



**UNIVERSIDADE ESTADUAL DE CAMPINAS**  
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

AMPARO DIAZ MUÑOZ

**Desenvolvimento de uma Interface  
Homem-Máquina para o Controle de  
Próteses de Membros Superiores usando  
Deep Learning**

Campinas  
2020

AMPARO DIAZ MUNOZ

DESENVOLVIMENTO DE UMA INTERFACE HOMEM-MÁQUINA PARA O  
CONTROLE DE PRÓTESES DE MEMBROS SUPERIORES USANDO DEEP  
LEARNING

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na Área de Engenharia da Computação.

*Supervisor/Orientador:* Prof. Dr. Eric Rohmer

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DISSERTAÇÃO DEFENDIDA PELO(A) ALUNO(A) AMPARO DIAZ MUNOZ, E ORIENTADA PELO(A) PROF. DR. ERIC ROHMER

Campinas  
2020

Ficha catalográfica  
Universidade Estadual de Campinas  
Biblioteca da Área de Engenharia e Arquitetura  
Rose Meire da Silva - CRB 8/5974

D543d      Diaz Munoz, Amparo, 1982-  
Desenvolvimento de uma interface homem-máquina para o controle de  
próteses de membros superiores usando deep learning / Amparo Diaz Munoz.  
– Campinas, SP : [s.n.], 2020.

Orientador: Eric Rohmer.  
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade  
de Engenharia Elétrica e de Computação.

1. Membros superiores. 2. Prótese. 3. Visão por computador. 4. Carga de  
trabalho. 5. Sistemas homem-máquina. 6. Interação homem-máquina. 7.  
Android (Recursos eletrônicos). I. Rohmer Eric, 1974-. II. Universidade  
Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação.  
III. Título.

Informações para Biblioteca Digital

**Título em outro idioma:** Development of a human machine interface for the control of upper  
limb prostheses using deep learning

**Palavras-chave em inglês:**

Upper limbs

Prosthesis

Computer vision

Workload

Man-machine systems

Human-machine interaction

Android (Electronic resources)

**Área de concentração:** Engenharia de Computação

**Titulação:** Mestra em Engenharia Elétrica

**Banca examinadora:**

Eric Rohmer [Orientador]

Eduardo Tavares Costa

Gerardo Antonio Idrobo Pizo

**Data de defesa:** 19-10-2020

**Programa de Pós-Graduação:** Engenharia Elétrica

**Identificação e informações acadêmicas do(a) aluno(a)**

- ORCID do autor: <https://orcid.org/0000-0001-8398-302X>

- Currículo Lattes do autor: <http://lattes.cnpq.br/2901503864384483>

## Comissão Julgadora – Dissertação de Mestrado

**Candidato:** Amparo Diaz Munoz   **RA:** 152301

**Data da defesa:** 19 de outubro de 2020

**Título da Tese:** “Desenvolvimento de uma Interface Homem-Máquina para o Controle de Próteses de Membros Superiores usando Deep Learning”

Prof. Dr. Eric Rohmer (Presidente, FEEC/UNICAMP)

Prof. Dr. Gerardo Antonio Idrobo Pizo (UNB)

Prof. Dr. Eduardo Tavares Costa (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de PósGraduação da Faculdade de Engenharia Elétrica e de Computação.



## Dedicatória

Para minha família e amigos.

## Agradecimentos

Devo minha mais profunda gratidão ao meu orientador, o professor Eric Rohmer, pela paciência, confiança, pelas ideias, comentários e todo o esforço para não me deixar desistir. Me sinto muito feliz pelo aprendizado nos últimos anos sobre sua orientação. Também quero agradecer aos meus amigos do laboratório por todos os momentos que compartilhamos. Um agradecimento especial aos meus colegas do grupo de pesquisa Júlio Fajardo e Victor Ferman pelas numerosas recomendações e apoio durante os testes. Tive o apoio, o amor e o incentivo de minha família especialmente meus pais e irmãos, Dora, Berny, Herme e Nolbert por sempre estarem presentes em todas as fases da minha vida. Finalmente quero agradecer a meus grandes amores Leandro e Sarita, minha filha, que são minha felicidade e as pessoas mais importantes na minha vida.

## Resumo

O objetivo desta dissertação é desenvolver e avaliar uma interface homem-máquina denominada UPI (*User-Prosthesis Interface*) para o controle e atuação de próteses de mão. Usando diferentes tecnologias de visão computacional e sinais de eletromiografia, a interface permite minimizar o desgaste físico e emocional de pacientes durante o processo de aprendizagem e adaptação no controle de próteses transradiais. O sistema é implementado usando dispositivos de fácil aquisição comercialmente disponíveis.

Um aplicativo desenvolvido no *Android* recebe a intenção do usuário enviada por uma braçadeira electromiográfica, via três contrações diferentes. O controlador no aplicativo inicializa a previsão do tipo de pegada usando o modelo de aprendizado de máquina incorporado no dispositivo, com a associação de 14 interações predefinidas indexadas às necessidades do usuário, retornando uma ordem para a prótese.

Para fins de teste, se integrou a UPI com uma prótese de membro superior de código aberto, de múltiplas capturas e antropomórfica [1], [2]. A interface também foi avaliada com uma simulação de prótese robótica, desenvolvida em um *framework* chamado V-REP.

Para avaliação da interface, vinte e um voluntários receberam exemplares das tecnologias para adequar a preensão da mão protética à situação vigente. A quantificação dos resultados foi feita com base no sistema de avaliação NASA TLX, uma ferramenta que avalia a quantidade de trabalho despendida para que uma tarefa seja efetuada. Essa avaliação acessa níveis de demanda mental, física, temporal, esforço, desempenho e frustração para calcular a carga de trabalho geral das tarefas.

Uma tabela comparativa enfatiza as vantagens e desvantagens da interface, com outros módulos desenvolvidos em outros projetos para o mesmo propósito em um ambiente instrumentado e não instrumentado.

Os resultados mostram que a interface desenvolvida neste projeto requer menos esforço cognitivo do usuário, seguido pelo fato de que seus usuários não necessitam realizar várias contrações, como ocorre em outros sistemas.

**Palavras-chave:** Próteses de Membro Superior; Visão Computacional; Carga de Trabalho; Esforço Cognitivo.

## Abstract

The objective of this research is to develop and evaluate a Human-Machine interface named UPI (User – Prosthesis Interface) for the hand's prosthesis control and performance. Using different computer vision technologies and electromyography signals, the interface minimizes the physical and emotional stress of the patients during the learning and adaptation process to control transradial prostheses. The system is implemented using easy to purchase and commercially available devices. An application developed on Android receives the user's intention sent by an electromyographic armband, via 3 different muscular contractions. The controller in the application initializes the grip's type prediction using the machine learning model built into the device, with the association of 14 predefined interactions indexed to the user's needs, returning an order to the prosthesis. For testing purposes, the UPI was integrated with an open-source, multiple-capture and anthropomorphic upper limb prosthesis [1], [2]. The interface was also evaluated with a robotic prosthesis simulation, developed in a framework called V-REP. To assess the interface, twenty-one volunteers received copies of the technologies to adapt the prosthetic hand grip to the current situation. The quantification of the results was based on the NASA TLX evaluation system, a tool that evaluates the amount of work expended for a task to be performed. This assessment accesses Mental Demand, Physical Demand, Time Demand, Effort, Performance and Frustration levels to calculate the overall workload of tasks. A comparative table emphasizes the advantages and disadvantages of the interface, with other modules developed in other projects for the same purpose in an instrumented and non-instrumented environment. The results show that the interface developed in this project requires less cognitive effort from the user, followed by the fact that its users do not need to perform several contractions, as occurs in other systems.

Keywords: Upper Limb Prostheses; Computer vision; Work load; Cognitive Effort.

# Lista de Figuras

2.1	Categorias de pegada de mão segundo Shlesinger, 1919. . . . .	18
2.2	Sete (7) principais categorias de garras estáticas: a1) palmar tridigital, a2) palmar dois dedos, b) ponta, c) lateral, d) gancho, e) esfera, f) cilíndrica [21].	18
2.3	Taxonomia de agarre de mão desenvolvida por Cutkosky, fonte: adaptado de [23]. . . . .	19
3.1	Diagrama de blocos - Interação entre os diferentes dispositivos do sistema.	27
3.2	Integração e interação entre os diferentes módulos da UPI. . . . .	27
3.3	Diagrama de blocos dos módulos sugestores de pegadas. (A) Sugestor baseado no reconhecimento do objeto. (B) Sugestor baseado em reconhecimento de pegada. . . . .	28
3.4	Fluxo de trabalho da interface proposta. . . . .	29
3.5	Mão protética antropomórfica Galileo Hand. . . . .	31
3.6	Grupo de pegadas estáticas disponíveis neste projeto. (A) <i>Precision open grip</i> . (B) <i>Precision close grip</i> . (C) <i>Key grip</i> . (D) <i>Column grip</i> . (E) <i>Abduction grip</i> . (F) <i>Finger point</i> . (G) <i>Hook grip</i> . (H) <i>Open palm grip</i> . (I) <i>Pinch grip</i> . (J) <i>Power grip</i> . (K) <i>Relaxed hand</i> . (L) <i>Tripod grip</i> . . . . .	31
3.7	Grupo de pegadas dinâmicas disponíveis neste projeto. (A) <i>Mouse grip: movimento inicial</i> . (B) <i>Mouse grip: sinal de acionamento — clique do mouse</i> . (C) <i>Active index grip: movimento inicial</i> . (D) <i>Active index grip: sinal de disparo — usando um frasco de spray, por exemplo</i> . . . . .	32
3.8	Braçadeira Myo: gestos detectados. . . . .	33
3.9	Braçadeira Myo posicionada no braço. . . . .	34
3.10	Arquitetura com o protótipo utilizando o celular acoplado na prótese. . . .	35
3.11	Arquitetura com o protótipo utilizando uma câmera mais leve acoplada a prótese. . . . .	35
3.12	Diagrama de interação entre o usuário, a interface de leitura de comandos e o controlador. . . . .	36
3.13	Implementação por máquina de estado finito da interface usuário-prótese usando o controlador EMG híbrido. . . . .	38
3.14	Exemplo de sugestão de pegada. (A) usando reconhecimento de objetos. (B) Usando reconhecimento de categorias de pegada. . . . .	40
3.15	funcionamento do módulo sugestor de pegada baseado no reconhecimento de objeto. . . . .	42
3.16	Diagrama de fluxo do módulo sugestor de pegada usando reconhecimento de objetos. . . . .	43
3.17	Arquitetura para o classificador de objetos no celular. . . . .	44
3.18	Diagrama de Classes do banco de dados que contém informações dos objetos, categorias de pegadas e interações dos usuários. . . . .	48

3.19	Visão geral do módulo sugestor de pegada baseado no reconhecimento de pegada. . . . .	49
3.20	Arquitetura de comunicação entre módulo sugestor de pegada baseado no reconhecimento de pegada e o <i>hardware</i> que compõe o sistema. . . . .	50
3.21	Diagrama de fluxo do módulo sugestor de pegada usando reconhecimento de pegada. . . . .	51
3.22	Algumas imagens usadas para treinar a rede. . . . .	52
3.23	Interface de usuário - Configuração. . . . .	55
3.24	Interface de usuário -Funcionamento. . . . .	56
3.25	Exemplo de preenchimento de dados para medir as magnitudes. . . . .	58
3.26	Exemplo de perguntas para avaliação das escalas NASA TLX. . . . .	59
3.27	Exemplo de resultado para avaliação das escalas NASA TLX. . . . .	60
4.1	O erro de entropia cruzada do treinamento e validação definidos em cor azul e vermelho, respetivamente. . . . .	62
4.2	Precisão da classificação de um subconjunto de objetos da vida cotidiana. . . . .	63
4.3	Comparação da precisão de treino e validação de seis classificadores sem a análise. . . . .	64
4.4	Precisão da classificação por pegada para o algoritmo Neural Net. . . . .	65
4.5	Comparação da precisão de treino e validação de seis classificadores após a análise. . . . .	66
4.6	Gráfico de precisão para o modelo treinado com as imagens classificadas por pegada. . . . .	67
4.7	Apertos realizados no teste: (1) <i>Precision grasp</i> . (2) <i>Hook grasp</i> . (3) <i>Lateral grasp</i> . (4) <i>Power grasp</i> . . . . .	69
4.8	UPI usando o módulo sugestor de pegada baseado no reconhecimento de objetos: média das escalas de avaliação. . . . .	71
4.9	Módulo de visão: dados brutos das escalas de avaliação . . . . .	74

# Lista de Tabelas

2.1	Lista de <i>frameworks</i> que permitem a execução em aplicativos móveis. . . .	24
3.1	Lista de materiais usados para construir a UPI. . . . .	29
3.2	Comunicação entre o usuário, a UPI e a prótese. . . . .	39
3.3	Recomendações de estratégia de treinamento, adaptado de [52]. . . . .	46
3.4	Definições das 6 dimensões do método de mensuração NASA-TLX. . . . .	57
4.1	Comparação de resultados de detecção de objetos usando diferentes estruturas e arquiteturas de rede. . . . .	61
4.2	Comparação do modelo sugestor de pegada UPI e outros modelos CNN em termos de precisão, número de parâmetros de modelo, tamanho do modelo, consumo de memória do modelo e tempo de inferência do modelo com conjunto de dados GraspSeg. . . . .	68
4.3	UPI usando o módulo sugestor de pegada baseado no reconhecimento de objetos - magnitude de cada uma das escalas de avaliação. . . . .	70
4.4	Média e Desvio Padrão (DP) da Carga de Trabalho para cada Módulo . .	73

# Sumário

<b>1</b>	<b>Introdução</b>	<b>14</b>
1.1	Justificativa . . . . .	14
1.2	Objetivo . . . . .	15
1.3	Contribuições . . . . .	16
1.4	Organização do Estudo . . . . .	16
<b>2</b>	<b>Revisão de literatura</b>	<b>17</b>
2.1	A mão humana . . . . .	17
2.2	Mãos protéticas . . . . .	19
2.3	Controle das mãos protéticas ativas . . . . .	20
2.3.1	Controle de próteses usando soluções não híbridas . . . . .	20
2.3.2	Controle de próteses usando soluções híbridas . . . . .	21
2.3.3	Controle de próteses usando visão computacional . . . . .	21
2.4	Ferramentas para execução de algoritmos de visão computacional em plata- formas móveis . . . . .	22
2.5	Resumo do capítulo . . . . .	25
<b>3</b>	<b>Metodologia</b>	<b>26</b>
3.1	Arquitetura geral . . . . .	26
3.2	Fluxo de trabalho . . . . .	28
3.3	Lista de materiais . . . . .	29
3.3.1	Smartphone . . . . .	29
3.3.2	Mão protética . . . . .	30
3.3.3	Sensor Eletromiógrafo (EMG) com a braçadeira Myo . . . . .	33
3.3.4	Câmera . . . . .	35
3.4	Interface de leitura de comandos . . . . .	36
3.5	Controlador . . . . .	37
3.6	Interface de prótese . . . . .	38
3.7	Sugestor de pegada . . . . .	40
3.7.1	Sugestor de pegada baseado no reconhecimento de objeto . . . . .	41
3.7.2	Sugestor de pegada baseado no reconhecimento de pegada . . . . .	49
3.8	Interface do usuário . . . . .	54
3.9	<i>Task Load Index</i> - TLX (Índice de Carga de Tarefa em tradução livre) . . .	56
3.10	Resumo do capítulo . . . . .	60
<b>4</b>	<b>Resultados e análises</b>	<b>61</b>
4.1	Resultados . . . . .	61
4.1.1	Resultados para o módulo sugestor de pegada baseado no reconheci- mento de objetos . . . . .	61



4.1.2	Resultados para o módulo sugestor de pegada baseado no reconhecimento de pegada . . . . .	64
4.1.3	Resultados para o teste da Interface UPI integrada com a prótese .	68
4.1.4	Resultados da carga de trabalho usando o método de avaliação NASA TLX . . . . .	69
4.1.5	Resultados Obtidos com os Testes dos Voluntários Hígidos . . . . .	70
4.1.6	Resultados Obtidos com os Testes do Voluntário Amputado . . . . .	73
4.2	Discussão geral . . . . .	74
4.3	Resumo do capítulo . . . . .	75
<b>5</b>	<b>Conclusão</b>	<b>76</b>
<b>A</b>	<b>Aprovação do Comitê de Ética</b>	<b>84</b>
<b>B</b>	<b>Código fonte</b>	<b>89</b>
<b>C</b>	<b>Formulário NASA-TLX</b>	<b>90</b>

# Capítulo 1

## Introdução

### 1.1 Justificativa

Para pessoas que sofreram amputação de membros superiores, existem muitas soluções desenvolvidas para melhorar a qualidade de vida, desde próteses estéticas até próteses feitas em impressoras 3D de alta qualidade que possuem movimentos suaves e precisos tais como [3], [4] e [5].

