otimizados neste capitulo são intencionados para serem usados na mão robótica desenvolvida no capítulo anterior. A implementação do esquema de controle será embarcada em dispositivos FPGA e Microcontroladores no intuito de realizar comparações numéricas em termos de desempenho e tempo de execução. Dado que os controladores mudam o seu comportamento com diferentes frequências de amostragem, a frequência de $200Hz$ usada no simulador, será usada na implementação em sistemas embarcados.

Capítulo 5

Implementação da Estratégia de Controle

A mão robótica desenvolvida no trabalho atual visa embarcar os seus componentes eletrônicos de controle de baixo nível, o qual exige o uso de dispositivos eletrônicos portáveis dedicados. Atualmente, o protótipo da mão robótica inclui na palma somente a interface de adaptação dos sinais e amplificação de potência para os atuadores (vide Seção 3.3.3), deixando por fora a placa de controle dedicada ao controle de baixo nível dos dedos da mão, o qual é o esquema de controle apresentado no capítulo anterior.

O objetivo deste capitulo é apresentar a implementação do esquema de controle usando arquiteturas reconfiguráveis; tais como FPGAs (vide Seção 2.4). No intuito de identificar os componentes dos algoritmos que o esquema de controle detém, uma primeira implementação a nível médio em linguagem de programação C foi desenvolvida, para depois mapear o algoritmo em hardware usando VHDL. A plataforma usada para essa implementação é um microcontrolador; especificamente um *Atmel-ATmega2560*, o qual vem na placa de desenvolvimento Arduino MEGA. O motivo pelo qual decidiu-se usar esse microcontrolador é pela quantidade de periféricos, recursos e capacidade de processamento que o mesmo tem.

Na seguinte subseção descreve-se a plataforma de desenvolvimento usada (Arduino MEGA), a implementação do esquema de controle e os resultados do mesmo experimento usado na simulação do capítulo anterior, onde no dedo tem-se um objeto alvo de agarre para validar o algoritmo de controle.

5.1 Implementação em C-Arduino

Nessa seção é descrita a implementação de nível médio do esquema de controle da mão robótica, como já foi dito, a implementação será em linguagem de programação C e usara-se uma abordagem de descrição dos componentes *top-down*; onde primeiro se descreve o módulo mais abstrato do algoritmo de controle e seguidamente, detalham-se os sub-módulos. O módulo principal já foi descrito em alto nível na Seção 4.1 e na Fig. 4.2.

Na seguinte subseção descreve-se o pseudocódigo utilizado para implementar o esquema de controle em C.

5.1.1 Pseudocódigo da Implementação

O pseudocódigo do esquema de controle para um dedo robótico está descrito a seguir:

Algorithm 3 Pseudocódigo do esquema de controle do dedo robótico

processo DEFINIR CONSTANTES
    $motor \leftarrow$ valor do pino PWM correspondente
    $sensor_\theta, sensor_i \leftarrow$ valores dos pinos ADC correspondentes
    $R_\theta, \sigma_\theta, R_i, \sigma_i \leftarrow$ desvio padrão e covariança dos sensores
    $T_{kalman\_\theta}, T_{kalman\_i} \leftarrow$ valores dos $1/\Delta t$ correspondente a cada filtro
    $K_P, K_I, K_D \leftarrow$ constantes PID do controlador de posição
    $M, B, K \leftarrow$ constantes do controlador de impedância
    $T_{PID}, T_{Imp} \leftarrow$ valores dos $1/\Delta t$ correspondente a cada controlador
    $\tau_{sp} \leftarrow$ o valor alvo de torque de contato
processo LOOP DE CONTROLE
loop:
    se T_{kalman_θ} **então**
        $[\hat{\theta}, \hat{\omega}] \leftarrow$ kalmanPosicao($sensor_\theta$, R_θ, σ_θ)
    fim se
    se T_{kalman_i} **então**
        $\hat{i} \leftarrow$ kalmanCorrente($sensor_i$, R_i, σ_i)
    fim se
    $\tau_e \leftarrow$ torqueObservador(ω, i, v)
    se T_{Imp} **então**
        $x \leftarrow$ Impedance(τ_e, τ_{sp}, $\hat{\theta}$, $\hat{\omega}$, M, B, K)
        $\theta_{sp} \leftarrow x$
    fim se
    se T_{PID} **então**
        $v \leftarrow$ PID(θ_{sp}, $\hat{\theta}$, K_P, K_I, K_D)
        $motor \leftarrow v$
    fim se
    irpara *loop*

Saliente-se que para o algoritmo funcionar, os valores de período dos filtros e os controladores devem ser múltiplos, alem disso, as seguintes relações devem-se cumprir: $T_{kalman\_\theta} \geq T_{PID}$ e $T_{PID} = T_{Imp}$.

No pseudocódigo anterior podem-se identificar os filtros dos sensores de corrente e posição do dedo. As filtragens são implementadas usando filtros de Kalman [53]. Para o sensor de corrente é usando un filtro de Kalman linear simples, no entanto, no caso do sensor de posição o filtro também é usado para estimar a velocidade angular do dedo.

5.1.1.1 Blocos de Filtro de Kalman

O pseudocódigo para filtragem de Kalman é o apresentado a continuação.

Algorithm 4 Pseudocódigo de filtro de Kalman

função KALMAN(z_{sensor}, R, σ)
  processo PRIMEIRA ITERAÇÃO
    $my\_R \leftarrow R$
    $Q \leftarrow$ ruido do processo com covariância σ
    $F \leftarrow$ matriz do modelo em estado de transição
    $H \leftarrow$ vetor do modelo de observação
    $P \leftarrow$ valor inicial, matriz de covariância a *posteriori*
  processo ESTIMAR
    $xp = A \times z_{sensor}$
    $Pp = A \times P \times A^T + Q$ } Predição
    $K = Pp \times H^T \times [H \times Pp \times H^T + my\_R]^{-1}$
    $x = x_p + K \times [z - H \times xp]$ } Atualizar Saída
    Persistente $P = Pp - K \times H \times Pp$ } Atualizar Matriz P
fim função

Para o filtro de Kalman, os coeficientes que precisam ser projetados são R, σ, F, H e P. No caso do filtro Kalman simples para estimar corrente, esses coeficientes são determinados de maneira fácil; $R = \sigma_z^2$ (σ_z^2 é o desvio padrão), $F = 1$, $H = 1$, $P = 0$ e finalmente $\sigma = \sigma_{processo}$. No entanto, para o filtro-estimador de posição e velocidade, embora o pseudocódigo seja o mesmo, a designação dos coeficientes torna-se mais complexa.

Por simplicidade, na estimação de velocidade angular do dedo assume-se que na junta onde se encontra o sensor de posição não existe fricção e que inicialmente o dedo está estacionário em uma posição conhecida. Adicionalmente, as medidas do sensor são feitas a uma taxa de amostragem Δt e segue como é derivado o modelo de estimação com filtro de Kalman de velocidade e posição do dedo.

A posição e velocidade do dedo é descrita pela seguinte representação no espaço de estados

$$x_k = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{5.1}$$

onde x_k é a medida do sensor no instante k e $\dot{\theta}$ é a velocidade angular do dedo. Da mesma maneira, assume-se também que entre os instantes $(k-1)$ e k forças exteriores causam uma aceleração constante a_k, a qual tem uma distribuição normal com média 0 e desvio padrão σ_a. Das leis de Newton deriva-se então

$$\theta_k = Fx_{k-1} + Ga_k \tag{5.2}$$

onde $F = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$ e $G = \begin{bmatrix} \frac{1}{2}\Delta t \\ \Delta t \end{bmatrix}$ de tal maneira que

$$\theta_k = F x_{k-1} + w_k \tag{5.3}$$

onde $x \sim N(0, Q)$ e $Q = GG^T\sigma_a^2$, portanto $Q = \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix} \sigma_a^2$

Adicionalmente, a cada instante uma medida com ruído da posição real do dedo é realizada. Supondo que o ruído do sensor $v_k \sim N(0, \sigma_z)$, tem-se que

$$z_k = H x_k + v_k \tag{5.4}$$

onde $H = [1\ 0]$ e $v_k = R = E[v_k v_k^T] = [\sigma_z^2]$.

Finalmente, dado que a posição inicial do sensor é conhecida, o filtro inicializa-se com $\hat{\theta}_{0|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ e $P_{0|0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$.

Os dados de posição e velocidade estimada do filtro de Kalman de posição são usados para os controladores PID e de Impedância, em contraste a corrente estimada de do motor deve passar por mais um processo de adaptação do sinal. A seguinte subseção detalha e justifica este processo extra, o qual é incluído no bloco do observador do torque.

5.1.1.2 Bloco do Observador de Torque

O bloco do observador de torque estima o torque gerado pelo eixo do motor, essa grandeza é usada no controlador de impedância. A Eq. 4.4 determina o torque utilizando valores de tensão elétrica do motor, velocidade do eixo e corrente do motor. Dado que a tensão do motor está sendo controlada via PWM, o comportamento da corrente não é proporcional ao valor do *ciclo útil* do PWM.

O anterior foi demostrado realizando várias observações apresentadas na Fig. 5.1, nessa figura é observada como estrelas (*) azuis os valores entregues pelo sensor de corrente (eixo x) e o valor correspondente em Amperes (eixo y). Em seguida está a curva linear sugerida pelo fabricante do sensor (ACS712-5b).