Entretanto, a principal dificuldade que as mãos protéticas de múltiplas pegadas tem é a maneira como elas interpretam a intenção do usuário. Algumas próteses controlam o movimento dos dedos por meio de um controlador *on-off* ou proporcional baseado apenas no reconhecimento de padrões eletromiográficos (EMG), que apresenta problemas em relação à robustez clínica, como deslocamento de eletrodos, variação de força, posição do membro e alterações transitórias nos sinais [6]. Além disso, o esforço cognitivo necessário e o tempo gasto em treinamento para controlar próteses baseadas em EMG não garantem que os amputados alcancem o controle total do dispositivo. Esse fato, combinado à funcionalidade reduzida das soluções de baixo custo, traz frustração aos usuários e os leva a parar de usar os dispositivos rapidamente [6], [7]. De acordo com os últimos relatórios mundiais sobre deficiências, existe um número significativo de pessoas com amputações que residem nos países em desenvolvimento sem a possibilidade de adquirir assistência protética fornecida por entidades de saúde pública. Sem mencionar a aquisição problemática das principais próteses comerciais dos membros superiores ou mesmo convencionais devido aos seus preços elevados [8], [9]. Assim, as soluções baseadas na tecnologia impressa em 3D estão crescendo, pois abordam problemas de disponibilidade, alto custo, e oferecem um conjunto extenso de captações e gestos [1], [10]–[12]. Para melhor interpretar a intenção do usuário, alguns projetos de pesquisa se concentraram no desenvolvimento de abordagens multimodais para controlar as mãos protéticas dos membros superiores [13]–[15].

Optar por próteses que possuem funcionalidades motoras como as próteses *High-end* também não se mostra uma opção viável nos países não desenvolvidos, uma vez que

essas próteses chegam a custar mais de 60.000 reais (sem considerar custos extras com importação e manutenção do equipamento), um valor que deixa esse tipo de tecnologia inviável de ser utilizada pela grande maioria da população.

Sendo assim, buscando solucionar o problema do desgastante processo de aprendizado para que as próteses mioelétricas sejam controladas, este trabalho apresenta uma interface que utiliza inteligência artificial para reduzir a carga cognitiva, e consequentemente, o tempo de aprendizado para controlar próteses de mão para pessoas que sofreram amputação transradial. A tecnologia existente atualmente permite a construção de próteses avançadas, com dispositivos que permitem replicar uma quantidade considerável de movimentos, suprimindo assim algumas das necessidades motoras de uma pessoa amputada. Porém, a forma como o usuário interage com essas próteses é limitada à interface homem-máquina, sendo este um dos maiores desafios tecnológicos da usabilidade.

A escolha deste tema partiu da necessidade da existência de alternativas que promovam o desenvolvimento de interfaces fáceis de serem controladas por usuários de próteses de mão, inserindo um maior número de pessoas dentro do espectro de atendimento dos sistemas de saúde em países sul-americanos. Dentro deste contexto, é notória a importância social deste projeto, que visa o auxílio às pessoas com mãos amputadas a retomar o controle de funções e movimentos utilizando técnicas não invasivas, com uma curva de aprendizagem mais rápida e com interfaces mais simples de serem usadas e que não requerem esforço cognitivo dos usuários.

Além da questão social, vale destacar o aspecto tecnológico que estará contemplado nas mãos protéticas propostas por este projeto como, por exemplo, reconhecimento de imagens e algoritmos de aprendizagem de máquina.

## 1.2 Objetivo

O objetivo principal deste trabalho é desenvolver e avaliar uma interface que possibilita a interação entre o usuário e as próteses de mão que possuem funcionalidades motoras. Usando técnicas não invasivas, permite a seleção e ativação de padrões de preensão mais usados em tarefas do dia-a-dia, com uma curva de aprendizado mais simples e com menor esforço cognitivo. Para esta finalidade, foram definidos os seguintes objetivos específicos:

- Fazer uma revisão da literatura sobre os diversos tipos de controle das próteses de mão.
- Definir uma arquitetura que permita o controle das próteses de membros superiores com curva de aprendizado mais simples, usando técnicas não invasivas e com baixo custo.

- Implementar a interface definida.
- Integrar e avaliar a interface desenvolvida.

## 1.3 Contribuições

Esta dissertação contribui para a área de engenharia da computação; especificamente, detalha a arquitetura de uma interface desenvolvida em *Android* para o controle das próteses para membros superiores de uma forma simples e com menor esforço cognitivo e sem fazer uso da internet.

Neste trabalho também foi elaborado um algoritmo para analisar banco de dados de imagens, e um dataset que permite classificar padrões de pegada de objetos usados no cotidiano.

Esta dissertação também avalia a interface da perspectiva dos usuários usando o Índice de Carga de Tarefa da NASA-TLX para acessar a carga cognitiva necessária deles durante o uso das interfaces.

## 1.4 Organização do Estudo

A apresentação deste trabalho é feita em 5 capítulos. No capítulo 1 a justificativa para o desenvolvimento da pesquisa e os objetivos foram apresentados. O capítulo 2 apresenta uma revisão da literatura sobre a taxonomia da mão humana, as diversas mãos protéticas existentes e seus diferentes tipos de controle. Algumas ferramentas que permitem a execução de algoritmos usando visão computacional em dispositivos móveis também são descritos.

No capítulo 3 é descrita a metodologia utilizada para o desenvolvimento da interface híbrida para controle das próteses. No capítulo 4 é apresentada a interface do usuário e os resultados obtidos e, finalmente as conclusões deste trabalho são descritas.

O código utilizado neste projeto pode ser encontrado no Apêndice B.

## Capítulo 2

# Revisão de literatura

Este capítulo apresenta uma revisão da taxonomia da mão humana, vários modelos de mãos protéticas desenvolvidos, seguidos da descrição das diferentes formas que essas próteses são controladas no ambiente de pesquisa e na vida real. Finalmente são descritas algumas ferramentas que permitem a execução de algoritmos de visão computacional em aplicativos móveis.

### 2.1 A mão humana

A mão humana pode ser considerada um dos órgãos de grande importância e complexidade para o ser humano. Funções como abrir e fechar a mão são tarefas de considerável complexidade que requerem a contração simultânea de vários músculos individuais. Além disso, a presença de um dedo oponível (o polegar), ou seja, capaz de colocar a ponta em contato com qualquer outro dedo, permitindo ao homem manipular objetos de diferentes tamanhos com eficácia, precisão e força [16], [17].

O conhecimento e caracterização das principais funções motoras da mão humana são fundamentais no desenvolvimento de próteses multifuncionais. Com base nesta caracterização os agarramentos são reproduzidos pelo dispositivo protético [18]. Para descobrir como os humanos escolhem a maneira como os objetos são apreendidos, uma variedade de pesquisas nas áreas de medicina e robótica foram desenvolvidas, permitindo categorizações de apreensão baseadas na forma ou função.

Uma taxonomia é descrita no trabalho de Schlesinger sobre a construção de mãos artificiais [19]. Ele caracterizou quais funcionalidades nas mãos protéticas são necessárias para agarrar certos objetos e dividiu a apreensão humana em *Spherical*, *Cylindrical*, *Palmar*, *Tip*, *Lateral* e *Hook* como é apresentado na figura 2.1.

Com base neste trabalho, Napier analisou os movimentos da mão desde as perspectiva anatômica e funcional, pesquisou os requisitos básicos de tarefas de apertos diferenciando entre dois tipos básicos: o aperto de potência (*Power grip*), que prende um objeto firmemente com uso da palma da mão, e o aperto de precisão (*Precision grip*), onde

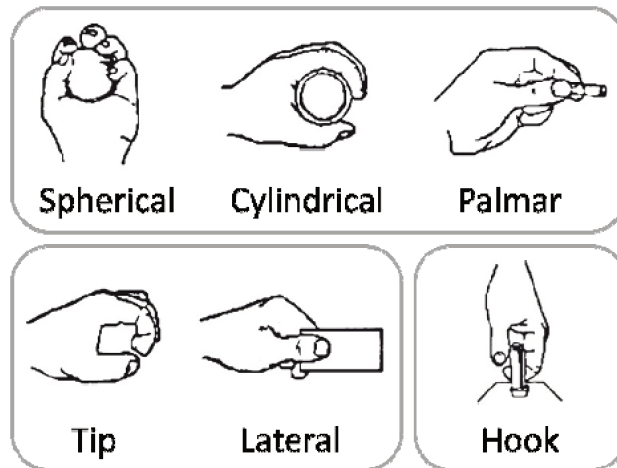


Figura 2.1: Categorias de pegada de mão segundo Shlesinger, 1919.

o polegar e outros dedos beliscam o objeto. Segundo ele, esses dois padrões parecem cobrir toda a gama de atividade preênsil da mão humana [20].

Em 1974 Keller definiu as seguintes sete (7) principais categorias de garras estáticas para uso em dispositivos protéticos: palmar tridigital, palmar dois dedos, ponta, lateral, gancho, esférica e cilíndrica, como é apresentado na figura 2.2.

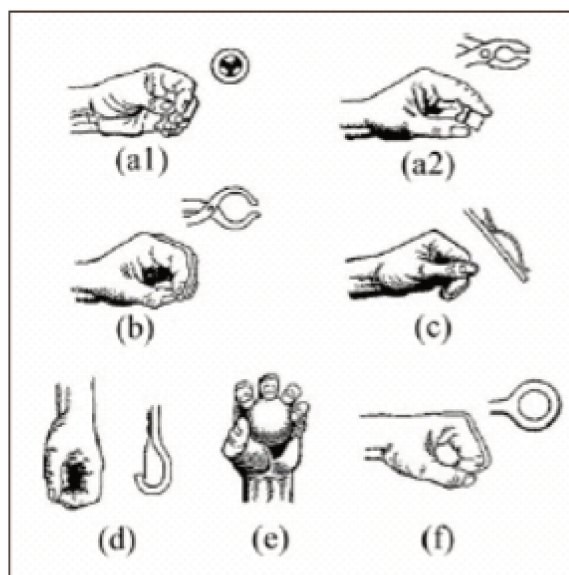


Figura 2.2: Sete (7) principais categorias de garras estáticas: a1) palmar tridigital, a2) palmar dois dedos, b) ponta, c) lateral, d) gancho, e) esfera, f) cilíndrica [21].

Em 1985, o conceito do dedo virtual foi definido como "um grupo de dedos que age como uma única unidade funcional", e sugeriram que a apreensão envolve a oposição de pares de dedos virtuais [22]. De acordo com essa teoria, os dedos virtuais podem ser usados para caracterizar diferentes tipos de apertos de forma abstrata e, fornecer um princípio organizador em um nível mais elevado de planejamento e controle. Em sequência, Iberall (1987) descreve os tipos de pegadas em três configurações relacionadas ao conceito

de dedos virtuais (oposição): o bloco para forças entre as almofadas dos dedos e polegar; palma para forças entre os dedos e a palma; e lado para forças entre o polegar e o lado do dedo indicador. Todas essas posições são independentes e podem ser usadas em uma única tarefa.

Mais tarde, Cutkosky forneceu, em 1989, uma classificação ainda mais detalhada dos movimentos de garra executados pelo homem, baseada nas tarefas de fabricação. A figura 2.3 apresenta a árvore de taxonomia de Cutkosky dividida primeiramente em força e precisão e posteriormente quanto ao formato do objeto [23].

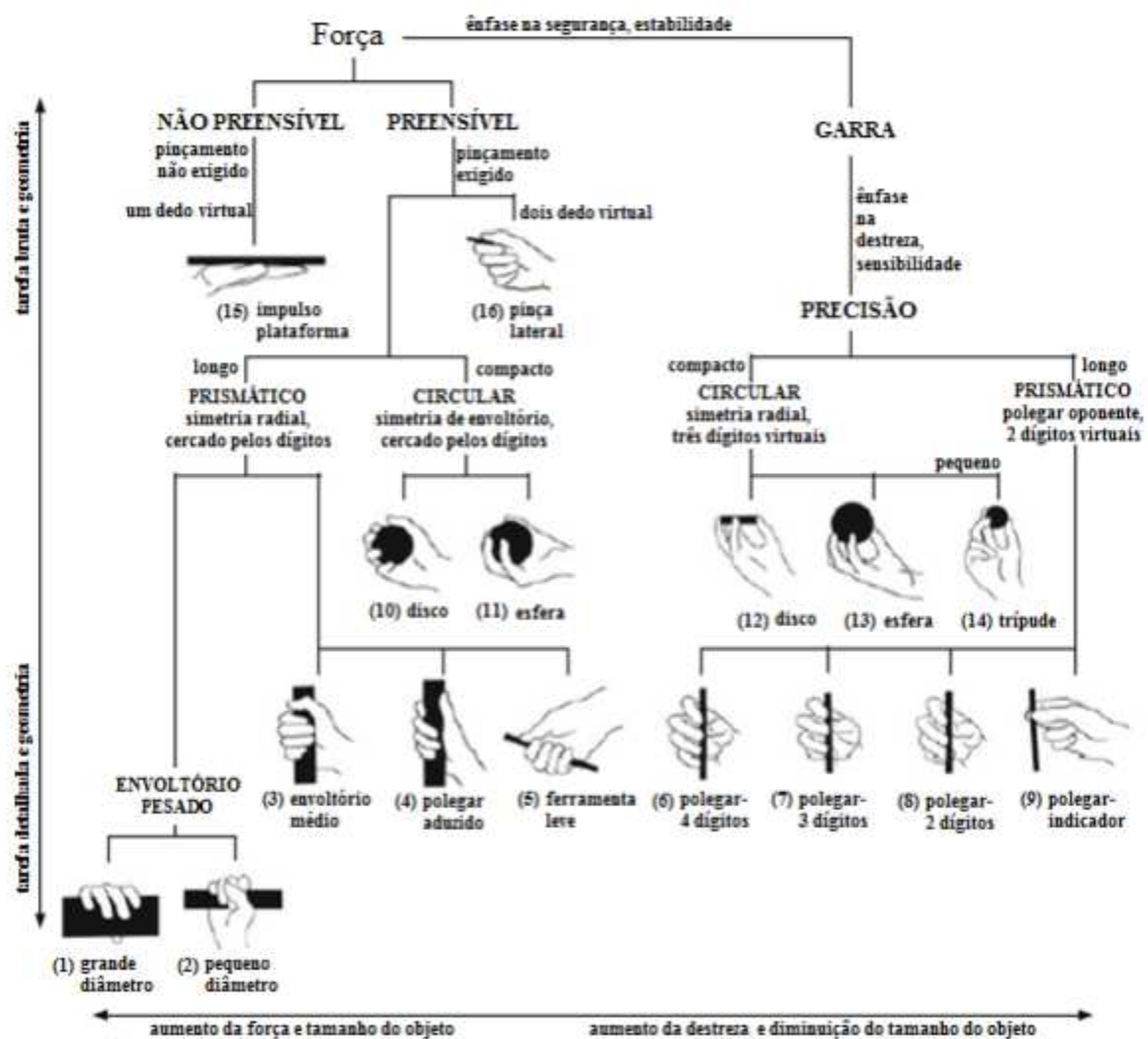


Figura 2.3: Taxonomia de agarre de mão desenvolvida por Cutkosky, fonte: adaptado de [23].

## 2.2 Mãos protéticas

A perda de um membro ou parte dele é um fator que sempre esteve presente na humanidade; além da notável deficiência, o impacto emocional que as pessoas passam após

o acidente e as profundas repercussões sociais após a perda do membro, levam-nas a serem encaradas como diferentes dentro da sociedade em que vivem. Por este motivo o homem desde a antiguidade procurou desenvolver diversas próteses. Define-se como Prótese um dispositivo artificial que tem como objetivo substituir membros ausentes ou má formação congênita.

De acordo com [24], os tipos de próteses podem ser separados em dois grupos: prótese passiva e ativa. As próteses passivas também chamadas de próteses estéticas, são as que não têm movimentos e geralmente são usadas por amputados para fins estéticos. Embora o objetivo principal dessas próteses seja estético, eles têm algumas características funcionais, como empurrar, equilibrar e apoiar. As mãos protéticas ativas têm peças mecânicas ou eletrônicas embutidas nelas e podem ser divididas em dois tipos: próteses alimentadas pelo corpo e alimentadas externamente. Próteses alimentadas pelo corpo são dispositivos que controlam a mão da prótese através dos movimentos do membro residual do amputado ou outras partes dos músculos da parte superior do corpo, como ombros. Próteses alimentadas externamente controlam os movimentos dos dedos usando motores e são alimentadas por baterias.

## **2.3 Controle das mãos protéticas ativas**

Nos últimos anos, muito esforço tem sido feito para reduzir a carga cognitiva exigida pelos amputados para controlar membros superiores protéticos alimentados externamente. Essas estratégias de controle podem ser não híbridas ou híbridas. As soluções não híbridas utilizam apenas um tipo de aquisição para controlar a prótese, podendo ser realizadas mediante sistemas eletromiográficos. As soluções híbridas utilizam uma combinação de sensores de entrada para o controle das mãos protéticas.

### **2.3.1 Controle de próteses usando soluções não híbridas**

Tradicionalmente, a pesquisa no controle de próteses de membros superiores concentrava-se em diferentes técnicas baseadas no pré-processamento de sinais de EMG para analisar a intenção do usuário, com o propósito de ativar a prótese com um perfil de ativação específico. Mãos comerciais típicas usam máquinas de estados ativadas por uma única característica de um subconjunto pré-definido de atividade muscular; já os pesquisadores baseiam-se principalmente no reconhecimento de padrões em uma abordagem multimodal com um conjunto de características EMG combinadas com informações de outro tipo de atividade muscular. Essa abordagem é usada para tratar algumas das questões bem conhecidas das técnicas de EMG, como o efeito da posição do membro, que é resolvido usando unidades de medida inercial com a finalidade de melhorar o processo de classificação associado a essa questão [13], [25]. Por outro lado, algumas abordagens



multimodais implementam uma combinação de características de EMG e mecanomiografia (MMG) captadas por um microfone (mMMG), por um acelerômetro (aMMG) e fibra óptica (FMG), mostrando um aumento na precisão da classificação [26].

### 2.3.2 Controle de próteses usando soluções híbridas

Sistemas híbridos ou multimodais também foram introduzidos para melhorar o controle do usuário de dispositivos protéticos. Como exemplo, um sistema híbrido de EMG e de identificação por radiofrequência (RFID) em objetos específicos para reduzir o esforço cognitivo de operar uma prótese [27]. Da mesma forma, outros sistemas foram testados com o uso do controle híbrido EMG de diferentes maneiras, como abordagens controladas por voz em uma combinação de feedback visual gráfico através de um LCD touchscreen, permitindo que os usuários decidam entre diferentes modalidades para controlar seus dispositivo protético de maneira mais flexível e amigável [15]. Além disso, existem várias abordagens para o uso de interfaces cérebro-máquina (IMC) como meio de controle de próteses de membros superiores. O trabalho mais recente é baseado em eletrocorticografia de alta densidade (ECoG), que permite ao usuário controlar individualmente os dedos naturalmente. Deixando de lado seus excelentes e promissores resultados, o uso de ECoG é um método invasivo e caro que requer o implante no cérebro e uma reinervação muscular direcionada (TMR) em um conjunto específico de músculos. Tanto o implante de ECoG como a TMR são procedimentos desafiadores para a maioria dos amputados [28].

### 2.3.3 Controle de próteses usando visão computacional

Aproveitando o reconhecimento de voz, rastreamento ocular e algumas técnicas de visão computacional, outros estudos foram implementados. No entanto, esses sistemas exigem altos níveis de concentração e treinamento, o que implica um elevado esforço cognitivo para o usuário [29], [30]. Além disso, abordagens de visão computacional também foram propostas para controlar dispositivos protéticos, como um método de aprendizado único, implementado para gerar agarramentos específicos para objetos desconhecidos [31]. Enquanto isso, outro trabalho propôs um controle híbrido empregando óculos de realidade aumentada (AR). Com uma câmera (estéreo) integrada, o sistema pode selecionar automaticamente o tipo de pegada usando técnicas de visão estéreo enquanto os usuários podem ajustar a seleção usando o *feedback* AR, obtendo controle de baixo esforço e resultados significativamente melhores [32]. Para aumentar a funcionalidade da prótese de membro superior com múltiplas pegadas, alguns estudos desenvolveram sistemas híbridos de visão artificial de aprendizado profundo combinados com EMG. Com o objetivo de melhorar a maneira como o sistema interpreta a intenção do usuário, o sistema associa um subconjunto de objetos a um tipo específico de compreensão com base nas propriedades geométricas do objeto. A tarefa de classificação é realizada através de um classificador de

objetos implementado com uma CNN em [33], [34].

Pesquisas recentes em preensão robótica concentraram-se amplamente na realização de detecção de preensão usando aprendizado profundo [35]. Nesse contexto, métodos como [36], [37], focados na previsão de múltiplas pegadas por objeto, são muito úteis especialmente em objetos que podem ser apreendidos de várias maneiras.

Em [2] e [38], foram desenvolvidas e avaliadas três interfaces para o controle e atuação de próteses de mão. As interfaces desenvolvidas combinam sinais de eletromiografia com identificação de radiofrequência, ou técnicas de visão computacional para selecionar os tipos de interação. Os sinais EMG são responsáveis por desencadear o sistema, enquanto os outros sensores são responsáveis pela seleção da preensão para que o usuário possa interagir com o ambiente; as interações do usuário com a prótese são vistas em um *software* de simulação. São os primeiros trabalhos que tiveram a ideia de usar o reconhecimento de objetos para definir o tipo de interação (não só pegada, mas interação não necessariamente física) com eles. Os outros trabalhos usavam visão computacional focada, com a definição da pegada baseada unicamente no formato do objeto.

Não se encontrou na literatura um sistema de controle de próteses de mão que use o *smartphone* como o dispositivo central para a sugestão do tipo de pegada usando aprendizado profundo. Em todos os casos descritos anteriormente foi utilizado um computador ou uma computador de placa única tipo *raspberrypi*.

## 2.4 Ferramentas para execução de algoritmos de visão computacional em plataformas móveis

Existem uma grande variedade de *frameworks* de *software* que fornecem uma base necessária e contêm diferentes conjuntos de elementos, como camadas convolucionais para implementar redes neurais profundas sem codificar do zero. Eles também fornecem toda a infraestrutura necessária para implementar funções, como ler um arquivo de descrição de rede, vincular funções principais a uma rede, ler dados de bancos de dados de treinamento e validação, etc.

Vários *frameworks* apareceram nos últimos cinco anos para treinar redes neurais profundas. Alguns exemplos incluem Caffe, Cognitive Toolkit da Microsoft, Darknet, MXNet, TensorFlow, Theano e Torch. Além disso, vários fornecedores de *chips* fornecem ferramentas proprietárias para quantificar e otimizar redes para aplicativos incorporados com recursos limitados. Às vezes, essas ferramentas são integradas a uma estrutura autônoma; outras vezes, elas exigem ou incluem uma versão personalizada de outra estrutura existente.

Embora algumas das estruturas acima incluam uma opção específica para execução em plataformas móveis, seu objetivo fundamental é projetar e treinar redes neurais

profundas em GPUs poderosas, onde faz todo o sentido treinar uma rede profunda, em vez de realizar inferência em dispositivos móveis com recursos limitados. Em resposta a esse problema, várias estruturas alternativas foram desenvolvidas para adaptar o design e o treinamento de Deep learning aos requisitos de dispositivos móveis.

O *DeepLearningKit* [39] é uma estrutura de código aberto que suporta o uso de modelos pré-treinados de aprendizagem profunda (redes neurais convolucionais) para iOS, OS X e tvOS. *DeepLearningKit* é desenvolvido para usar a GPU de forma eficiente e para a integração com aplicativos, por exemplo, aplicativos móveis baseados em iOS no iPhone/ iPad. O objetivo dos autores é oferecer suporte a modelos de aprendizado profundo previamente treinados com estruturas populares como Caffe, TensorFlow, Torch e Theano.

O CNNdroid [40] é uma biblioteca acelerada por GPU de código aberto, que foi projetada e otimizada especificamente para a execução de redes neural convolucional profundas (CNN) treinadas em dispositivos móveis baseados no *Android*. Ela suporta quase todos os tipos de camada CNN e é compatível com os modelos de rede treinados por Caffe, Torch e Theano. Além disso, ele pode usar os recursos de computação de GPU e CPU do celular.

Deepsense [41] é uma estrutura de aprendizado profundo, particularmente robusta para filtrar o ruído de sensores móveis. O Deepsense foi especialmente projetado para tratar dados de séries temporais e é capaz de executar modelos para diferentes aplicações, como rastreamento de carros, detecção de objetos, reconhecimento de imagens e reconhecimento de faces em tempo real.

O Boda-RTC [42] é um sistema de código aberto que permite desenvolver rapidamente novos *kernels* computacionais para destinos de *hardware* existentes. A base deste sistema é usar uma abordagem de geração de código para direcionar as plataformas OpenCL neutras ao fornecedor.

Para a implementação de redes neurais em um dispositivo móvel Tensorflow atualmente existem duas soluções:

TensorFlow Mobile: O TensorFlow foi desenvolvido desde o início para ser uma boa solução de aprendizado profundo para plataformas móveis como *Android* e iOS. O TensorFlow Mobile representa a versão móvel da estrutura que você pode usar em seus aplicativos móveis. Ele também contém vários guias e scripts para a implantação de um modelo em um aplicativo móvel.

TensorFlow Lite: [43] é a solução oficial do TensorFlow para executar modelos de aprendizado de máquina em dispositivos móveis e incorporados. Ele permite a inferência de aprendizado de máquina no dispositivo com baixa latência e um pequeno tamanho binário no *Android*, iOS e outros sistemas operacionais. Permite aos desenvolvedores executarem modelos aprendidos por máquina em dispositivos móveis com baixa latência, para que possam tirar proveito deles para que faça classificação, regressão ou qualquer

outra coisa que possam desejar sem necessariamente incorrer em uma ida e volta para um servidor. Atualmente, ele é suportado no *Android* e iOS por meio de uma API C++, além de possuir um Java Wrapper para desenvolvedores do *Android*. Além disso, nos dispositivos *Android* compatíveis, o intérprete também pode usar a API de redes neurais do *Android* para acelerar o *hardware*, caso contrário, o padrão será a CPU para execução.

Com a popularidade do aprendizado de máquina e aplicativos móveis, a Apple lançou sua biblioteca Core ML, que permite aos desenvolvedores de aplicativos móveis treinar modelos em computadores poderosos; depois salvar os modelos de treinamento no telefone e executar sua versão otimizada no local [44].

O HG-Caffe fornece aceleração de até 20 vezes as GPUs em comparação com as implementações originais. Além da aceleração, o pico de uso da memória também é reduzido para cerca de 80% com a HG-Caffe [45].

O MXNet é uma biblioteca de aprendizado de máquina que oferece diferenciação automática para derivar gradientes e eficiente em computação e memória. -Executado em vários sistemas heterogêneos, variando de dispositivos móveis a clusters de GPU distribuídos [46].

A tabela 2.1 apresenta um resumo de algumas ferramentas existentes na internet para a execução/visão computacional nos dispositivos móveis.

Tabela 2.1: Lista de *frameworks* que permitem a execução em aplicativos móveis.

Biblioteca	Plataforma	GPU	Modelos suportados	Arquiteturas suportadas
Tensorflow	iOS/Android	sim	Tensorflow	CNN, RNN, LSTM, outras
CnnDroid	Android	sim	Caffe, Torch, Theano	CNN
DeeplearnigKit	iOS	sim	Caffe, TensorFlow, Torch e Theano	CNN
MXNet	iOS/Android	não	MXnet	CNN, RNN, LSTM, outras
Caffe	iOS/Android	não	Caffe	CNN
Core ML	iOS	sim	TensorFlow, Torch e Theano	CNN

## 2.5 Resumo do capítulo

Este capítulo apresentou alguns dos trabalhos mais importantes sobre taxonomia para mãos humanas; uma visão geral de diferentes tipos de mãos protéticas e as estratégias de controle em próteses com alimentação externa encontradas na literatura, utilizando técnicas híbridas e não híbridas com RFID, reconhecimento de voz, visão computacional e outros tipos de tecnologias.

## Capítulo 3

# Metodologia

Devido às limitações dos perfis de ativação tradicionais para próteses de membros superiores, este projeto implementou uma interface denominada UPI que permite ao usuário o controle de uma prótese de mão. A interface desenvolvida é a continuação de uma metodologia híbrida baseada em um trabalho anterior testado e validado com uma simulação protética usando a estrutura robótica V-REP [2], [38], [47].

A seção 3.1 descreve a arquitetura proposta. O fluxo do trabalho é apresentado no item 3.2, seguido da lista dos materiais na seção 3.3.