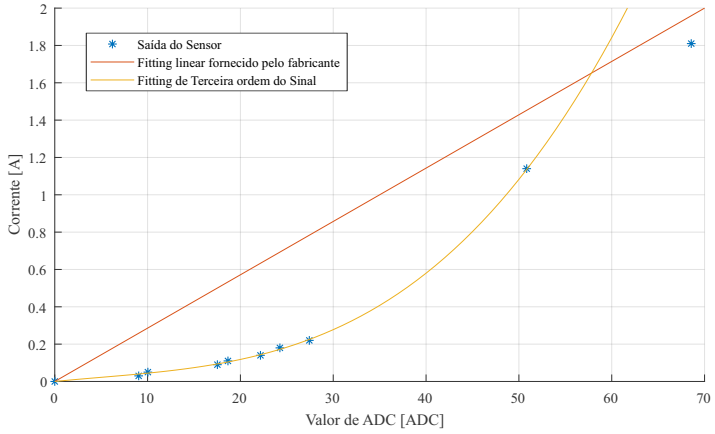


Figura 5.1: Dados do sensor de corrente com curvas de *fitting*. Fonte: o autor

.

Dado que claramente o sensor não detêm um comportamento linear, decidiu-se realizar através de regressão linear (*fitting*) uma aproximação de terceira ordem para calcular o valor correspondente de corrente para cada leitura. A aproximação calculada (linha laranja) se adapta de maneira correta até chegar aproximadamente aos $1.2A$, no entanto, para a aplicação atual, a faixa de valores de corrente usadas para este trabalho estão entre 0 e $600mA$. Essa limitação é feita porque os motores normalmente serão usados em modo de parada, no entanto, o fabricante do motor recomenda que para essas aplicações os motores devem ser usados a não mais do 40% da sua corrente máxima para evitar danos no motor. Contudo, a equação de aproximação é a seguinte

$$i = 9.5 \times 10^{-6} i_{sen}^3 - 1.4 \times 10^{-4} i_{sen}^2 + 4.9 \times 10^{-3} i_{sen} \tag{5.5}$$

Finalmente, o algoritmo que realiza o cálculo do torque é o seguinte.

Algorithm 5 Pseudocódigo do Observador do Torque

função OBSERVADOR_TORQUE(ω, v, i_{sensor}, $sensorCenter$,E)
    $i_{sen} \leftarrow i_{sensor} - sensorCenter$
    $i_{sen}\_2 \leftarrow i_{sen} \times i_{sen}$
    $i_{sen}\_3 \leftarrow i_{sen}\_2 \times i_{sen}$
    $i \leftarrow 9.5 \times 10^{-6} i_{sen}\_3 - 1.4 \times 10^{-4} i_{sen}\_2 + 4.9 \times 10^{-3} i_{sen}$
    $\omega \leftarrow$ valorAbsoluto(ω)
    se $\omega > 0.1$ **então**
        $\tau \frac{\leftarrow E \times i \times v}{\omega}$
    senão
        $\tau \frac{\leftarrow E \times i \times v}{10}$
    fim se
    devolve τ
fim função

O cálculo do torque é entregado para o bloco controle de impedância apresentado na seguinte subseção.

5.1.1.3 Bloco de Controle de Impedância

Como já foi descrito na Subseção 4.1, o método de integração é usado nesta abordagem para implementar o controle de impedância na mão robótica. O algoritmo usado nesta abordagem é apresentado do Algoritmo 6.

Algorithm 6 Pseudocódigo do Controlador de Impedância

função IMPEDANCE(τ_e, τ_{sp}, x, $\dot{x}$, M, K, B)
  processo PRIMEIRA EXECUÇÃO
    Δt constante de passo de tempo
    x_k, $\dot{x}_k$, $\ddot{x}_k \leftarrow$ posição inicial da posição do dedo e derivadas em 0
    x_{k-1}, $\dot{x}_{k-1}$, $\ddot{x}_{k-1} \leftarrow$ posição inicial das memorias das derivadas
    $my\_M \leftarrow 1/M$
    $my\_K \leftarrow K$
    $my\_B \leftarrow B$
  processo COMPUTAR
    $\tau_{error} \leftarrow \tau_{sp} - \tau_e$
    $x_{error} \leftarrow x_k - x$
    $\dot{x}_{error} \leftarrow \dot{x}_k - \dot{x}$
    $\ddot{x}_k \leftarrow my\_M \times (\tau_{error} - my\_K \times x_{error} - my\_B \times \dot{x}_{error})$
    $\dot{x}_k \leftarrow (\frac{\Delta t}{2}) \times (\ddot{x}_k + \ddot{x}_{k-1}) + \dot{x}_{k-1}$
    $x_k \leftarrow (\frac{\Delta t}{2}) \times (\dot{x}_k + \dot{x}_{k-1}) + x_{k-1}$
    persistente $x_{k-1} \leftarrow x_k$
    persistente $\dot{x}_{k-1} \leftarrow \dot{x}_k$
    persistente $\ddot{x}_{k-1} \leftarrow \ddot{x}_k$
    devolve $output \leftarrow x_k$
fim função

Neste caso, a posição x_k calculada será utilizada como *set point* para o controlador PID de posição. Na seguinte subseção é descrita a composição desse bloco.

5.1.1.4 Bloco do Controlador PID de Posição

Neste bloco é implementado um controlador PID clássico discreto. No pseudocódigo do Algoritmo 7 é apresentada a lógica do funcionamento do controlador PID implementado em C.

Algorithm 7 Pseudocódigo do Controlador PID

função PID($input_{sp}$, $input$, K_P, K_I, K_D)
    processo PRIMEIRA EXECUÇÃO
        Δt constante de passo de tempo
        $P \leftarrow K_P$
        $I \leftarrow K_I \times \Delta t$
        $D \leftarrow K_D / \Delta t$
        out_{max}, out_{min} saída máxima e mínima do controlador
        $input_{k-1}$ última posição
    processo COMPUTAR
        $error \leftarrow input_{sp} - input$
        $I_{term} \leftarrow I_{term} + (I \times error)$
        se $I_{term} > out_{max}$ **então** $I_{term} \leftarrow out_{max}$
        fim se
        se $I_{term} < out_{min}$ **então** $I_{term} \leftarrow out_{min}$
        fim se
        $\dot{input}_k \leftarrow input_k - input_{k-1}$
        $output \leftarrow P \times error + I_{term} - D \times \dot{input}$
        se $output > out_{max}$ **então** $output \leftarrow out_{max}$
        fim se
        se $output < out_{min}$ **então** $output \leftarrow out_{min}$
        fim se
        Persistente $input_{k-1} \leftarrow input_k$
        devolve $output$
fim função

No pseudocódigo os componentes out_{max} e out_{min} são os valores máximos y mínimos respectivamente que o controlador pode computar. Para o caso atual, os limites do controlador são as tensões elétricas máximas que pode executar a fonte elétrica, ou seja, $\pm 8V$. A seguir se apresenta uma descrição da plataforma utilizada para a implementação em C revistada até agora e um estudo do gasto de recursos com o algoritmo desenvolvido.

5.1.2 Resultados de Implementação

A *Arduino Mega* usada como plataforma embarcada para abordagem do esquema de controle conta com os recursos.

Tabela 5.1: Recursos da Arduino Mega utilizados na compilação do algoritmo de controle. Fonte: o autor

Caraterística	**Recurso Disponível**	**Recurso Necessário: 1 Dedo**	**Recurso Necessário: Mão Completa**
Microcontrolador	ATmega2560	-	-
Voltagem de operação	$5V$	-	-
Voltagem de Entrada	$7 - 12V$	$8V$	$8V$
Pinos Digitais de E/S	54 (15 com PWM)	2 com PWM	14 com PWM
Pinos ADC	16	2	12
Memória Flash	$256KB$ ($8KB$ usados)	$12,1200KB(5\%)$	$45,7000KB(18,5\%)$
SRAM	$8KB$	$0,7170KB(9\%)$	$2,784KB(34\%)$
EEPROM	$4KB$	$0KB$	$0KB$
Clock Speed	$16MHz$	-	-

Com intuito de testar o comportamento em um ambiente real controlado, foi feita uma emulação do experimento realizado com o simulador; ou seja, adicionou-se um objeto alvo de agarre e utilizou-se a plataforma de mão robótica (Vide Fig. 3.24). A Fig. 5.2(a) ilustra a inclusão do objeto alvo de agarre na plataforma de mão robótica, adicionalmente, a Fig. 5.2(a)-(c) ilustra o movimento realizado do dedo médio ao se flexionar e realizar o agarre do objeto.

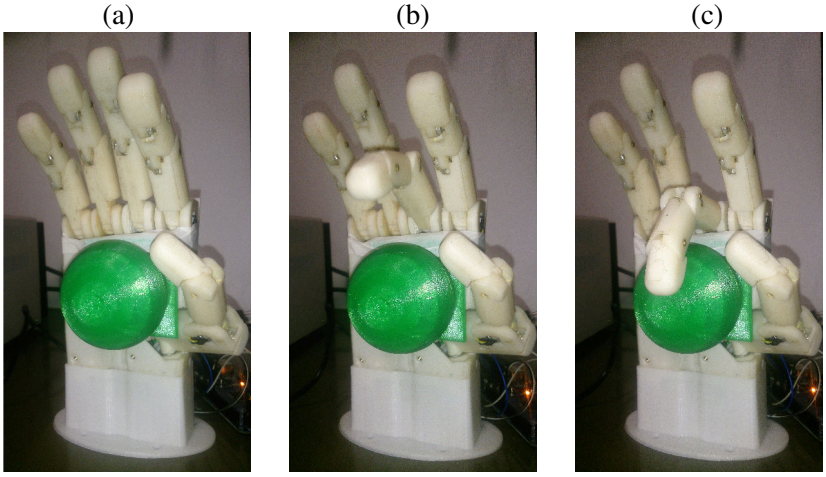


Figura 5.2: Mão robótica com bola para agarrar. As figuras (a)-(c) descrevem o movimento ao realizar o agarre da bola (vide: https://youtu.be/kGw9jN-9ns8). Fonte: o autor

Finalmente, a Fig 5.3 apresenta o resultado do controlador. Na Fig. 5.3(a) ilustra a evolução do torque em função do tempo e pode-se observar um sobre-pico alto de 299.42%, o qual era quase inexistente no simulador. Além disso, se vê da mesma maneira uma oscilação maior no estado estável, no entanto, observa-se também um tempo de contato com o objeto menor ($0.665s$). O tempo de estabilização também mudou para esta abordagem, sendo de $735ms$.

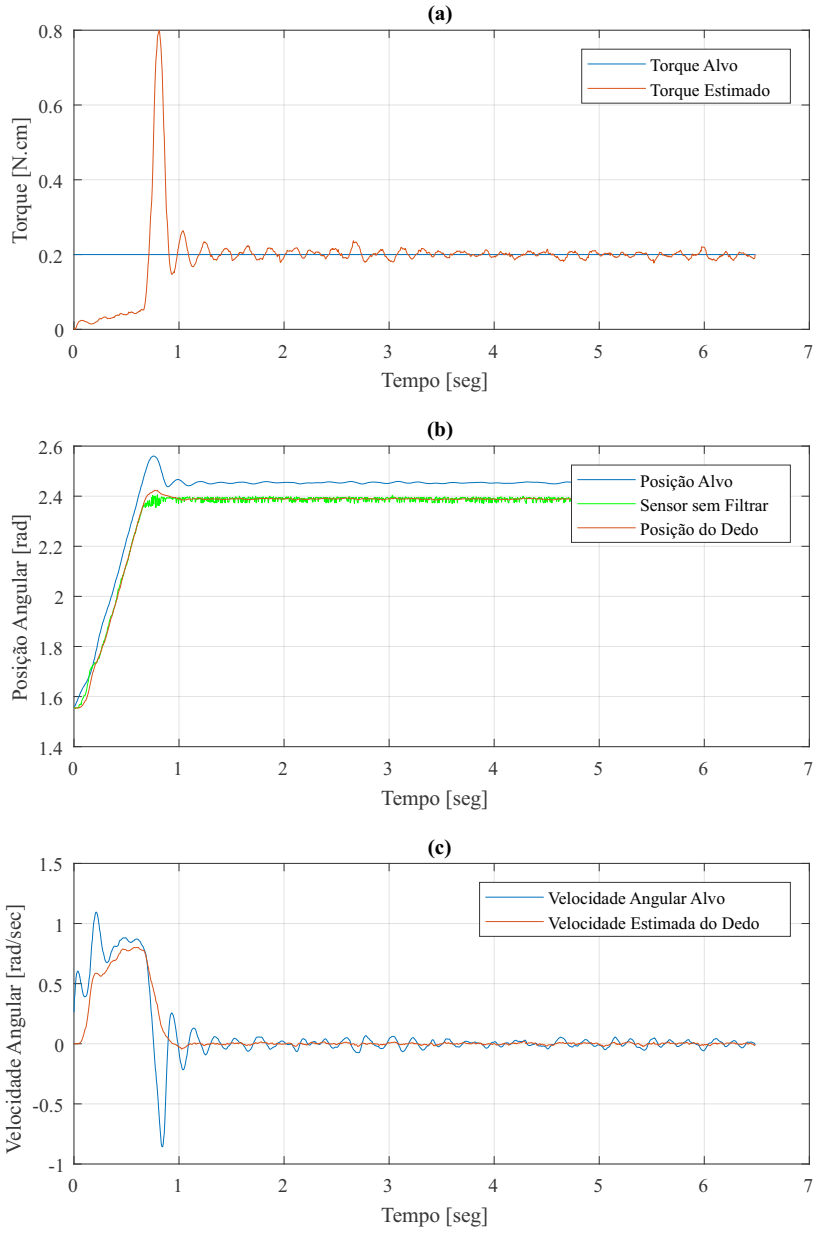


Figura 5.3: Comportamento do controlador de impedância implementado em C-Arduino, (a) Torque vs Tempo, (b) Posição vs Tempo, (a) Velocidade vs Tempo. Fonte: o autor

Adicionalmente, a Fig. 5.3(b) detalha o efeito do filtro de Kalman utilizado para o sensor de posição. Pode-se ver (na cor verde) o valor que proporciona o sensor de posição sem filtrar, em contraste do sinal atenuado (na cor vermelha) da posição estimado pelo filtro. Similarmente a Fig. 5.3(c) ilustra a evolução da velocidade estimada pelo filtro de Kalman.

Dessa maneira, conclui-se a implementação em C-Arduino do esquema de controle de cada dedo e passa-se à implementação em VHDL-FPGA que visa atingir os mesmos resultados do C, porém, com performance maior.

5.2 Implementação em VHDL-FPGA

A implementação do esquema de controle foi desenvolvida em linguagem de descrição de hardware VHDL para FPGA, a placa utilizada para a implementação do mesmo é a placa ZedBoard, a qual contém o CI Zynq XC7Z020-1CLG484 que além de outras caraterísticas, detêm um XADC integrado no chip, o qual é mister para a aplicação deste trabalho. Uma descrição mais detalhada desta placa será realizada depois nesta seção.

Cabe salientar que, embora implementar algoritmos em VHDL tem muitas vantagens, por exemplo a exploração do paralelismo que proporciona um aumento de desempenho na maioria das aplicações, o mesmo tem algumas limitações, dentre as quais a mais importante é a quantidade de recursos lógicos disponíveis. O projetista deve levar esse aspecto em consideração quando estiver implementando em VHDL.

Com o anterior em mente, é relevante esclarecer certos detalhes que demarcaram o projeto de implementação de hardware. Em particular, a decisão de utilizar a representação aritmética de números reais em ponto flutuante de $27bits$ diversamente dos padrões de *single* e *double*. A deliberação anterior produz questões de precisão nos cálculos realizados, no entanto, essa configuração de do tamanho de palavra tem sido provada pelo nosso laboratório tendo uma boa relação recursos-resolução.

Esta seção apresenta a descrição das máquinas de estados que usadas para o sistema de controle, avançando com a mesma abordagem da seção passada, onde primeiro será descrita o bloco principal e depois os blocos componentes do mesmo.

5.2.1 Descrição de Hardware

Em primeiro lugar, é importante frisar o fato que a arquitetura global do esquema de controle em hardware abrange dois processos paralelos; o primeiro processo é o que atualiza os filtros dos sensores e o segundo calcula a ação dos controladores e atualiza o driver do motor. Com isso em mente, a Fig. 5.4 ilustra as máquinas de estados finita (FSM) de ambos processos onde na esquerda está a FSM do processo de atualização dos sensores de corrente e posição e, na direita, encontra-se a FSM do proceso de atualização do driver.

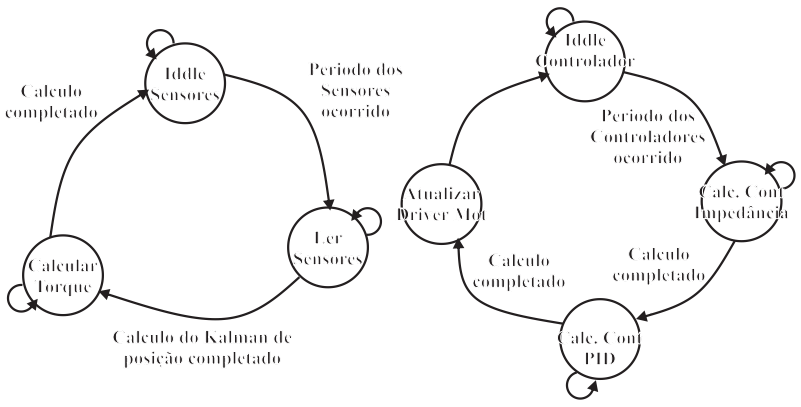


Figura 5.4: Representação da FSM do esquema de controle em hardware. Fonte: o autor

Pode-se observar que ambos processos são executados independentemente, e apenas dependem do período dos controladores e dos filtros, o que enfatiza a importância da sincronia do processo. Isto porque mesmo que os processos sejam executados independentemente, eles são dependentes em quanto aos dados que calculam. Isto se reflete na Fig. 5.5 onde se vê a dependência de dados dos blocos dos controladores com os filtros e o observador de torque. A Fig. 5.5 descreve o fluxo de dados da arquitetura proposta para a implementação em hardware do esquema de controle.