A UPI está dividida em quatro partes específicas: a interface de leitura de comandos, o módulo reconhecimento de pegada, o controlador e interface prótese, os quais são apresentados nas seções 3.4, 3.5 e 3.6 respectivamente.

A UPI implementou um módulo chamado sugestor de pegada baseado em visão computacional para definir o tipo de pegada mais apropriado para um objeto com o qual o usuário quer interagir. Esse módulo é desenvolvido de duas maneiras diferentes explicadas nas seções 3.7.1 e 3.7.2.

### 3.1 Arquitetura geral

A figura 3.1 apresenta o diagrama de comunicação geral da interface proposta. A UPI é uma interface híbrida para o controle de uma prótese de mão e baseia-se na interação de quatro dispositivos diferentes: um smartphone, uma webcam, a braçadeira Myo e uma prótese de mão.

Um aplicativo desenvolvido no *Android* recebe os sinais do usuário enviados pela braçadeira Myo e os redireciona para o controlador para inicializar a previsão do tipo de pegada usando um modelo de aprendizado de máquina incorporado no dispositivo. O controlador interpreta a intenção do usuário e permite que ele controle um dispositivo protético de maneira fácil, permitindo a associação de um total de 14 interações predefinidas indexadas às necessidades do usuário. O sistema é implementado usando dispositivos de fácil aquisição e comercialmente disponíveis.



Figura 3.1: Diagrama de blocos - Interação entre os diferentes dispositivos do sistema.

A interação do usuário, a UPI e a prótese é apresentado na figura 3.2. A UPI é subdividida em módulos. O módulo interface de leitura de comandos descrito na seção 3.4 é encarregado de classificar as contrações feitas pelo usuário em três comandos: um comando para tirar uma foto, outro que pode ser usado para aceitar a sugestão de pegada do objeto, e um terceiro para cancelar a operação ou para recusar uma sugestão de interação. O controlador implementa uma máquina de estados que permite a interação entre todos os dispositivos; este módulo é apresentado na Seção 3.5. O módulo interface com a prótese codifica os comandos no formato *json* e os envia para a prótese através de *Bluetooth*. Este módulo é apresentado na seção 3.6.

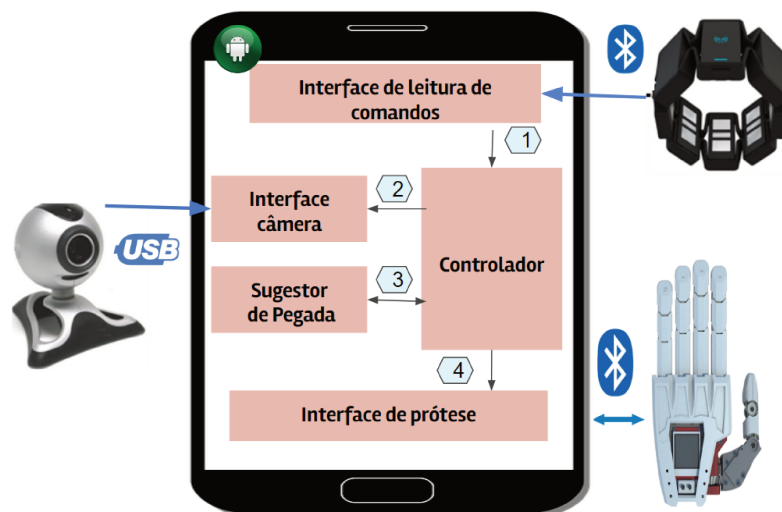


Figura 3.2: Integração e interação entre os diferentes módulos da UPI.

O módulo sugestor de pegada permite fazer a previsão do tipo de pegada de duas maneiras diferentes. O primeiro módulo de previsão do tipo de pegada é apresentado na figura 3.3 (A). Este módulo desenvolve uma proposta de interações baseada em um dicionário e usa um modelo de aprendizado de máquina baseado na classificação de objetos. Com a imagem capturada, o controlador reconhece o nome do objeto que o usuário quer interagir e realiza uma busca em um banco de dados para saber qual tipo de preensão é a mais provável de ser utilizada pelo usuário para interagir com aquele objeto. Os detalhes deste módulo são descritos na seção 3.7.1.

O segundo módulo de previsão do tipo de pegada explicado na seção 3.7.2

e apresentado na figura 3.3 (B) usa técnicas de visão computacional para identificar diretamente a forma do objeto e fazer a sugestão da pegada sem uso do dicionário. Uma base de imagens com onze (11) categorias de preensão foi criada e utilizada para que a UPI faça a sugestão automaticamente do tipo de preensão sem consultar no banco de dados.

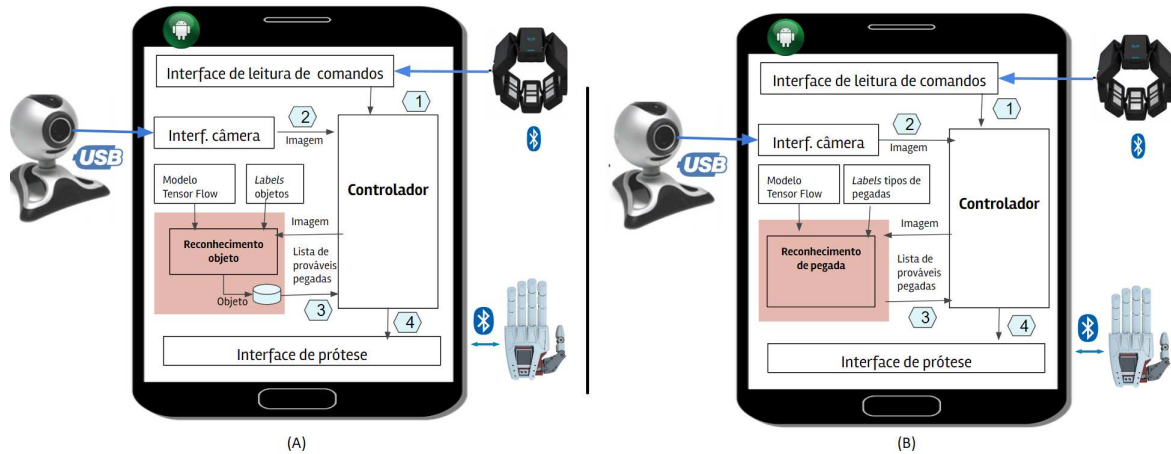


Figura 3.3: Diagrama de blocos dos módulos sugestores de pegadas. (A) Sugestor baseado no reconhecimento do objeto. (B) Sugestor baseado em reconhecimento de pegada.

## 3.2 Fluxo de trabalho

A figura 3.4 ilustra como a interface proposta funciona. O usuário envia um comando (uma contração) para que o sistema capture a imagem de um objeto que ele queira interagir. A imagem do objeto é então processada pelo sistema através de um algoritmo de reconhecimento de imagem para saber qual tipo de preensão é a provável de ser utilizada pelo usuário; esta preensão é então sugerida para o usuário através de um *feedback* visual e/ou vocal com o nome da interação.

Caso o usuário queira aceitar a preensão sugerida pelo sistema, ele pode realizar uma contração de confirmação para a prótese efetuar o movimento correspondente. Ao término da interação com o objeto, o usuário pode enviar uma contração de cancelamento para que a prótese retorne ao estado inicial de repouso.

Se o usuário não aceitar a primeira sugestão dada pelo sistema, ele poderá realizar uma contração de cancelamento e o sistema irá buscar a próxima contração provável até que encontre o tipo de interação correta.



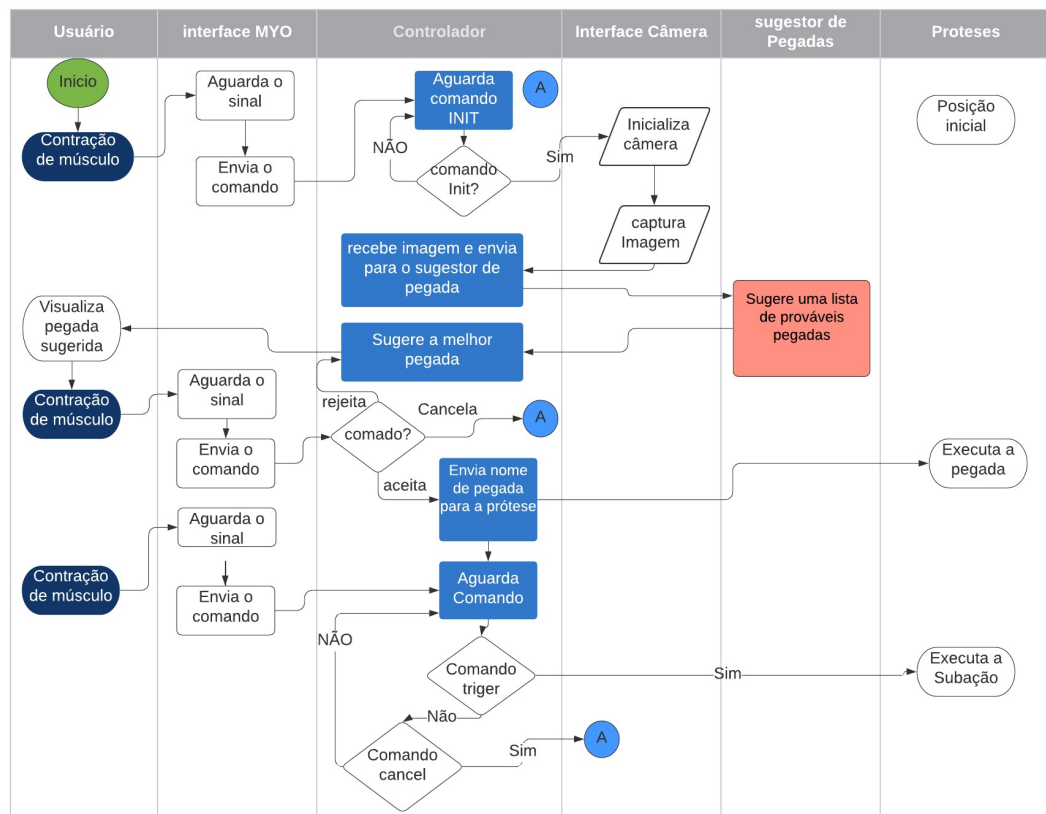


Figura 3.4: Fluxo de trabalho da interface proposta.

### 3.3 Lista de materiais

Os materiais listados na Tabela 3.1 foram utilizados para construir a interface proposta.

Material	Quantidade
Myo armband	1
Câmera usb	1
Smartphone com Android	1
Mão protética	1

Tabela 3.1: Lista de materiais usados para construir a UPI.

#### 3.3.1 Smartphone

O *smartphone* é uma plataforma compatível e de pequeno porte, com unidade de processamento de alta potência e sistema operacional sofisticado. Além disso, ele contém uma tela sensível ao toque grande o suficiente que pode ser utilizada para a interface gráfica pelo usuário neste projeto. Também contém um sistema de áudio que pode ser usado para comunicar e instruir o paciente audivelmente; também possui memórias internas e

externas para processamento computacional e armazenamento de dados. Todos os itens acima mencionados tornam o *smartphone* uma excelente opção para o nosso projeto.

O *smartphone* com *Android* atua como um dispositivo central que interage com os outros dispositivos de forma transparente, utilizando a contração muscular como a transição para navegar pela máquina de estado, para tirar uma foto e iniciar, invalidar ou cancelar a sugestão de interação proposta.

### 3.3.2 Mão protética

Inicialmente foi usado um simulador chamado V-REP, para simular o comportamento da mão protética e fornecer *feedback* visual sobre a interação humana da máquina [47]. A simulação (implementada em V-REP) recebe do controlador da interface o comando e posiciona a mão na pegada desejada.

Próteses impressas que implementem o protocolo de comunicação definido pela UPI podem usar a interface desenvolvida e sua aplicação móvel. Para testar a UPI proposta com uma prótese real, utilizamos o Galileo Hand, que é uma prótese de membro superior de código aberto e antropomórfica que foi desenvolvida pela Unicamp em parceria com a Universidade da Guatemala [1], [2].

A prótese Galileo Hand foi ligeiramente modificada. A adição de um módulo *Bluetooth* v3.0 foi realizada para estabelecer a comunicação *full-duplex* entre os dispositivos pelo uso de mensagens no formato *json*. Assim, os processos implementados no *smartphone* e no sistema embarcado são executados simultaneamente, permitindo modularidade e distribuição da carga computacional na UPI. Dessa forma, o controlador incorporado pode gerenciar e executar comandos recebidos do *smartphone* de maneira fácil e transparente. Além disso, um pequeno laser de baixa potência foi colocado estrategicamente visando a escolha de um quadro adequado para fotografar o objeto com o qual o usuário deseja interagir. Um módulo LCD inteligente (tela TFT LCD de 1,44") permite *feedback* visual ao usuário, mostrando textos e animações de algumas sugestões sugeridas, como mostra a figura. 3.5.

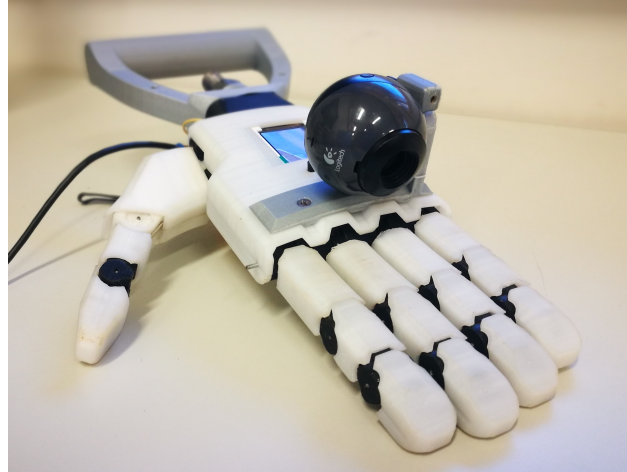


Figura 3.5: Mão protética antropomórfica Galileo Hand.

Este projeto permite trabalhar tipos comuns de agarre com base na taxonomia de prensão de Cutkosky [48] e nos projetos de próteses propostos a partir da prótese BeBionic [49]. A UPI lida com catorze categorias de pegadas, divididas em dois grupos: pegadas dinâmicas e pegadas estáticas.