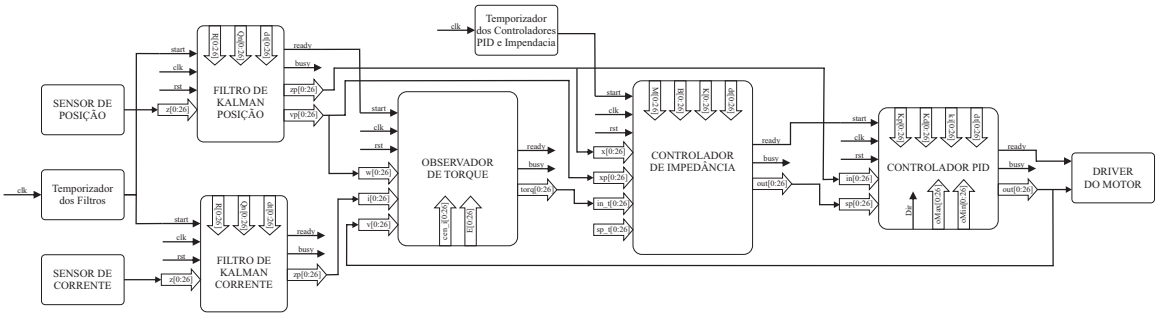


Figura 5.5: Diagrama do fluxo de dados do esquema de controle implementado em hardware. Fonte: o autor

Os blocos representados na figura anterior são descritos nas seguintes subseções, começando pelos blocos que compõem o processo de leitura dos sinais. As arquiteturas serão representadas através de diagramas de fluxo de dados separados por estados das suas respetivas máquinas de estado. Cabe enfatizar no fato que todas as arquiteturas foram implementadas usando FSM e não pipe-line. Adicionalmente, a Fig. 5.6 descreve a nomenclatura de símbolos usadas para todos os diagramas que serão presentados nesta seção.

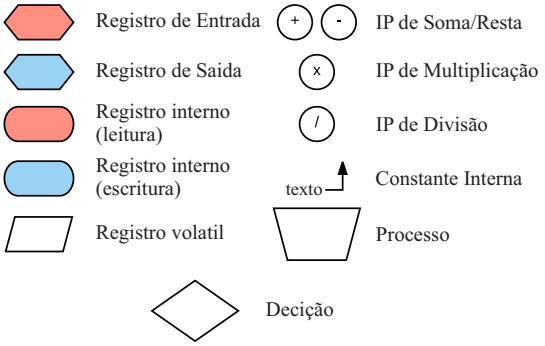


Figura 5.6: Nomenclatura de símbolos dos diagramas de fluxo de dados. Fonte: o autor

5.2.1.1 Processo de Leitura e Filtragem dos sinais

Como já foi descrito na Seção 5.1.1.1, neste projeto são implementados dois esquemas de filtros de Kalman, portanto, nesta seção serão implementadas duas arquiteturas para cada filtro. Primeiramente será descrita a filtragem simples para o sensor de corrente, apresentada na Fig. 5.7.

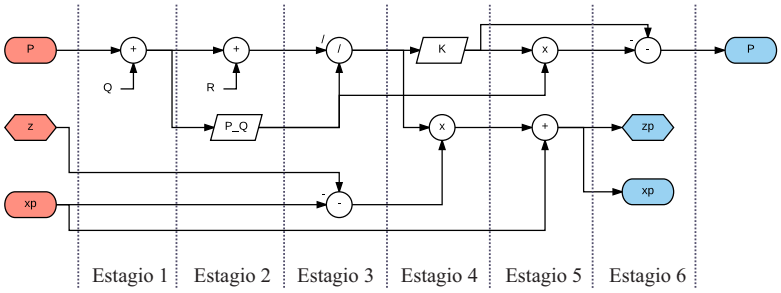


Figura 5.7: Diagrama de fluxo de dados do filtro de Kalman simples. Fonte: o autor

Devido à complexidade do filtro Kalman para estimação de posição e velocidade, decidiu-se dividir o processo em três partes com o intuito de tornar mais fácil a implementação. As partes no qual foi dividido o processo de filtragem estão descritas no Algoritmo 4 (predição, atualização das saídas e atualização da matriz P). A Fig. 5.8(a)-(c) descreve as fases da implementação do filtro de Kalman para posição.

Cabe salientar que na implementação em hardware do filtro de Kalman de posição, os valores de saída zp e vp são calculados antes de acabar todos os cálculos, é por isso que se implementou na arquitetura a possibilidade de poder acessar a esse dado antes do bloco ficar em modo de espera. Os dados dos filtros são encaminhados para o bloco observador de torque e os controladores PID e impedância.

O bloco observador de torque computa os dados do filtro de Kalman de corrente para interpretá-los em amperes para poder calcular o torque gerado. Portanto, o bloco observador de torque não realiza a equação de torque simplesmente, mas também calcula a equação que converte os dados de corrente descrita na Eq. 5.5. O processo realizado dentro do bloco é ilustrado na Fig. 5.9.

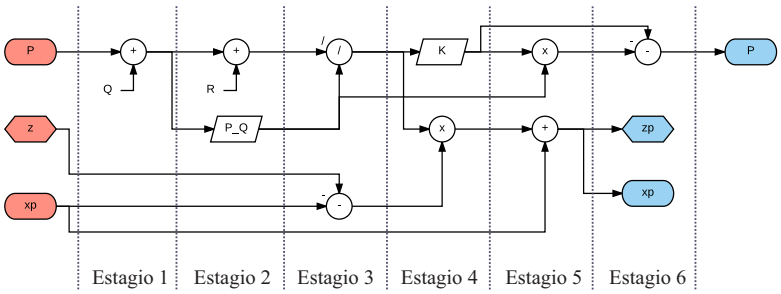


Figura 5.9: Diagrama de fluxo de dados do observador de torque. Fonte: o autor

Após de calcular o torque, o processo de leitura dos sinais é colocado de novo em modo de espera até o relógio indicar que é o momento de executar esta tarefa de novo. A continuação descrevem-se os blocos componentes do processo de controle da arquitetura.

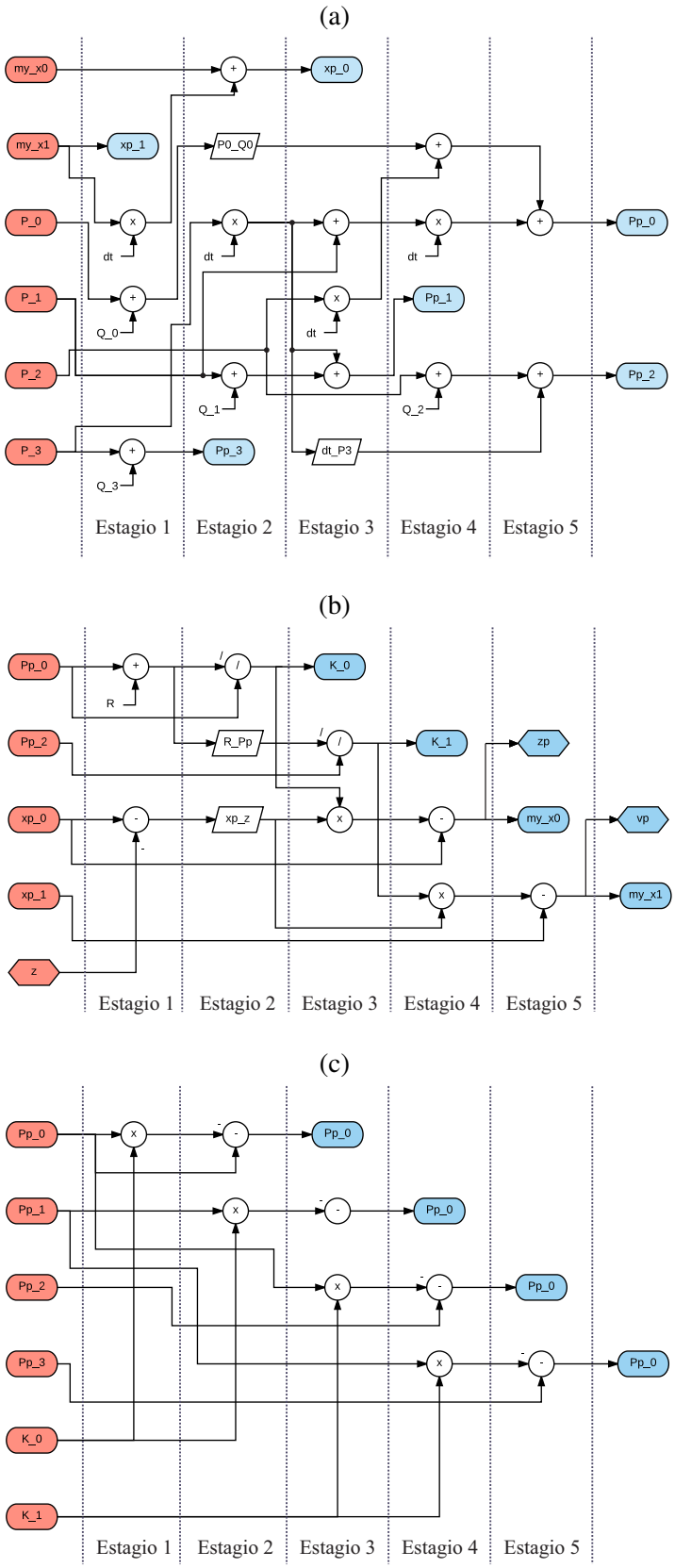


Figura 5.8: Diagrama de fluxo de dados do filtro de Kalman para posição. (a) Predição, (b) atualização das saídas e (c) atualização da matriz P Fonte: o autor

5.2.1.2 Blocos de controle cinemático e dinâmico

O controlador de impedância é ativado quando o temporizador do processo indica que é o momento de calcular a ação dos controladores. O primeiro bloco a ser calculado é o bloco de controle de impedância o qual recebe os dados do processo anterior e calcula o ponto alvo que deve atingir o dedo para conseguir a dinâmica desejada do dedo, assim, essa posição é enviada para o controlador PID o qual calcula a ação que deve realizar o motor. As Fig. 5.10-5.11 detalham as arquiteturas implementada para emular esses blocos.

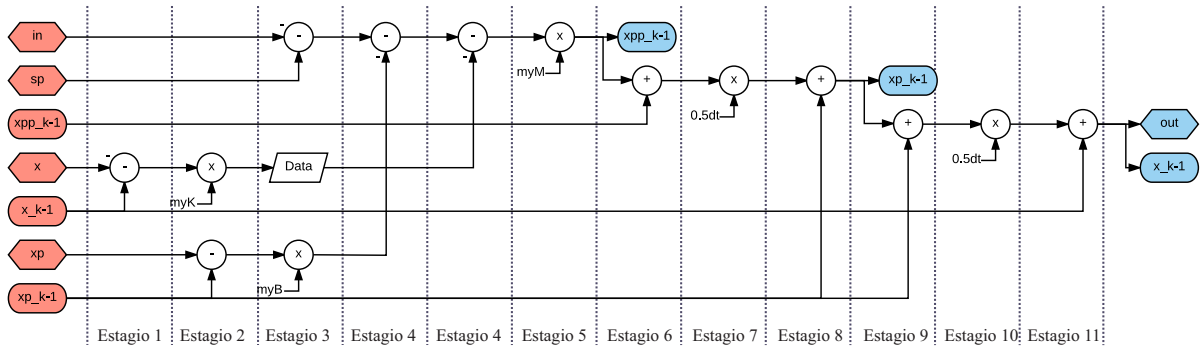


Figura 5.10: Diagrama de fluxo de dados do controlador de impedância. Fonte: o autor

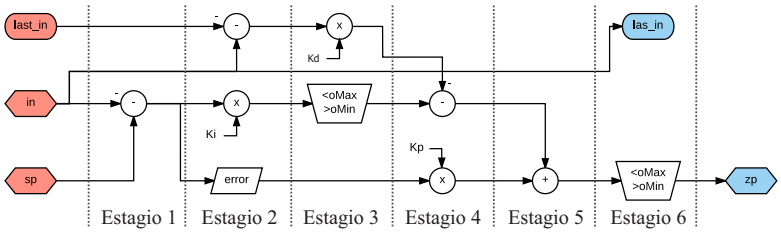


Figura 5.11: Diagrama de fluxo de dados do controlador PID. Fonte: o autor

Finalmente, depois de calcular a ação do motor, este dado é enviado para o driver do mesmo.

5.2.2 Resultado e Análise de Síntese

O análise de performance do mapeamento em FPGA foi realizado via simulação numérica comparando os resultados com a abordagem realizada em Arduino e Matlab. A comparação comportamental dos cálculos dos módulos de filtragem e de estimação de torque são apresentados na Fig. 5.14(a)-(c).

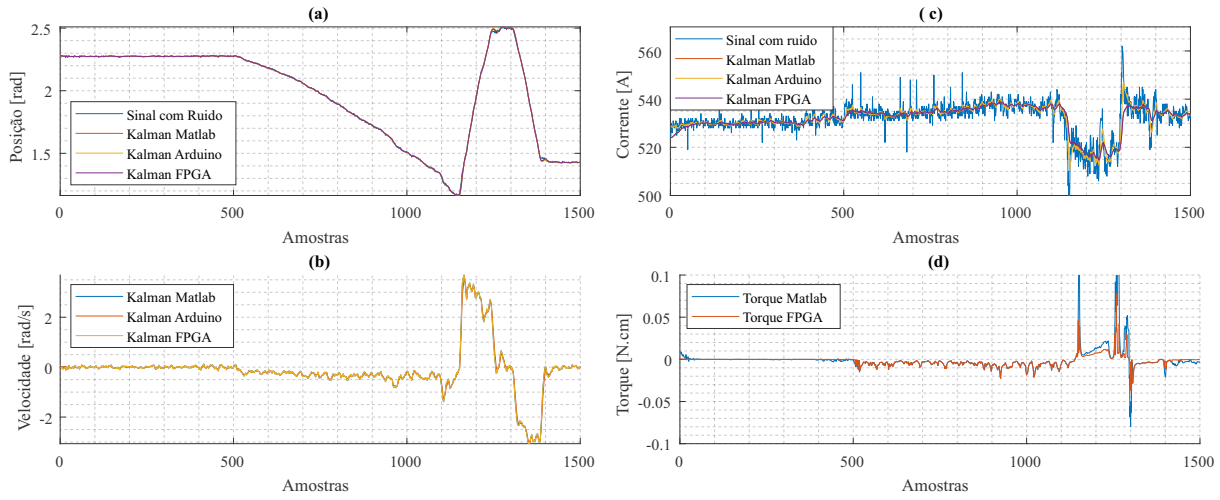


Figura 5.12: Simulação comportamental dos módulos de filtragem de corrente e posição e estimação de torque, (a) Posição vs Amostra, (b) Velocidade vs Amostra e, (c) Corrente vs Amostra, (d) Torque vs Amostra. Fonte: o autor

A Fig. 5.13 apresenta o MSE ponto a ponto dos cálculos da FPGA (em ponto flotante de $27bits$) com Matlab (em ponto flotante *double* $64bits$) pode-se observar que na maioria das amostras o erro é menor que o valor da tolerância de erros (5%), porém, para algumas amostras de velocidade e torque o erro alcança valores muito altos, contudo, estes erros são o resultado de valores muito baixos produzidos quando o dedo não está executando nenhuma ação e, portanto, esses valores são desprezíveis.

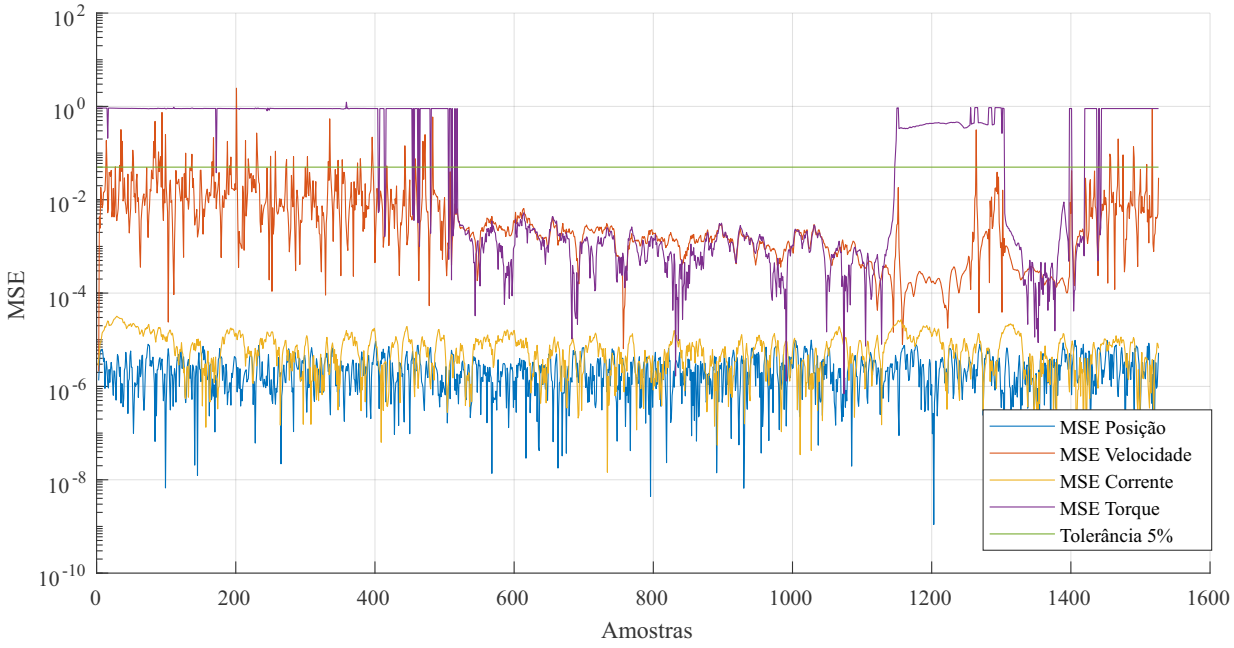


Figura 5.13: MSE da precisão dos cálculos realizados em FPGA vs Matlab. Fonte: o autor.

Consequentemente, estes erros altos não afetam o performance do controlador mapeado o qual é apresentado na Fig. 5.14, a qual ilustra dito comportamento realizando o mesmo experimento de agarre via simulação com o mapeamento em FPGA, para o qual foram coletados os dados de entrada (posição e corrente) usados pelo Arduíno.