As pegadas estáticas ilustradas na figura 3.6 são os movimentos iniciais da prótese que permitem segurar o objeto:

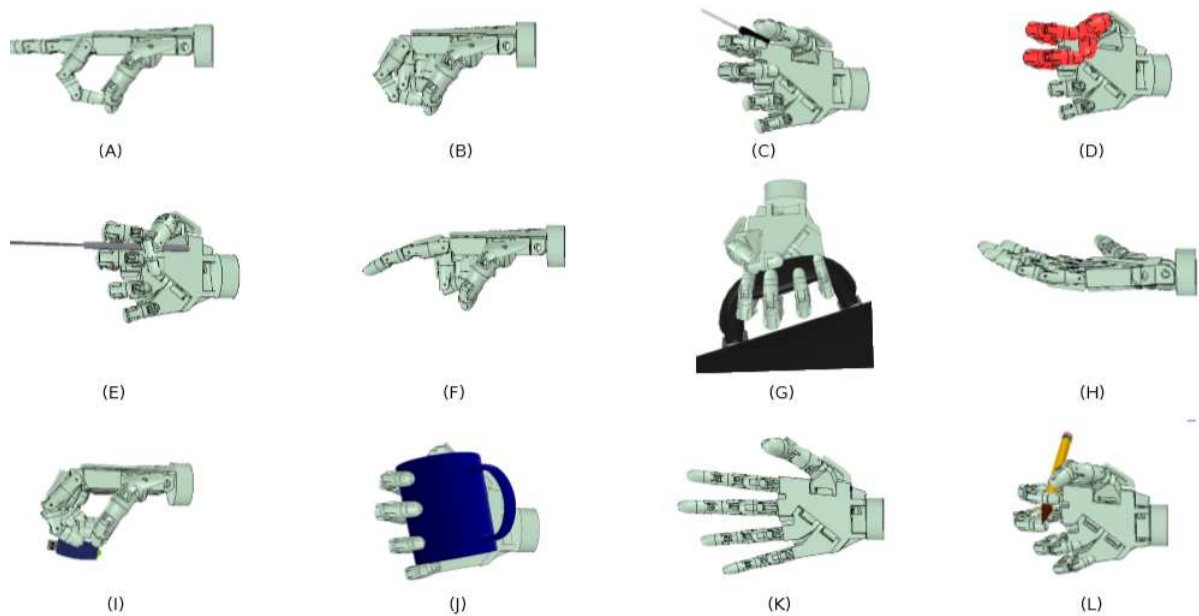


Figura 3.6: Grupo de pegadas estáticas disponíveis neste projeto. (A) *Precision open grip*. (B) *Precision close grip*. (C) *Key grip*. (D) *Column grip*. (E) *Abduction grip*. (F) *Finger point*. (G) *Hook grip*. (H) *Open palm grip*. (I) *Pinch grip*. (J) *Power grip*. (K) *Relaxed hand*. (L) *Tripod grip*.

As pegadas dinâmicas são uma sequência de ações necessárias para interagir com o objeto ou o ambiente. A figura 3.7 ilustra as pegadas dinâmicas usadas neste projeto.

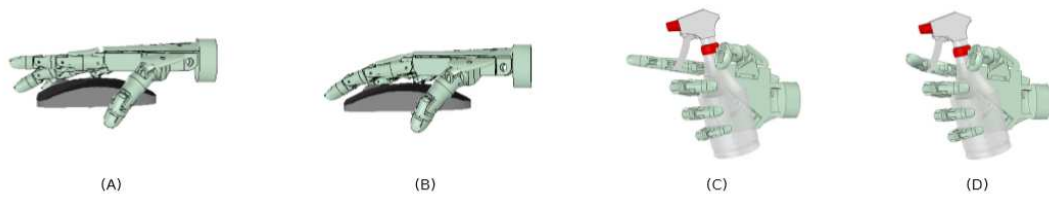


Figura 3.7: Grupo de pegadas dinâmicas disponíveis neste projeto. (A) *Mouse grip*: movimento inicial. (B) *Mouse grip*: sinal de acionamento — clique do mouse. (C) *Active index grip*: movimento inicial. (D) *Active index grip*: sinal de disparo — usando um frasco de spray, por exemplo.

Segue a descrição das pegadas estáticas e dinâmicas:

1. ***Relaxed hand***: ajuda a obter uma posição natural. Figura 3.6 (K).
2. ***Active index grip***: oferece a posição ideal do dedo para digitar ou operar um frasco de spray. Permite controlar o dedo indicador independentemente e posicioná-lo de acordo. Figura 3.7 (C, D).
3. ***Column grip***: permite empurrar objetos mais pesados ou maiores, botões e interruptores. Figura 3.6 (D).
4. ***Abduction grip***: permite segurar objetos finos com mais segurança, como talheres ou escova de dentes, entre os dedos. Figura 3.6 (E).
5. ***Finger point***: a ponta do dedo permite digitar em um teclado e pressionar uma campainha ou botão. Figura 3.6 (F).
6. ***Hook grip***: é ideal para transportar uma sacola de compras ou uma mala. Figura 3.6 (G).
7. ***Key grip***: permite carregar um pedaço de papel ou cartão, usar uma colher ou segurar objetos finos, como um prato, um cartão de crédito ou uma chave. Na posição não oposta do polegar, os quatro dedos se fecham parcialmente. O polegar então se fecha para o lado do dedo indicador. Figura 3.6 (C).
8. ***Mouse Grip***: permite operar um mouse. Figura 3.7 (C, D).
9. ***Open palm grip***: é adequada para transportar uma bandeja ou um prato. Figura 3.7 (H).
10. ***Pinch grip***: pode ser aplicado especificamente à manipulação fina de objetos como uma unidade de flash, por exemplo. Figura 3.7 (I).
11. ***Power grip***: esse padrão permite segurar objetos redondos ou cilíndricos com mais facilidade e segurança; permite segurar uma bola ou um pedaço de fruta, além de

garrafas ou alças de casa e jardim utensílios. O aperto de força também permite que você agite as mãos. Figura 3.7 (J).

12. ***Precision close grip***: para brincar com um cubo mágico. Figura 3.7 (B).
13. ***Precision open grip***: permite pegar e manipular objetos pequenos quando o polegar está em oposição. Figura 3.7 (A).
14. ***Tripod grip***: permite segurar e manipular uma variedade de objetos do dia a dia, como chaves de carro, moedas, tampas de garrafas e canetas. figura 3.7 (L).

### 3.3.3 Sensor Eletromiográfico (EMG) com a braçadeira Myo

A braçadeira Myo [50] é necessária para adquirir os sinais EMG que são usados como uma das entradas para controlar a interface híbrida desenvolvida. A braçadeira Myo é desenvolvida pela Thalmic Labs Inc. e possui oito sensores EMG.

A braçadeira Myo é mostrada na figura 3.8. É composta por *leds* de indicador duplo que informam o nível da bateria e conexão Bluetooth, um dispositivo eletrônico de nove eixos (acelerômetro de três eixos, giroscópio de três eixos e magnetômetro de três eixos) e 8 sensores eletromiográficos. A bateria de íons de lítio recarregável do braço Myo pode durar um dia inteiro com uma carga. Para recarregar, é preciso apenas conectar um cabo micro-USB padrão no dispositivo.



Figura 3.8: Braçadeira Myo: gestos detectados.

O usuário usa esta braçadeira no membro residual (para amputados) ou no antebraço (para voluntários saudáveis).



Figura 3.9: Braçadeira Myo posicionada no braço.

### **Características técnicas**

O Armband Myo tem seu comprimento ajustável entre 19 cm e 34 cm, de modo que possa melhor se adaptar ao braço de cada utilizador. A massa do equipamento é de 93 g e sua espessura é de 1,1 cm. O equipamento é compatível com sistemas operacionais da Microsoft e da Apple para computadores, Para celulares existe compatibilidade tanto para sistemas IOS, quanto para sistemas Android. A comunicação entre os aparelhos é feita pelo protocolo de comunicação Bluetooth, existindo um adaptador para os computadores utilizando uma porta USB para a recepção e transmissão de dados. Com relação às características de Hardware, são dois tipos de sensores, oito do tipo de eletromiografia parcial, capazes de informar impulsos elétricos dos músculos do braço, reconhecendo movimentos e poses; além de sensores de medidas inerciais, com base em nove eixos: três eixos de giroscópio, três eixos de acelerômetro e três eixos magnetrônicos. O equipamento contém LEDs para informar o estado do equipamento, como ligado, desligado, sincronizando e em espera. Além dos LEDs, existem três tipos de vibração que o equipamento é capaz de produzir uma curta, uma média e uma longa, que também informam ao usuário o estado do equipamento. O processador utilizado é um ARM Cortex M4 e possui uma bateria de Lítio, capaz de durar 24 horas de uso e, para carregá-la, existe uma porta micro USB.

### **Funcionamento**

Como a braçadeira MYO é um sensor que possibilita o reconhecimento de gestos e movimentos de qualquer pessoa, existem parâmetros que são configurados com auxílio do computador, antes do primeiro uso de cada usuário, deve ser feita uma configuração, é instalado o software inicial presente no site do Armband Myo, o myoconnect. Com este software, são dadas as informações básicas ao usuário do passo a passo de configuração, em que o usuário realiza uma série de movimentos e gestos, que calibram o dispositivo, para que depois seja salvo em um arquivo as configurações daquele usuário. Após a calibração, o usuário pode colocar o bracelete em qualquer um dos dois braços, logo abaixo do cotovelo,

fazer um gesto de reconhecimento, e o dispositivo vai vibrar para informar que está pronto para uso.

## Integração com o Android

Para a integração da braçadeira MYO com o aplicativo no android se baixou o SDK MyO do site e se colocou na pasta do projeto criado no Android Studio, a versão usada do Compile SDK é API 23 Android 6.0 (Marshmallows), versão da ferramenta de compilação é 23.0.2, SDK de destino mínimo é 18. e se usou um Galaxy s3 para testes, depois de adicionadas às dependências, se implementou um método que permite receber a informação e responder com base nos sinais que recebidos.

### 3.3.4 Câmera

A figura 3.10 apresenta o protótipo com o uso da câmera própria do celular, onde o *smartphone* pode ser encaixado na mão da prótese. O ponto da câmera frontal do *smartphone* aponta em direção à frente da mão com um espelho ou uso de uma câmera USB.

O *smartphone* acoplado à prótese pode ser muito pesado para o uso a longo prazo, criando dor nas costas do usuário. Por este motivo, foi desenvolvida outra versão que faz uso de uma câmera, menor e mais leve, conectada ao controlador de prótese a qual envia a foto para o *smartphone* mediante solicitação via *Bluetooth*. O *smartphone* processa a imagem, sugere e ordena as ações exatamente como a versão anterior figura 3.11.



Figura 3.10: Arquitetura com o protótipo utilizando o celular acoplado na prótese.



Figura 3.11: Arquitetura com o protótipo utilizando uma câmera mais leve acoplada a prótese.



### 3.4 Interface de leitura de comandos

Os sensores EMG existentes no Myo registram a atividade elétrica produzida pelos músculos dos usuários e enviam os dados para o aplicativo em *Android* através de *Bluetooth*.

O aplicativo possui uma interface que faz uso do Kit de Desenvolvimento de Software para Android (SDK) fornecido pela Thalmic Labs para acessar as funções do Myo. Esta interface recebe uma notificação através de uma comunicação *Bluetooth* e envia ao controlador qual contração foi executada. Sempre que a interface detecta uma contração, ela envia um sinal para o controlador que, dependendo do tipo de contração, pode iniciar, cancelar, aceitar ou rejeitar uma ação.

O conjunto de movimentos para controlar o dispositivo protético é ilustrado pela figura 3.12.

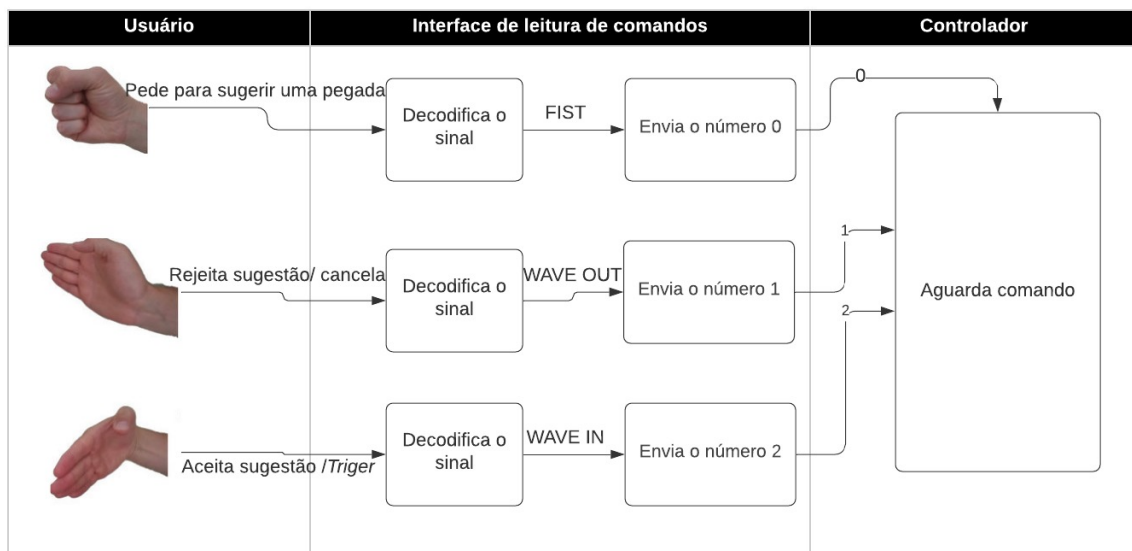


Figura 3.12: Diagrama de interação entre o usuário, a interface de leitura de comandos e o controlador.



### 3.5 Controlador

Para permitir a interação entre todos os dispositivos mostrados na figura 3.1, o controlador implementa uma máquina de estados usando a biblioteca TinyMachine [51]. No estado inicial  $S_0$ , a prótese permanece em repouso natural enquanto o UPI permanece no estado inicial até que o usuário aponte a câmera montada para o objeto específico com o qual deseja interagir. Em seguida, o usuário executa a contração  $q_0$  para acionar uma transição para o estado  $S_1$ . Nesse estado, o sistema fotografa o objeto e envia a imagem para o módulo sugestor de pegadas, o qual retorna uma lista ordenada por prioridades das possíveis pegadas para o objeto. O controlador apresenta ao usuário o tipo da pegada com mais probabilidade. A validade de um rótulo é obtida com a certeza de classificação que consequentemente desencadeia a transição para mudar para o estado  $S_2$ . Se o processo não retornar um rótulo válido, o UPI retornará ao estado  $S_0$ , após um tempo limite predefinido ( $t$ ). Deste modo, no estado  $S_2$  o resultado da classificação é enviado ao *smartphone*, um áudio com o nome do objeto, enquanto uma animação associada ao tipo sugerido de compreensão é apresentada em um LCD montado na tela da prótese. O usuário pode aceitar a sugestão executando a contração  $q_2$  e disparar a transição para o estado ou estado  $S_3$ , ou rejeitá-lo executando a contração  $q_3$ . Neste caso o sistema fica em estado  $S_2$  e sugere o próximo aperto mais apropriado, desta forma o UPI se adapta na escala de atividades da vida diária de forma customizada apresentando flexibilidade ao usuário, devido às sucessivas propostas do sistema que modifica os valores de probabilidade no dicionário. Caso um objeto no dicionário nunca tenha sido detectado, toda a sugestão de interação é equiprovável e a sequência de sugestões será exibida por seu crescente índice de interação. Então, no estado  $S_3$  a prótese executa a pegada aceita e a libera executando a contração  $q_1$ . Em casos especiais, como as contrações "mouse grip" ou "index grip", uma ação secundária será ativada ao executar a contração  $q_3$ . O diagrama de máquina de estados finitos da implementação é mostrado na figura 3.13. Um vídeo demonstrativo pode ser visualizado em <https://youtu.be/xP2cQRMUm0A>.