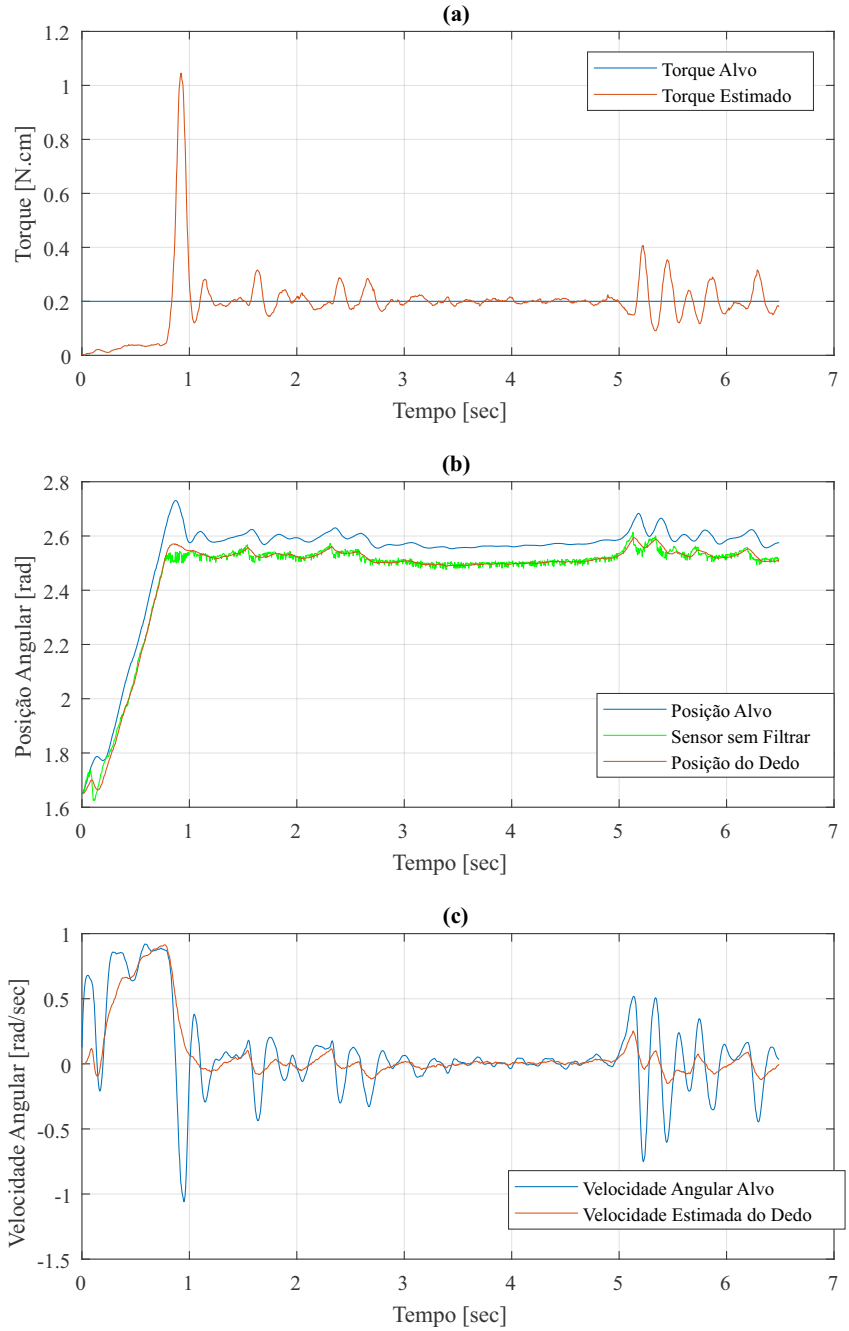


Figura 5.14: Comportamento do controlador de impedância mapeado em FPGA, (a) Torque vs Tempo, (b) Posição vs Tempo, (c) Velocidade vs Tempo. Fonte: o autor

A plataforma alvo do mapeamento em hardware do esquema de controle é uma ZedBoard que embarca o chip XC7Z020-1CLG484, a utilização de recursos da implementação do esquema de controle para um dedo é o apresentado na Tabela 5.2.

Tabela 5.2: Recursos utilizados da FPGA para o controle de um dedo

Módulo	**Kal** i	**Kal** ω	τ **Obs**	**Imp**	**PID**	Modulo Top		
Recurso	Uso	Uso	Uso	Uso	Uso	Uso	Uso %	Disponível
Slice LUT	813	1468	925	616	610	4522	8.5	53200
Slice Register	523	983	531	332	359	3230		
DSP	2	2	3	1	1	10	4.55	220

Fonte: o autor.

Adicionalmente, a arquitetura do controlador foi replicada 7 vezes na FPGA com o intuito de controlar os 7 GDLs da mão e conferir que a arquitetura não sobrepassa os recursos do CI. O uso de recursos está detalhado na Tabela 5.3. No entanto, isto seria uma estimação pessimista porque não todos os dedos tem sensor de corrente, portanto, O consumo de recursos final vai obrigatoriamente ser menor.

Tabela 5.3: Recursos utilizados da FPGA mapeando a arquitetura para a mão completa.

Recurso	Uso	Uso %	Disponível
Slice LUT	31907	59.98	53200
Slice Register21794	20.48	106400	
DSP	70	31.82	220

Fonte: o autor.

A titulo de comparação com os resultados obtidos na implementação com Arduíno se realiza a Tabela 5.4, nela pode-se ver os tempos de execução de cada módulo. Saliente-se que em Arduino a execução dos módulos é sequencial; ou seja, apenas é possível executar um modulo cada vez e, portanto o tempo de execução é a soma do tempo de cada modulo. Desta maneira, conclui-se que a frequência máxima da implementação em Arduino é aproximadamente de $799Hz$ operando com um clock de 16MHZ. Em contraste, a arquitetura de hardware proposta paraleliza os blocos de filtragem além de explorar o paralelismo dos algoritmos com as FSM descritas anteriormente. Operando com um clock de 100 MHz, a frequência máxima alcançada para o controlador em FPGA foi de $813KHz$.

Tabela 5.4: Comparação de desempenho das implementações (Arduino vs FPGA)

Módulo	Dispostivo				Speed Up
	Arduino		FPGA		
	t [μs]	Cic/Rel	t[μs]	Cic/Rel	
Kalman Corrente	234	3744	0.25	50	74.88
Kalman Posição	668	10688	0.5	100	106.88
Torque Observador	110	1760	0.19	38	46.31
Controlador de Impedancia	153	2448	0.37	74	33.08
Controlador PID	86	1376	0.17	34	40.47

Fonte: o autor.

Na Tabela 5.3 a coluna de *speed-up* é a relação de incremento de performance no ponto de vista dos ciclos de maquina necessários para completar os algoritmos. A Fig. 5.15 exibe de maneira grafica o mapeamento realizado pela ferramento de síntese e implementação em FPGA *Vivado 2016.4*, na figura pode-se observar na esquerda a implementação de apenas um dedo e destacando cada módulo da arquitetura proposta implementada na FPGA. Na direita está o mapeamento para 7 módulos da mão enfatizando a locação do módulo para cada dedo.

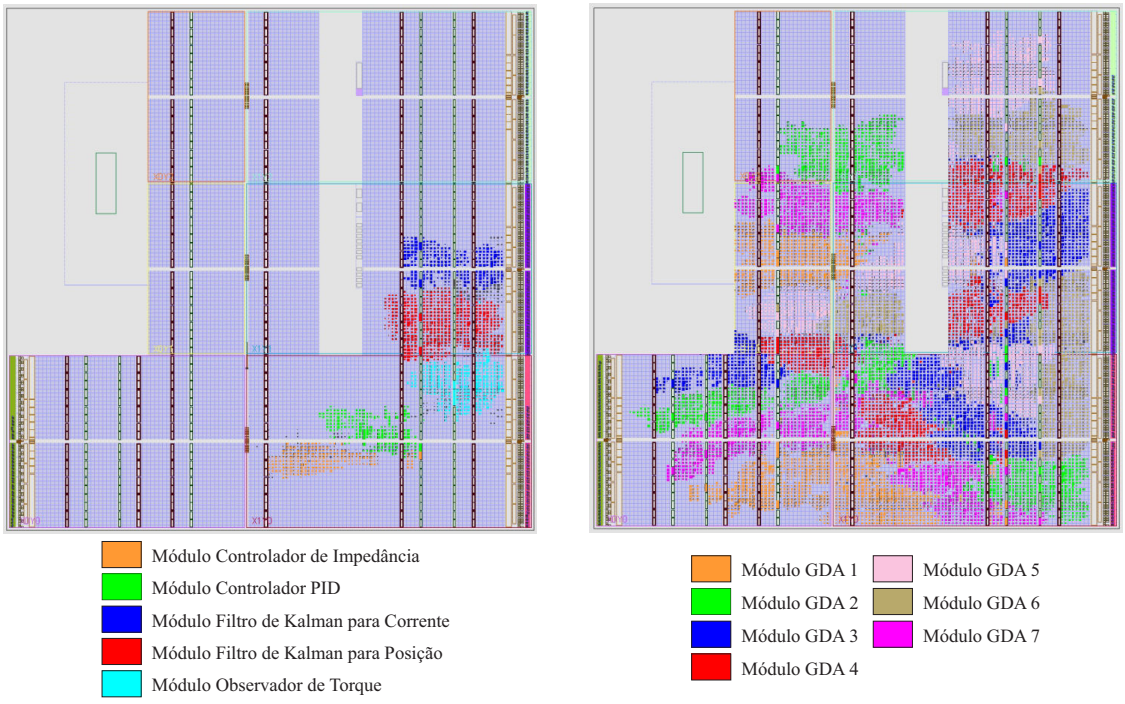


Figura 5.15: Mapeamento dos módulos das arquiteturas na FPGA Zynq XC7Z020-1CLG484 para apenas um dedo e para 7 GDA

5.3 Conclusões

Na implementação do esquema de controle em C-Arduino e VHDL-FPGA demostrou-se que o controlador segue a referência de maneira aceitável, no entanto, destaca-se que o comportamento em ambos casos difere em grande quantidade com o demonstrado no simulador, o qual é menos sub-amortecido. Cabe salientar que esta diferencia era de se esperar dado que o simulador não tinha em consideração fenômenos físicos complexos, tais como fricções entre as peças, folgas e tolerâncias, entre outros. Mesmo assim, a otimização dos coeficientes do controlador de impedância realizada no capítulo anterior continua sendo justificada e não se considera necessário otimizar o controlador com a plataforma real, isto é, porque para poder realizar este tipo de otimização seria mister realizar uma quantidade de experimentos que poderiam causar danos permanentes no protótipo. Esses não seriam necessariamente causados pelo desgaste de realizar numerosos experimentos, mas sim porque a escolha aleatória dos coeficientes do controlador poderiam causar um comportamento instável que potencialmente quebraria o protótipo.