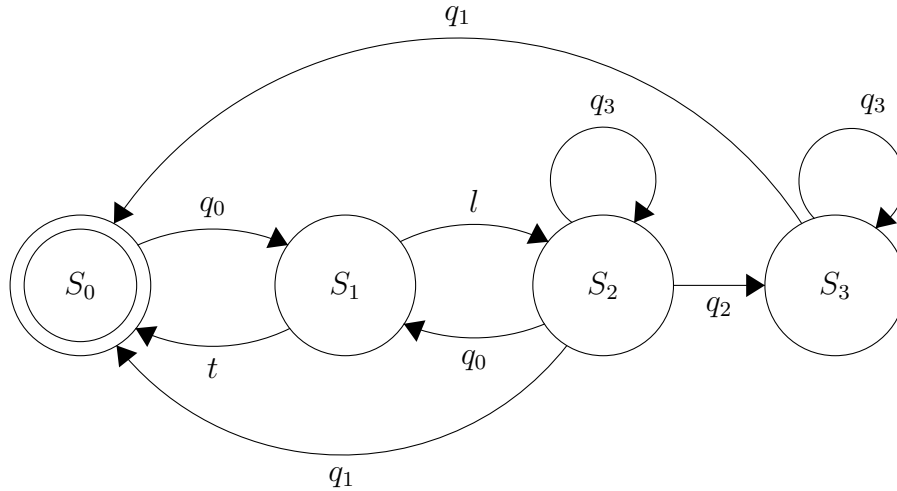


Figura 3.13: Implementação por máquina de estado finito da interface usuário-prótese usando o controlador EMG híbrido.

### 3.6 Interface de prótese

Para estabelecer a comunicação entre a UPI e as próteses foi desenvolvido um protocolo de comunicação *full-duplex Bluetooth*, onde as mensagens são enviadas no formato *json*. Assim, os processos implementados no *smartphone* e no sistema embarcado são executados simultaneamente, o controlador incorporado pode gerenciar e executar comandos recebidos do *smartphone* de maneira fácil e transparente. As funções básicas do módulo interface próteses são as seguintes:

- Se conectar com a prótese mediante o protocolo *Bluetooth*.
- Enviar informação do estado da interface para ser visualizada no LCD da prótese.
- Enviar o valor do tipo de pegada sugerido para ser visualizado pelo usuário.
- Enviar os comandos de aceitação do tipo de pegada para ser executado na prótese.

A tabela 3.2 apresenta as ações necessárias para a comunicação entre o usuário, a UPI e a prótese.

Tabela 3.2: Comunicação entre o usuário, a UPI e a prótese.

Usuário	Estado	Controlador/Interface de próteses	Ação próteses
	S0	Envia os seguintes dados {"st":0}	Permanece em repouso natural
Inicia a intenção de pegar o objeto	S1	Envia os seguintes dados {"st":1}	Aguarda os dados da pegada sugerida
	S2	Sugere o tipo de pegada Active index grip e envia os seguintes dados {"st":2,"g":1}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Column grip e envia os seguintes dados {"st":2,"g":2}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Abduction grip e envia os seguintes dados {"st":2,"g":3}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Finger point e envia os seguintes dados {"st":2,"g":4}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Hook grip envia os seguintes dados {"st":2,"g":5}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Key grip envia os seguintes dados {"st":2,"g":6}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Mouse grip e envia os seguintes dados {"st":2,"g":7}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Open palm grip e envia os seguintes dados {"st":2,"g":8}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada pinch grip e envia os seguintes dados {"st":2,"g":9}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Power grip e envia os seguintes dados {"st":2,"g":10}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Precision close gripe envia os seguintes dados {"st":2,"g":11}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Precision open gripe envia os seguintes dados {"st":2,"g":12}	Salvar e visualizar os valores recebidos
	S2	Sugere o tipo de pegada Tripod grip e envia os seguintes dados {"st":2,"g":13}	Salvar e visualizar os valores recebidos
Aceita a sugestão do tipo de pegada	S3	Envia os seguintes dados {"st":3}	Aceita e executa a contração
Executa uma contração de ação secundária	S4	Envia os seguintes dados {"st":4}	Uma ação secundária será ativada

### 3.7 Sugestor de pegada

O módulo sugestor de pegada é o encarregado de sugerir ao usuário o melhor padrão de pegada para o objeto com que quer interagir. O controlador envia uma imagem do objeto e o sugestor de pegada retorna uma lista ordenada das possíveis pegadas para aquele objeto.

Como foi explicado na figura 3.2, o bloco sugestor de pegadas foi implementado de duas maneiras. A primeira faz uso de um algoritmo que reconhece o objeto que o usuário quer interagir e com o nome do objeto procura em um banco de dados a lista das melhores pegadas com base no histórico do usuário. Na segunda opção o algoritmo sugere diretamente uma lista das possíveis pegadas para aquele objeto.

A figura 3.14 representa um exemplo de sugestão de pegada, onde a entrada é uma imagem do objeto, a figura 3.14 (A) apresenta um exemplo da classificação de pegada baseado no reconhecimento de objetos. Inicialmente se faz a classificação do objeto onde a saída é a etiqueta *Coffe Mug*. Com essa informação se faz a procura no banco de dados, dependendo da frequência com que o usuário pegou aquele objeto e se retorna uma lista ordenada com as possíveis categorias de pegada. Neste caso, onde usuário pegou dez vezes o copo com a pegada *Power grip* e quatro vezes com *Hook grip*, o sistema vai sugerir então *Power grip* como primeira opção.

A figura 3.14 (B) representa o bloco sugestor de pegada baseado no reconhecimento de pegada; neste caso o nome do objeto não é importante, pois esse bloco faz diretamente a recomendação de tipo de pegada usando visão computacional. Para o *Coffe Mug* esse bloco sugere *Power grip* como primeira opção de pegada.



Figura 3.14: Exemplo de sugestão de pegada. (A) usando reconhecimento de objetos. (B) Usando reconhecimento de categorias de pegada.

As soluções de inteligência artificial existentes normalmente operam em plataformas poderosas com alta disponibilidade de recursos computacionais. No entanto, as próteses exigem novas soluções com inteligência artificial incorporada em uma plataforma altamente móvel. Este trabalho explora os recursos de aprendizado de máquina (ML) em uma plataforma móvel baseada em *smartphone* para aplicativos. A vantagem de executar modelos em aplicativos móveis em vez de enviá-los para a nuvem é a redução da latência e a capacidade de garantir a privacidade dos dados para os usuários.

### 3.7.1 Sugestor de pegada baseado no reconhecimento de objeto

O funcionamento deste módulo, ilustrado na figura 3.15, tem início com o envio de um comando (uma contração) por parte do usuário. O aplicativo no *Android* captura a imagem do objeto para ser processada através de um algoritmo de reconhecimento de imagem para saber com que objeto o usuário quer interagir. Ao descobrir qual é o objeto, o sistema realiza uma busca em uma base de dados para saber qual tipo de preensão é a mais provável de ser utilizada pelo usuário para interagir com aquele objeto. Esta preensão é então sugerida para o usuário através de um *feedback* visual e/ou sonoro com o nome da interação.

Caso o usuário aceite a preensão, o comando especificado será enviado para a prótese de mão. O usuário pode enviar uma contração de cancelamento para que a prótese retorne ao estado inicial de repouso.

Caso o usuário não queira interagir com o objeto da forma que o sistema sugeriu, ele poderá realizar uma contração de cancelamento. Ao detectar essa contração, o sistema irá buscar na base de dados a próxima contração mais provável que sirva para o usuário interagir com o objeto. Este procedimento deverá se repetir até que o usuário encontre o tipo de interação correta ou até que as opções se esgotem. Caso as opções se esgotem, o usuário poderá configurar um tipo de preensão personalizado. Ao adicionar uma nova forma de manipular um determinado tipo de objeto, o banco de dados é atualizado para levar as decisões do usuário em consideração nas próximas vezes em que ele queira interagir com o objeto.

Ainda assim, é possível que o usuário queira interagir com um objeto que ainda não foi visto pelo sistema. Neste caso, todos os tipos de preensões que estão disponíveis no sistema terão a mesma probabilidade de serem usadas. Então uma ordem pré-programada no sistema é sugerida para o usuário.

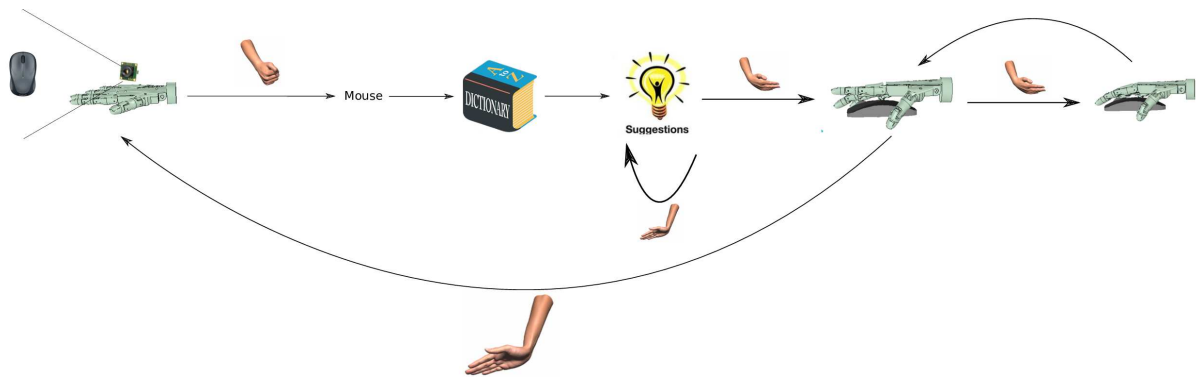


Figura 3.15: funcionamento do módulo sugestor de pegada baseado no reconhecimento de objeto.

### Diagrama de fluxo

A figura 3.16 mostra o diagrama de fluxo da interface implementada com o módulo sugestor de pegada fazendo uso do reconhecimento de objetos. Quando o usuário envia um comando ao controlador, este chama o bloco interface câmera para a captura da imagem, depois envia essa imagem para o bloco sugestor de pegada que recebe e ajusta a imagem, importa o modelo da rede treinado, classifica e retorna o nome do objeto. Com a etiqueta do objeto, procura no banco de dados e retorna uma lista ordenada das prováveis pegadas.

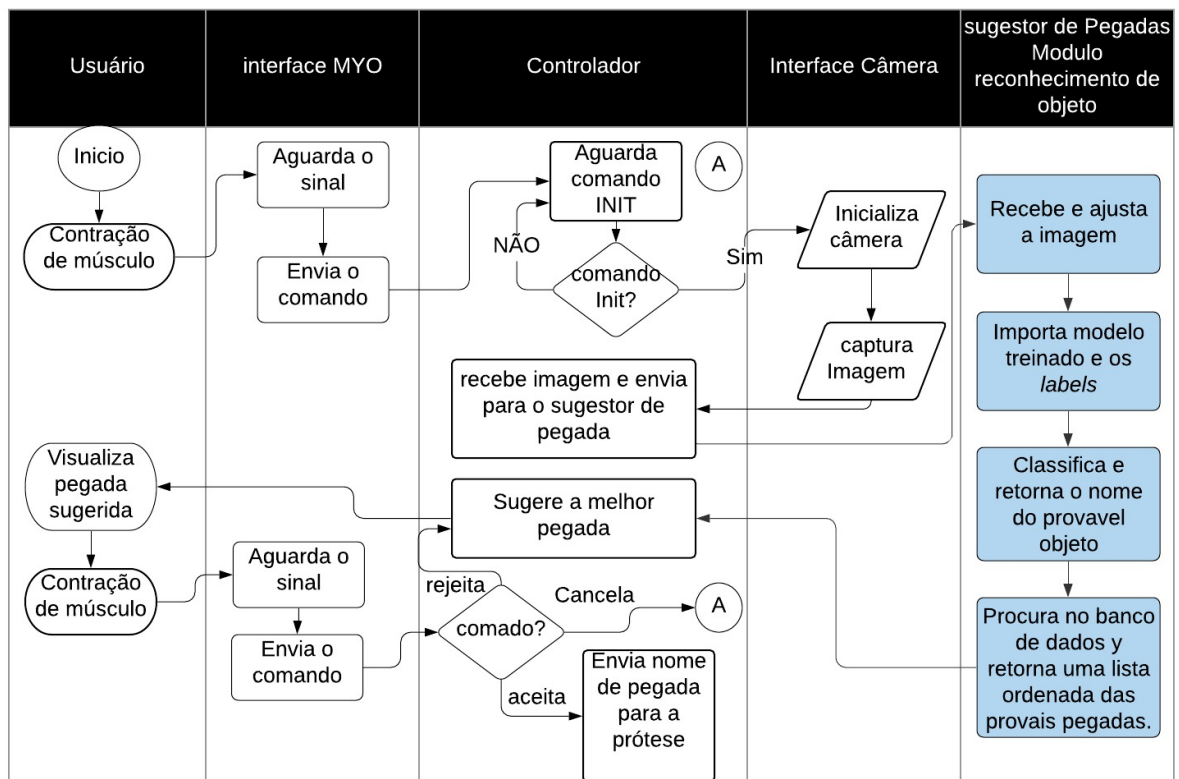


Figura 3.16: Diagrama de fluxo do módulo sugestor de pegada usando reconhecimento de objetos.

## Desenvolvimento

Para a implementação do módulo sugestor de pegada baseado no reconhecimento de objetos, gerou-se um modelo que permite a classificação dos objetos automaticamente no celular. Para esse propósito seguiram-se duas fases principais apresentadas na figura 3.17, o treinamento off-line do modelo e a operação no dispositivo para detectar objetos. Como resultado da primeira fase tem-se um modelo de classificação de padrões de imagens. Na segunda fase o classificador já está incorporado ao aplicativo *Android* que, após classificar uma imagem, gera uma pontuação para cada objeto que ele é capaz de identificar. Essa pontuação é a probabilidade daquele objeto estar presente naquela imagem. Dessa forma, é possível selecionar o objeto que está com a maior pontuação e direcionar a pegada conforme aquele objeto.

Inicialmente foram pesquisados os bancos de imagens de objetos existentes na internet, depois se definiu o banco de imagens para o reconhecimento de objetos que seriam utilizados neste projeto. Depois se procurou e testou, entre vários *frameworks* que permitem a instalação no celular, o melhor modelo com maior desempenho na hora de reconhecer o objeto. Finalmente se integrou e testou o modelo no *Android*.