A implementação de bibliotecas de Arduino para aplicar controle de impedância e filtros de Kalman é uma contribuição para a comunidade desenvolvedora de Arduino, dado que as mesmas não existiam até agora. O repositório das biblioteca vai ser publicado e submetido no site oficial de Arduino como parte dos resultados deste trabalho.

Capítulo 6

Conclusões

Neste trabalho apresentou-se o desenvolvimento de um sistema embarcado baseado em arquiteturas reconfiguráveis do controle dinâmico de uma mão robótica sintonizado com algoritmos bioinspirados aplicada ao problema de resolver agarres. Foi realizado o projeto mecânico da plataforma de mão biomimética, a qual contem 7 GDA, sensores de posição e de corrente em cada atuador. Os movimentos de flexão-extensão dos dedos são realizados com mecanismos de 4 elos. A otimização bioinspirada dos mecanismos resolveu um mecanismo que gera as trajetórias necessárias que emulam o movimento de uma mão humana. O processo de otimização atingiu um erro quadrático médio 0.00266.

O uso destes mecanismos permitiu a implementação de uma plataforma de mão biomimética com dimensiones e peso similar a uma mão real: $210 \times 84 \times 38\ mm$ com $413.13gr$ (projeto de mão robótica) versus $175 \times 89 \times 43\ mm$ com $409.5gr$ (mão humana). Adicionalmente, na implementação da mão completa comprovou-se a destreza da oponibilidade do polegar com o teste de Kapandji, no qual o protótipo conseguiu atingir todos os 10 níveis de posicionamento do dedo.

No entanto, a aptidão da mão para realizar agarres não vem dada apenas pela oponibilidade do polegar. Usualmente para este tipo de tarefas é necessário realizar estratégias de controle dinâmico ou de força para controlar as forças de contato com o ambiente. Com o anterior em mente, desenvolveu-se um controlador de impedância o qual primeiramente foi avaliado via simulação numérica. Para tal ação, um simulador do sistema de um dedo robótico foi realizado, o qual permitiu sintonizar os parâmetros controlador sem colocar em perigo o protótipo.

A sintonização do controlador de impedância foi realizada com algoritmos de otimização bioinspirada, especificamente o PSO. O processo de otimização resultou numa resposta sub-amortecido com tempo de subida do controlador de $355ms$ e erro em estado estável de 1.2%. Este método bioinspirado de sintonização para um controlador de impedância de um sistema robótico é uma nova abordagem e é uma das contribuições deste trabalho.

As implementações do controlador em dispositivos embarcados foram realizadas através de duas abordagens; um de nível médio em linguagem de programação C usando Arduino e a outra, mapeando os algoritmos em uma arquitetura de hardware em FPGA.

A implementação em Arduino demonstrou que a sintonização bioinspirada do controlador de impe-

dâncias obteve resultados corretos, no entanto, se observou que o comportamento não foi completamente igual, isto porque no simulador não foram levadas em consideração fenômenos físicos complexos como fricção dinâmica dos componentes, folgas e tolerâncias do dedo. No entanto, a resposta continua sendo estável, o que justifica a implementação anterior de sintonização bioinspirada do controlador.

Finalmente, a implementação do controlador de impedância em FPGA entregou uma resposta similar à reposta obtida com o Arduino. Mesmo usando representação aritmética de menor resolução, o erro se manteve na tolerância máxima de 5% na resposta dinâmica.

A implementação em FPGA melhorou a frequência do controlador, chegando a uma frequência máxima de $813KHz$ em comparação com os $799Hz$ da implementação em Arduino. Cabe salientar que devido ao paralelismo da FPGA, essa frequência é válida também quando se tem todos os dedos embarcados no mesmo chip. Em contraste com trabalhos correlatos o mapeamento em FPGA, também melhorou as respostas da KITECH-Hand [16] ($333Hz$), da LMS Hand [20] ($50Hz$) e a HIT/DLR Hand [25] ($1KHz$).

Outras melhorias qualitativas do projeto de mão robótica em comparação com outros trabalhos correlatos, é a inclusão do GDL de adução/abdução independente e robotizada inexistentes em trabalhos como Calderon *et al.* [17], Wang *et al.* [19], HIT/DLR Hand [25] e ISR SoftHand [27]. Em adição, é importante ressaltar que durante o desenvolvimento da plataforma foi alcançado um tamanho reduzido, similar a uma mão humana em comparação com outras abordagens onde isto não foi alcançado como KITECH-Hand [16], HIT/DLR Hand [25] e ISR SoftHand [27].

6.1 Trabalhos Futuros

Como trabalhos futuros, está o desenvolvimento da integração da arquitetura de hardware com o XADC embarcado na FPGA para desenvolver o controle dinâmico em tempo real na placa de desenvolvimento. Com o sistema desenvolvido completamente em FPGA propõe-se realizar mais experimentos com os diferentes tipos taxionômicos de agarres apresentados na Fig. 2.4 e analisar como é o comportamento do controlador ao realizá-los, para assim propor outras estratégias de agarre auxiliadas pelo controlador de impedância a baixo nível desenvolvido neste trabalho.

Com o intuito de embarcar o controle de baixo nível completamente na mão robótica, outro trabalho futuro é desenvolver uma placa controladora baseada em FPGA, na qual seria necessário utilizar de novo o laboratório de eletrônica do IFB.

O uso de mecanismos de redução de movimentos como o sem fim-coroa foi necessário para atingir as forças necessárias para realizar agarres de potência, no entanto, a caraterística auto-travante do mesmo, dificulta o controle dinâmico e manipulabilidade. Retirar esses mecanismos significaria usar motores mais potentes e, portanto mais caros.

O trabalho de Kitech-Hand [16] propôs uma nova abordagem para a implementação dos GDL de adução/abdução dos dedos indicador a mínimo, o qual entregaria uma destreza maior à proposta atual.

Através do análise cinemático foi descoberto mais uma possível aplicação para otimização bioinspirada, a qual é a otimização do angulo de oponibilidade para atingir uma área comum com os outros dedos maior possível (problema de maximização).

REFERÊNCIAS BILBLIOGRÁFICAS

[1] R. Vepa, "Biomimetic Robotics," in *Engineered Biomimicry*, Newnes, Ed. Elsevier, 2013, pp. 81–105. [Online]. Available: http://www.books24x7.com/marc.asp?bookid=30924http://linkinghub.elsevier.com/retrieve/pii/B9780124159952000040

[2] E. Mattar, "A survey of bio-inspired robotics hands implementation: New directions in dexterous manipulation," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 517–544, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2012.12.005

[3] Z. Kappassov, J.-A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands — Review," *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, dec 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0921889015001621

[4] D. Prattinchizzo and J. C. Trinkle, "Grasping," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2008, ch. 28, pp. 671–700.

[5] S. Seok, D. J. Hyun, S. Park, D. Otten, and S. Kim, "A highly parallelized control system platform architecture using multicore CPU and FPGA for multi-DoF robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014, pp. 5414–5419. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6907655

[6] Y.-f. Li and C.-w. Huang, "Force control for the fingers of the piano playing robot - A gain switched approach," in *2015 IEEE 11th International Conference on Power Electronics and Drive Systems*, no. June. IEEE, jun 2015, pp. 265–270. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7203429

[7] S. Pertuz, "DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO DE MANO ROBÓTICA MULTI-DEDOS," Ph.D. dissertation, University of Pamplona, Pamplona - Norte de Santander / Colombia, 12 2014.

[8] A. Barrientos, L. Peñín, C. Balaguer, and R. Aracil, *Fundamentos de robótica*. Madrid McGrawHill, 2007. [Online]. Available: http://www.lavoisier.fr/notice/frNWOARSOARSWKKS.html

[9] L. Villani and J. De Schutter, "Force Control," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2008, ch. 7, pp. 161–183.

[10] M. Tavakoli, B. Enes, J. Santos, L. Marques, and A. T. de Almeida, "Underactuated anthropomorphic hands: Actuation strategies for a better functionality," *Robotics and Autonomous Systems*, vol. 74, pp. 267–282, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0921889015001888

[11] T. Feix, J. Romero, H.-b. Schmiedmayer, A. M. Dollar, and D. Kragic, "The GRASP Taxonomy of Human Grasp Types," *IEEE Transactions on Human-Machine Systems*, pp. 1–12, 2015. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7243327

[12] K. Masselos and N. S. Voros, "Introduction to Reconfigurable Hardware," in *System Level Design of Reconfigurable Systems-on-Chip*. Berlin/Heidelberg: Springer-Verlag, 2005, ch. 1, pp. 15–26. [Online]. Available: http://link.springer.com/10.1007/0-387-26104-4-1

[13] S. Hauck and A. DeHon, *Reconfigurable computing: the theory and practice of FPGA-based computation*, 2010.