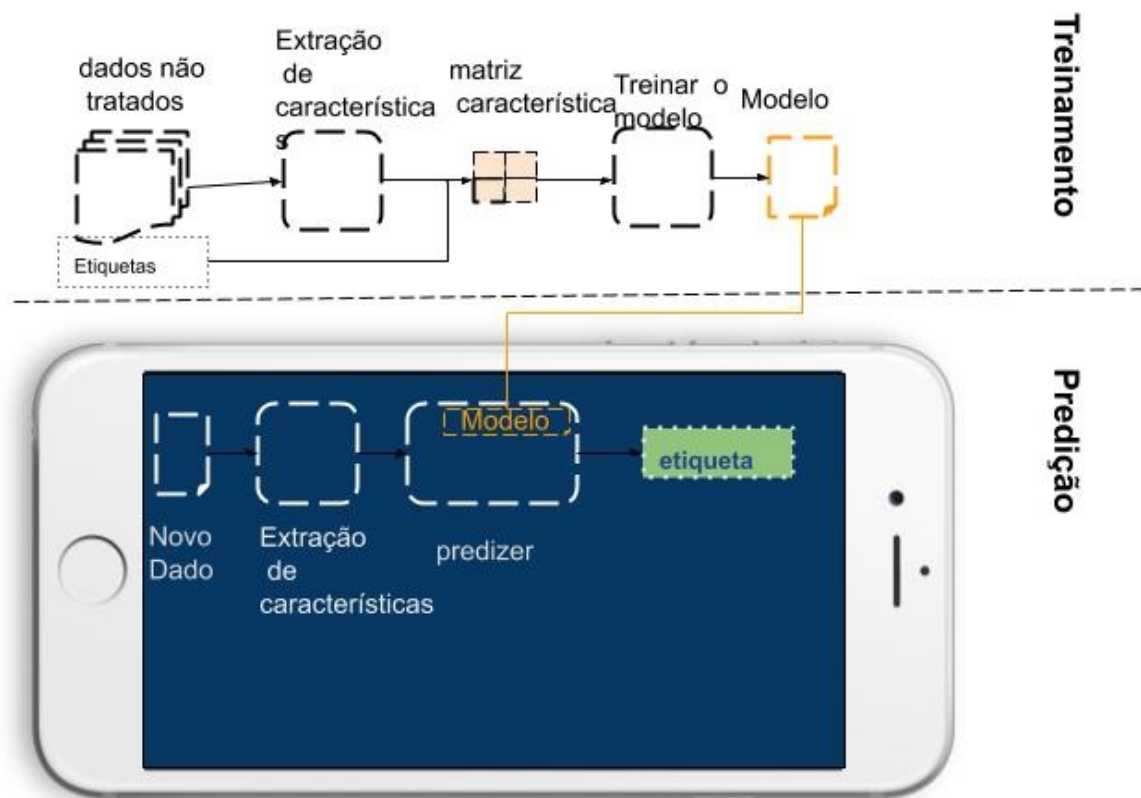


Figura 3.17: Arquitetura para o classificador de objetos no celular.

O banco de imagens foi usado para treinar a rede. Com o modelo e as etiquetas geradas, o modelo usando TensorFlow Lite foi otimizado. O código que permitiu a classificação de objetos no smartphone foi implementado e foram feitos testes com diferentes objetos para verificar o funcionamento. Finalmente, o reconhecedor de objetos foi integrado com o banco de dados e com o módulo controlador.

A criação do conjunto de dados é a primeira das muitas etapas necessárias para treinar o modelo com sucesso. Neste projeto, a rede foi treinada com milhares de imagens retiradas do banco de dados ImageNet, que compreende 100.000 imagens e cerca de 1.000 categorias de objetos. Essa base de dados teve de ser adaptada aos requisitos do sistema, a fim de reduzir a ambiguidade dada pelos rótulos específicos. Assim, o tamanho foi reduzido para 500 categorias usadas com frequência. O conjunto de dados foi dividido em: treinamento (80%), validação (10%) e teste (10%). Para determinar a melhor configuração de rede, o módulo foi treinado usando diferentes filtros convolucionais e tamanhos de entrada.

Foi criada uma pasta contendo subpastas com nome dos objetos, cada uma contendo várias imagens para cada etiqueta. Os nomes das subpastas são importantes, pois definem qual etiqueta é aplicada a cada imagem, mas os nomes dos arquivos não importam.

A detecção de objetos em imagens é uma tarefa com inúmeras aplicações e diversos



métodos para realizá-las. A seção 2.1 apresenta o resultado da pesquisa sobre várias ferramentas que permitissem a classificação de objetos usando um smartphone. Analisando as características de cada ferramenta, neste projeto se decidiu usar TensorFlow Lite porque este *framework* permite executar modelos aprendidos por máquina em dispositivos móveis com baixa latência, permitindo a classificação de objetos sem necessariamente incorrer em uma ida e volta para um servidor.

TensorFlow também disponibiliza uma biblioteca em Java que permite a integração fácil com os outros módulos desenvolvidos nesta interface.

A construção de uma rede convolucional no TensorFlow pode ser feita com vários graus de personalização, desde o simples treinamento da última camada de um modelo pré-treinado até o treinamento de uma rede completa com uma arquitetura personalizada (ou seja, estrutura do modelo). A quantidade de treinamento necessário depende da semelhança do conjunto de dados usado para um determinado modelo pré-treinado e do tamanho do conjunto de dados [52]. A tabela 3.3 apresenta uma recomendação de qual estratégia de treinamento usar dependendo do tamanho da base das imagens.

Tabela 3.3: Recomendações de estratégia de treinamento, adaptado de [52].

Tamanho do novo banco de imagens	Similar ao antigo banco de imagens	Que fazer
Grande	Alta	Modelo pré-treinado usando <i>fine tuning</i>
Pequeno	Alta	Treinar um linear classificador com recursos da CNN.
Pequeno	Baixa	Treinar um classificador a partir da ativação nas camadas inferiores.
Grande	Baixa	Treinar CNN do zero

Seguindo as recomendações de [52], e tendo em consideração que o tamanho do novo banco de imagens de objetos é grande (400 categorias), e que a nova base de dados é muito similar a antiga base ImageNet, se decidiu usar um modelo pré-treinado. Um modelo pré-treinado é uma rede salva que foi treinada anteriormente em um grande conjunto de dados, geralmente em uma tarefa de classificação de imagem em larga escala. Esses mapas de recursos aprendidos podem ser aproveitados sem precisar um começo do zero, treinando um modelo grande em um grande conjunto de dados.

A velocidade de processamento é um recurso essencial na implantação de redes convolucionais em tempo real em dispositivos móveis. Algumas arquiteturas ConvNet oferecem classificação mais eficiente do que outras. Por exemplo, o Inception-V3 do Google oferece uma precisão um pouco mais alta nas classes ImageNet com tempo de computação substancialmente menor [53]. O Inception-V3 é um ótimo modelo, mas é lento e pesado para dispositivos móveis, ocupa muito espaço e memória (quase 100 MB). Além disso, o tempo de processamento de entrada para saída leva de 200 a 300 ms para processar uma imagem de entrada  $224 \times 224$  em um telefone de *hardware* mais modesto (Nexus 5). Existem famílias de modelos de visão computacional para dispositivos móveis para o TensorFlow chamados MobileNets, projetados para maximizar efetivamente a precisão ao mesmo tempo em que estão atentos aos recursos restritos de um aplicativo incorporado ou no dispositivo. MobileNets são modelos pequenos, de baixa latência e baixa potência, parametrizados para atender às restrições de recursos de vários casos de uso. Eles podem ser construídos para classificação, detecção, incorporação e segmentação semelhantes à maneira como outros modelos populares de larga escala, como o Inception.

As MobileNets são uma classe de rede neural convolucional projetada por pesquisadores do Google arquitetados desde o início para facilitar o uso de recursos e executar rápido e diretamente no telefone. A principal diferença entre a arquitetura MobileNet e uma CNN 'tradicional' é que, em vez de uma única camada de convolução  $3 \times 3$ , o MobileNets usa uma convolução  $1 \times 1$  ponto a ponto [54].

Então o problema é a precisão. As MobileNets geralmente não são tão precisas quanto as redes maiores e com mais recursos. As MobileNets apresentam dois parâmetros ajustáveis para se adequar ao requisito de recurso/precisão: - O primeiro número

pode ser '100', '075', '050' ou '025' para controlar o número de neurônios (ativações de camadas ocultas); o número de pesos (e, em certa medida, o tamanho e a velocidade do arquivo) diminui com o quadrado dessa fração. - O segundo número é o tamanho da imagem de entrada e pode-se escolher '224', '192', '160' ou '128', com tamanhos menores, proporcionando velocidades mais rápidas.

O TensorFlow vem com ótimas ferramentas que podem ser usadas para treinar novamente o MobileNet. Para este módulo, foi usado o script existente na pasta de exemplo do site do TensorFlow.

Para iniciar o treinamento, foi executado o seguinte comando a partir da raiz do repositório TensorFlow:

```
python TensorFlow/examples/image_retraining/retrain.py -image
_dir objects/ -learning_rate=0.0001 -testing_percentage=10 -validation
_percentage=10 -train_batch_size=32 -validation_batch_size=-1 -flip
_left_right True -random_scale=30 -random_brightness=30 -eval_step
_interval=100 -how_many_training_steps=600 -architecture mobilenet
_1.0_224
```

O parâmetro *architecture mobilenet* é onde dizemos ao *script* qual versão do MobileNets queremos usar. Como se explicou existem diferentes modelos MobileNets para escolher. Por essa razão, uma pequena variedade de modelos foi treinada para ver como eles se saíam.

O parâmetro *learning\_rate* (taxa de aprendizado) é um hiperparâmetro que controla o quanto se está ajustando os pesos da rede em relação ao gradiente de perda [55].

Os parâmetros *-testing\_percentage* e *-validation\_percentage* é onde o script divide os dados em porcentagem de treino, teste e validação. Nossa base de dados foi dividida em 80% treino, 10% validação e 10% teste.

Foram treinadas várias arquiteturas com diferentes modelos e parâmetros. Os resultados do treino das redes serão apresentados na seção 4.1.1.

Depois de obter um modelo leve, rápido e preciso o suficiente para o nosso caso de uso, o modelo foi carregado em um aplicativo *Android* para conseguir testá-lo no mundo real.

Aproveitando as ferramentas fornecidas pelo TensorFlow se usou o projeto de exemplo do *Android* existente no site do TensorFlow. O projeto *Android* para executar esse tipo de tarefa pode ser encontrado na pasta */examples/android*.

Foram copiados o modelo MobileNets personalizado e as etiquetas para o projeto no *Android* e depois foram configurados alguns parâmetros para finalmente testar o classificador de objetos.

Depois de testado o classificador de objetos, desenvolveu-se um sub-bloco que permite procurar no banco de dados que contém o histórico de pegadas do usuário o provável tipo de pegada para um objeto classificado.

Na figura 3.18 é apresentado o modelo de banco de dados. Uma tabela contém os dados com a descrição dos objetos e outra com os nomes dos tipos de pegada. Quando um objeto é classificado, é executada uma consulta na tabela *INTERACTION* para obter o tipo de pegada que mais vezes é aceito para esse objeto olhando o valor armazenado no contador.

Se o usuário aceita essa sugestão de pegada, um contador é incrementado tendo em vista a identidade do objeto e o tipo de pegada.

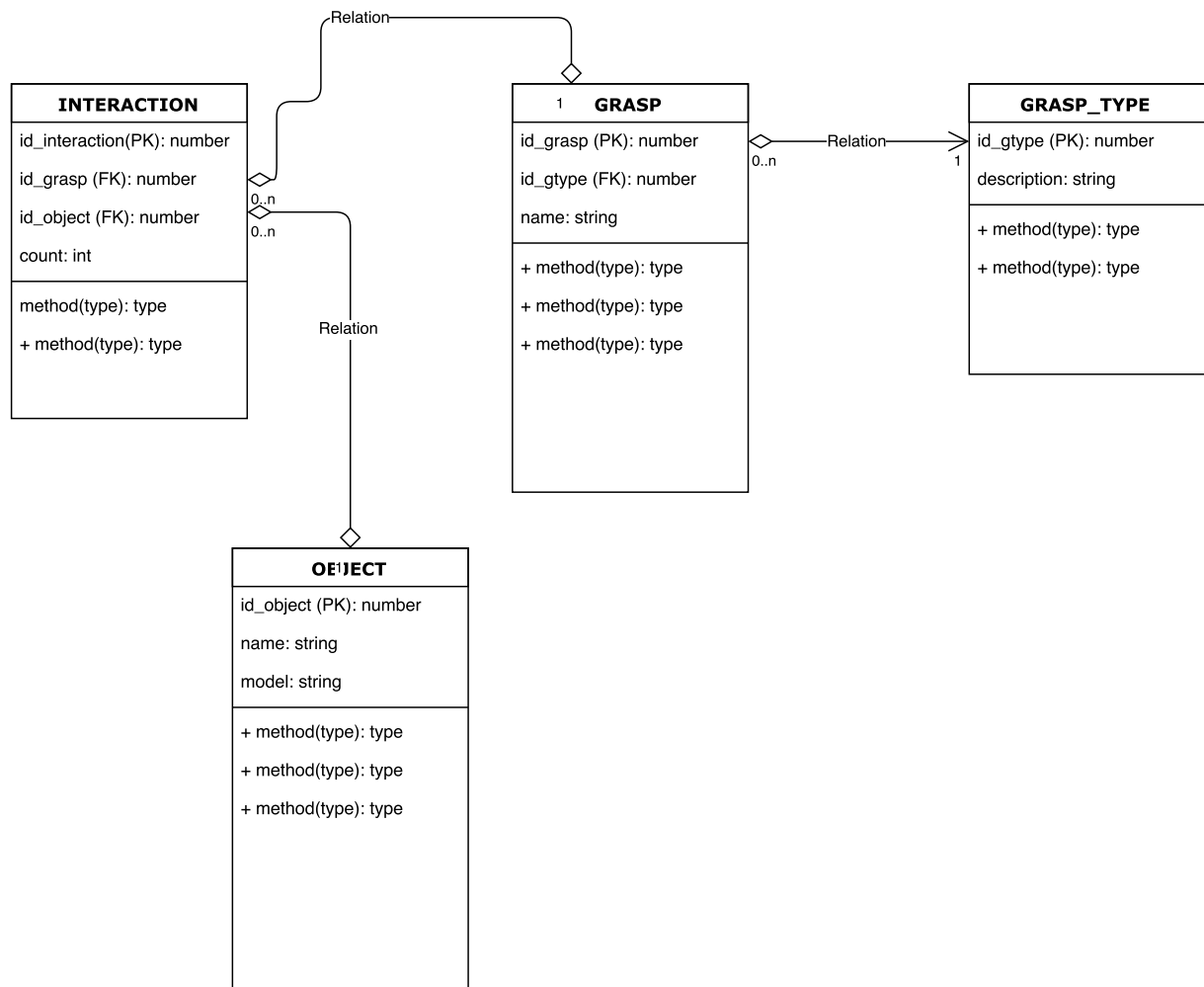


Figura 3.18: Diagrama de Classes do banco de dados que contém informações dos objetos, categorias de pegadas e interações dos usuários.

### 3.7.2 Sugestor de pegada baseado no reconhecimento de pegada

A principal novidade conceitual nesta versão é a classificação dos objetos com relação ao padrão de apreensão sem identificá-los explicitamente ou medir suas dimensões. Os objetos não são classificados com base na categoria ou identidade do objeto, mas sim com base no padrão de pegada adequado. Portanto, esta abordagem profunda baseada na aprendizagem que se generaliza para objetos não definidos, conceitualmente diferente do reconhecimento de objeto no qual os detalhes e o nome do objeto importam.

A figura 3.19 ilustra como esse módulo funciona. O usuário envia um comando (uma contração) para que o sistema capture a imagem de um objeto que ele queira interagir fisicamente, a imagem do objeto é processada pelo sistema através de um algoritmo de reconhecimento padrão de apreensão. Ao descobrir qual é a apreensão, o sistema sugere ao usuário através de um *feedback* visual e/ou vocal com o nome da interação.

Caso o usuário aceite a preensão sugerida pelo sistema, ele pode realizar uma contração de confirmação. Ao enviar o comando de confirmação, a posição dos dedos da prótese para que seja realizado o movimento correspondente é enviada através de uma conexão *Bluetooth* para a prótese de mão, que realiza o movimento. Ao término da interação com o objeto, o usuário pode enviar uma contração de cancelamento para que a prótese retorne ao estado inicial de repouso.