[14] T. Weise, *Global Optimization Algorithms–Theory and Application*, 2009, vol. 1. [Online]. Available: http://www.it-weise.de/projects/book.pdf$\delimiter"026E30F$nhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.64.8184{&}rep=rep1{&}type=pdf

[15] R. Storn and K. Price, "Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces," *Technical report, International Computer Science Institute*, vol. 11, no. TR-95-012, pp. 1–15, 1995. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.5398{&}amp;rep=rep1{&}amp;type=pdf

[16] D.-H. Lee, J.-H. Park, S.-W. Park, M.-H. Baeg, and J. Bae, "KITECH-Hand: A Highly Dexterous and Modularized Robotic Hand," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 2, pp. 876–887, apr 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7763772/

[17] C. A. Calderon, C. Ramirez, V. Barros, and G. Punin, "Design and Deployment of Grasp Control System applied to robotic hand prosthesis," *IEEE Latin America Transactions*, vol. 15, no. 2, pp. 181–188, feb 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7854610/

[18] S. H. Jeong, K.-S. Kim, and S. Kim, "Designing Anthropomorphic Robot Hand with Active Dual-Mode Twisted String Actuation Mechanism and Tiny Tension Sensors," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1–1, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7805142/

[19] X. Wang, Y. Liu, D. Yang, N. Li, L. Jiang, and H. Liu, "Progress in the biomechatronic design and control of a hand prosthesis," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, oct 2010, pp. 5880–5885. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5651172

[20] N. Daoud, J. Gazeau, S. Zeghloul, and M. Arsicault, "A real-time strategy for dexterous manipulation: Fingertips motion planning, force sensing and grasp stability," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 377–386, mar 2012. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2011.07.011http://linkinghub.elsevier.com/retrieve/pii/S0921889011001333

[21] J. Gazeau, S. Zehloul, M. Arsicault, and J. Lallemand, "The LMS hand: force and position controls in the aim of the fine manipulation of objects," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 3. IEEE, 2001, pp.

2642–2648. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=933021http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=933021

[22] S. Lee, S. Noh, Y. Lee, and J. H. Park, "Development of bio-mimetic robot hand using parallel mechanisms," in *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, dec 2009, pp. 550–555. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5420706

[23] D. Zhao, L. Jiang, H. Huang, M. Jin, H. Cai, and H. Liu, "Development of a Multi-DOF Anthropomorphic Prosthetic Hand," in *2006 IEEE International Conference on Robotics and Biomimetics*. IEEE, 2006, pp. 878–883. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4141981

[24] H. Huang, Li Jiang, Y. Liu, L. Hou, H. Cai, and H. Liu, "The Mechanical Design and Experiments of HIT/DLR Prosthetic Hand," in *2006 IEEE International Conference on Robotics and Biomimetics*, no. 50435040. IEEE, 2006, pp. 896–901. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4141984

[25] H. Liu, K. Wu, P. Meusel, N. Seitz, G. Hirzinger, M. Jin, Y. Liu, S. Fan, T. Lan, and Z. Chen, "Multisensory five-finger dexterous hand: The DLR/HIT Hand II," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, sep 2008, pp. 3692–3697. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4650624

[26] T. Lan, Y. Liu, M. Jin, S. Fan, H. Fang, J. Xia, and H. Liu, "DSP&FPGA-based joint impedance controller for DLR/HIT II dexterous robot hand," in *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, jul 2009, pp. 1594–1599. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5229817

[27] M. Tavakoli and A. T. de Almeida, "Adaptive under-actuated anthropomorphic hand: ISR-SoftHand," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, no. Iros. IEEE, sep 2014, pp. 1629–1634. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6942773

[28] J.-U. Chu, D.-H. Jung, and Y.-J. Lee, "Design and control of a multifunction myoelectric hand with new adaptive grasping and self-locking mechanisms," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, may 2008, pp. 743–748. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4543294

[29] D. H. Jeong, J. U. Chu, and Y. J. Lee, "Development of myoelectric hand with infrared led-based tactile sensor," *Journal of Institute of Control, Robotics and Systems*, vol. 15, no. 8, pp. 831–838, aug 2009.

[30] H. van Duinen and S. C. Gandevia, "Constraints for control of the human hand," *The Journal of Physiology*, vol. 589, no. 23, pp. 5583–5593, dec 2011. [Online]. Available: http://doi.wiley.com/10.1113/jphysiol.2011.217810

[31] S. Cobos, "ANALISIS DE TAREAS DE MANIPULACION VIRTUAL BASADAS EN MODELOS REDUCIDOS DE LA MANO HUMANA," Ph.D. dissertation, Universidad Politécnica de Madrid, 2010.

[32] K. Waldron and J. Schimiedeler, "Kinematics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2008, ch. 1, pp. 9–33.

[33] S. R. L. Prensilia, "IH2 Azzurra Series. Self-Contained Robotic Hand Getting Started Guide," Peccioli, pp. 1–68, 2014. [Online]. Available: http://www.prensilia.com/files/support/doc/PRENSILIA{_}IH2{_}basic{_}16.pdf

[34] Xinqing Wang, Yiwei Liu, Dapeng Yang, Nan Li, Li Jiang, and Hong Liu, "Progress in the biomechatronic design and control of a hand prosthesis," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, oct 2010, pp. 5880–5885. [Online]. Available: http://ieeexplore.ieee.org/document/5651172/

[35] Jun-Uk Chu, Dong-Hyun Jung, and Yun-Jung Lee, "Design and control of a multifunction myoelectric hand with new adaptive grasping and self-locking mechanisms," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, may 2008, pp. 743–748. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4543294

[36] S. R. Company, "Shadow Dexterous Hand Technical Specification," Shadow Robot Company, Tech. Rep. August, 2015. [Online]. Available: http://www.shadowrobot.com/products/dexterous-hand/

[37] R. Wilcox, "Kolmogorov-Smirnov Test," pp. 83–90, 1998.

[38] W. H. Kruskal and W. A. Wallis, "Use of Ranks in One-Criterion Variance Analysis," *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, dec 1952.

[39] A. Kapandji and M. Torres, *Fisiología articular: esquemas comentados de mecánica articular. Hombro, codo, pronosupinacion, muñeca, mano. 1*, 6th ed., ser. Fisiología articular. Tomo 1. Hombro, codo, pronosupinación, muñeca, mano, M. Torres, Ed. Editorial Médica Panamericana, 2006. [Online]. Available: https://books.google.es/books?id=Fu1WNgAACAAJ

[40] D. Johansen, C. Cipriani, D. B. Popovic, and L. N. S. A. Struijk, "Control of a Robotic Hand Using a Tongue Control System—A Prosthesis Application," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 7, pp. 1368–1376, jul 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7381630/

[41] M. Beschi, E. Villagrossi, L. M. Tosatti, and D. Surdilovic, "Sensorless model-based object-detection applied on an underactuated adaptive hand enabling an impedance behavior," *Robotics and Computer-Integrated Manufacturing*, vol. 46, no. November 2016, pp. 38–47, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.rcim.2016.11.005

[42] J. Shi and G. S. Koonjul, "Real-time grasping planning for robotic bin-picking and kitting applications," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, vol. 2015-Octob, no. 2. IEEE, aug 2015, pp. 1632–1637. [Online]. Available: http://ieeexplore.ieee.org/document/7294334/

[43] C. Cipriani, J. L. Segil, J. A. Birdwell, and R. F. ff Weir, "Dexterous Control of a Prosthetic Hand Using Fine-Wire Intramuscular Electrodes in Targeted Extrinsic Muscles," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 828–836, jul 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6718168/

[44] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2015-Decem. IEEE, sep 2015, pp. 4415–4420. [Online]. Available: http://ieeexplore.ieee.org/document/7354004/

[45] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-Driven Grasp Synthesis - A Survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, apr 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6672028/

[46] D. Choi, S. Shin, K. Lee, J. C. Koo, H. R. Choi, and H. Moon, "Motion planning of multifingered robotic hand for turning the cap," in *2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. IEEE, nov 2011, pp. 451–454. [Online]. Available: http://ieeexplore.ieee.org/document/6145862/

[47] F. Petit, A. Dietrich, and A. Albu-Schaffer, "Generalizing Torque Control Concepts: Using Well-Established Torque Control Methods on Variable Stiffness Robots," *IEEE Robotics & Automation Magazine*, vol. 22, no. 4, pp. 37–51, dec 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7349335/

[48] N. Hogan, "Impedance Control: An Approach to Manipulation: Part II—Implementation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, p. 8, 1985. [Online]. Available: http://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1403623

[49] ——, "Impedance Control: An Approach to Manipulation: Part I—Theory," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, p. 1, 1985. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=4788393{%}5Cnpapers2://publication/uuid/B3C531B3-9D3B-4A24-B579-115FB24F446Bhttp://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1403621

[50] F. Caccavale, C. Natale, B. Siciliano, and L. Villani, "Integration for the next generation: embedding force control into industrial robots," *IEEE Robotics & Automation Magazine*, vol. 12, no. 3, pp. 53–64, sep 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1511869/

[51] J. G. Ziegler and N. B. Nichols, "Optimum Settings for Automatic Controllers," *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, no. 2B, p. 220, 1993. [Online]. Available: http://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1406231

[52] S. Shinners, *Modern Control System Theory and Design*, ser. A Wiley interscience publication. Wiley, 1998.

[53] P. Kim and L. Huh, *Kalman Filter for Beginners: With MATLAB Examples*. CreateSpace Independent Publishing Platform, 2011. [Online]. Available: https://books.google.com.br/books?id=W8u_XwAACAAJ