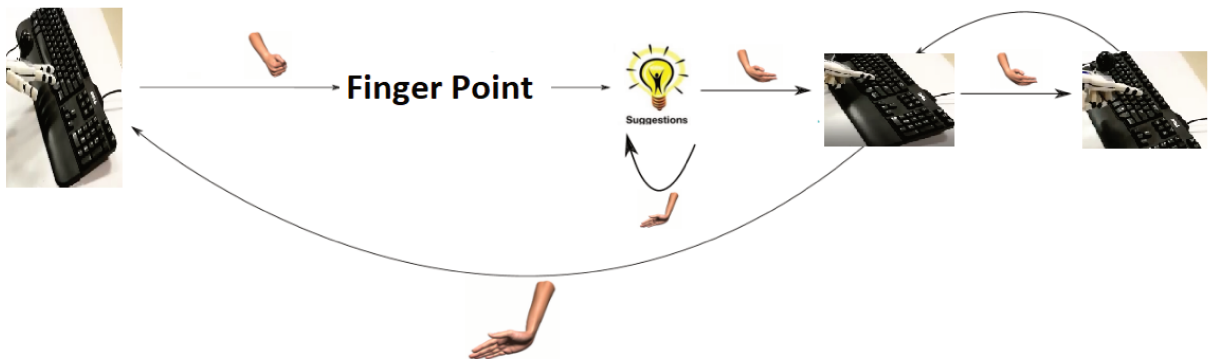


Figura 3.19: Visão geral do módulo sugestor de pegada baseado no reconhecimento de pegada.

Na figura 3.20 é apresentado o diagrama de comunicação entre o protótipo do Módulo sugestor de pegada baseado no reconhecimento de pegada e o *hardware* que compõe o sistema. Um aplicativo desenvolvido em *Android* mediante um controlador central recebe um sinal enviado pelo Myo para inicializar a predição do tipo de preensão usando um modelo de aprendizado de máquina embarcado no dispositivo e executado pelo TensorFlow. Finalmente, o aplicativo envia os sinais da preensão apropriada para uma placa de microcontrolador que se encarrega do controle da prótese de mão.

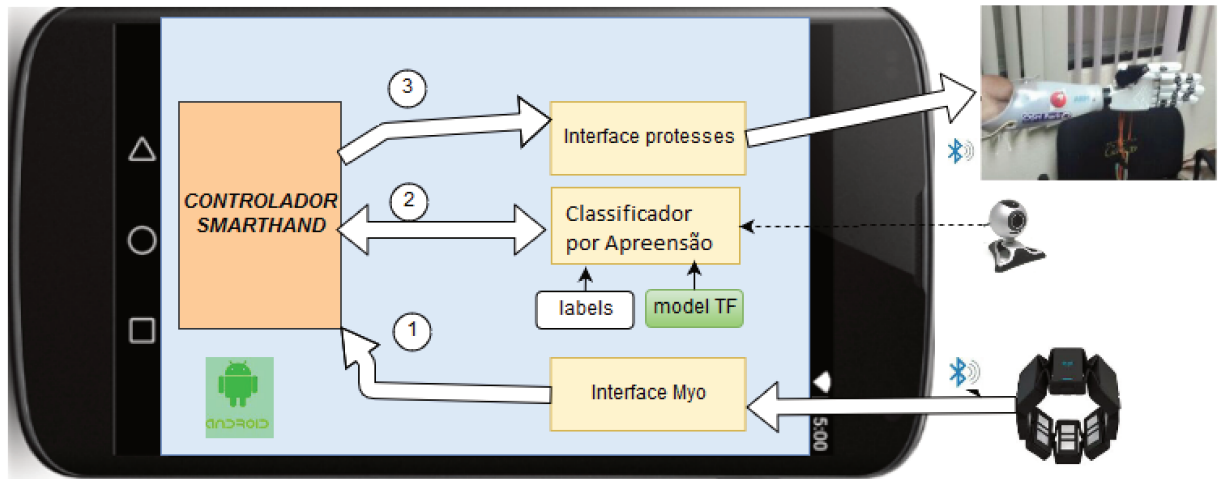


Figura 3.20: Arquitetura de comunicação entre módulo sugestor de pegada baseado no reconhecimento de pegada e o *hardware* que compõe o sistema.

## Diagrama de fluxo

A figura 3.21 mostra o diagrama de fluxo da interface implementada com o módulo sugestor de pegada fazendo uso do reconhecimento direto da pegada. Quando o usuário envia um comando ao controlador, o controlador chama o bloco interface câmera para a captura da imagem, depois envia essa imagem para o bloco sugestor de pegada que recebe e ajusta a imagem, importa o modelo treinado, classifica e retorna uma lista ordenada das prováveis pegadas.

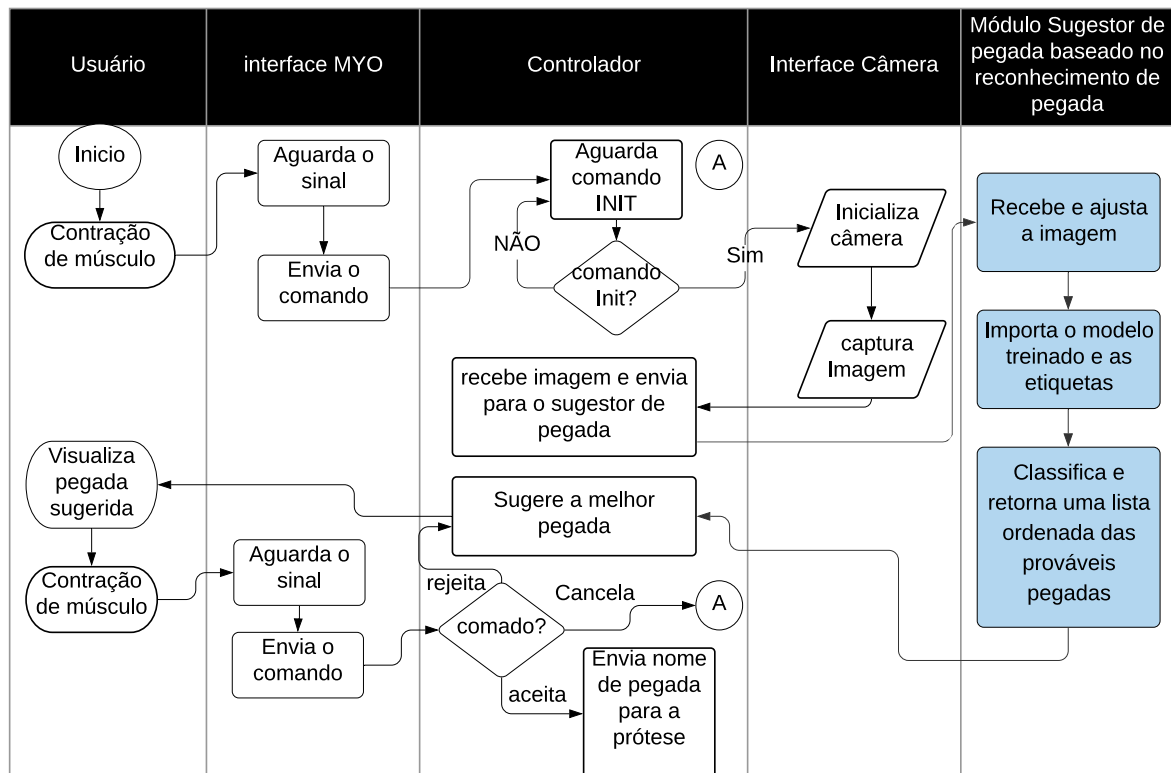


Figura 3.21: Diagrama de fluxo do módulo sugestor de pegada usando reconhecimento de pegada.

## Desenvolvimento

Para o propósito deste projeto que é sugerir diretamente ao usuário a forma para pegar um objeto, foi criado um banco de dados específico com os tipos de pegadas descritos na seção 3.3.2. As imagens dos objetos foram categorizadas em treze classes de apreensão, a saber: *Tripod grip*, *Precision closed grip*, *Hook grip*, *Column grip*, *Power grip*, *Finger adduction*, *Open palm grip*, *Pinch grip*, *Mouse grip*, *Precision open grip*, *Finger point*, *Active index grip*, *Key grip*. Alguns exemplos de imagens usadas para cada categoria são apresentados na figura 3.22.

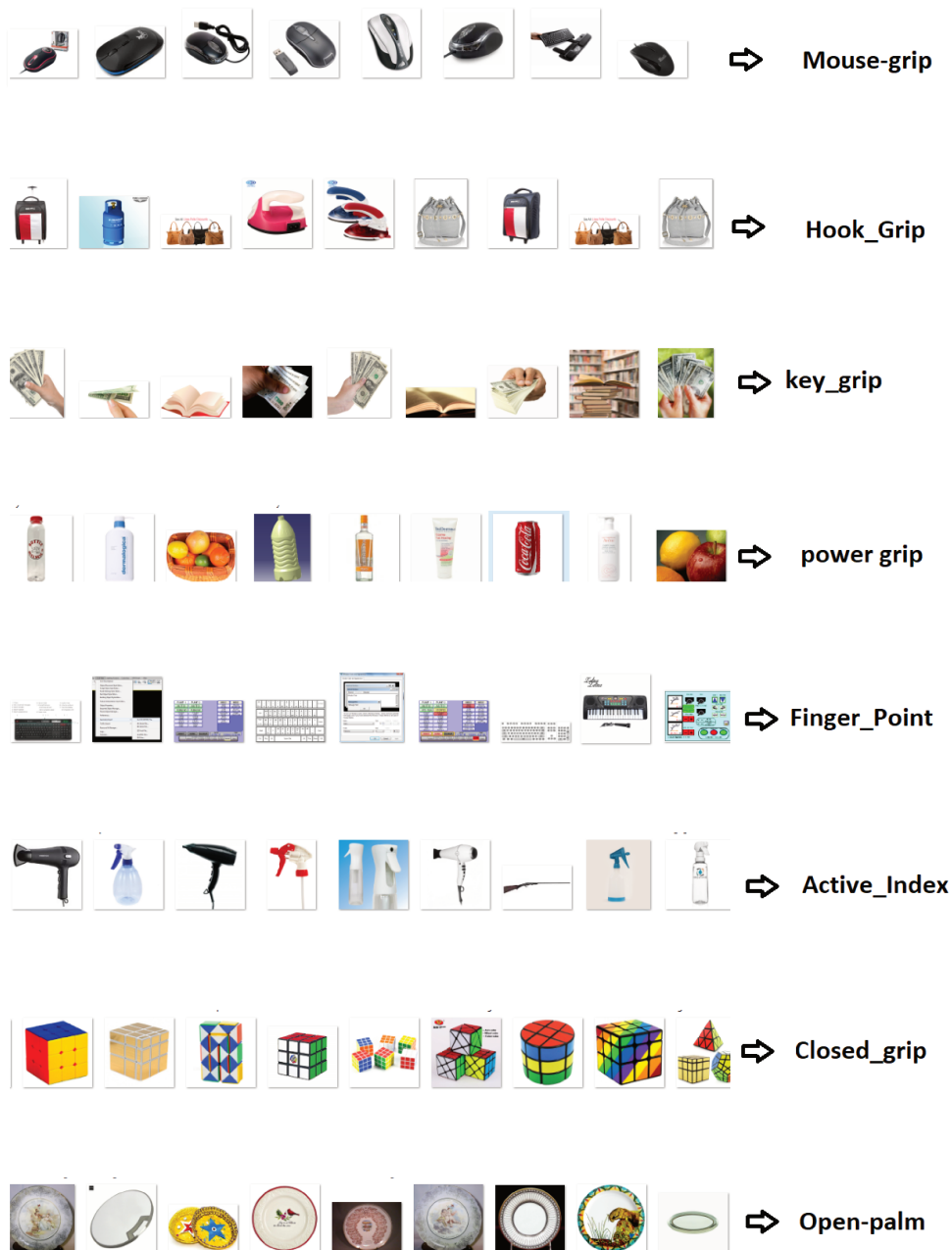


Figura 3.22: Algumas imagens usadas para treinar a rede.



A qualidade dos dados de entrada é fundamental para um bom treinamento. Por esse motivo que foi criado um algoritmo que permitisse avaliar a base de dados criada. Para visualizar o código junto com o *dataset* consulte, o Apêndice B.

Uma comparação de vários classificadores usando a biblioteca scikit-learn foi feita para a análise da base de dados. Inicialmente foi carregado o conjunto de dados que contém os subdiretórios com as imagens classificadas por pegada, usando uma divisão de 80% das imagens para treinamento e 20% para validação. Os classificadores usados são listados a seguir:

- *Decision Tree*: As árvores de decisão são um método de aprendizado supervisionado não paramétrico usado para classificação e regressão. O objetivo é criar um modelo que preveja o valor de uma variável de destino, aprendendo regras de decisão simples, inferidas dos recursos de dados [56].
- *Nearest Neighbors*: O princípio é encontrar um número predefinido de amostras de treinamento mais próximas em distância do novo ponto e prever o rótulo a partir deles [57].
- *SVM*: É um algoritmo interessante, com conceitos relativamente simples. O classificador separa os pontos de dados usando um hiperplano com a maior quantidade de margem. É por isso que um classificador SVM também é conhecido como classificador discriminativo. O SVM encontra um hiperplano ideal que ajuda a classificar novos pontos de dados [58].
- *VotingClassifier*: Combina vários modelos diferentes em um único modelo, o que é (idealmente) melhor do que qualquer um dos modelos individuais sozinho[59].
- *Gaussian Process*: É um método genérico de aprendizado supervisionado projetado principalmente para resolver problemas de regressão [59].

Inicialmente todos os classificadores ficaram com uma precisão muito baixa devido ao grande número de classes de pegada; finalmente se conseguiu uma melhoria reduzindo as categorias *Hook grip*, *Column grip*, *Power grip*, *Finger adduction*, *Open palm grip*, *Mouse grip*, *Finger point*, *Active index grip*, *Key grip*. No total existem 21390 imagens em 9 categorias disponíveis.

Depois da preparação geral e pré-processamento se configurou o treinamento usando o *Transfer Learning*. Na prática, uma rede neural convolucional inteira raramente é treinada do zero, porque é raro ter-se um conjunto de dados de tamanho suficiente. No entanto, com o *Transfer Learning* podemos treinar uma rede neural convolucional com um conjunto de dados de tamanho pequeno, porque estamos usando pesos pré-treinados da rede neural convolucional [60]. Por esse motivo ele apenas foi ajustado para nosso conjunto de dados.

Se usou um modelo da MobileNet na versão MobileNet\_v1\_1.0\_224, configurando o tamanho das imagens de entrada para 224x224x3, onde 3 representa as cores aditivas em que o Vermelho (Red), o Verde (Green) e o Azul (Blue) são combinados de várias formas de modo a reproduzir um largo espectro cromático. Os pesos pré-treinados foram baixados do site do TensorFlow.

O TensorFlow usa um gráfico de fluxo de dados para representar cálculos em termos das dependências entre operações individuais. O fluxo de dados é um modelo de programação comum para computação paralela que também se aplica às redes neurais no TensorFlow. O gráfico inteiro da MobileNet é armazenado num arquivo com extensão .pb fornecido para provisionar o novo modelo para dispositivos móveis.

Para iniciar o treinamento foi executado o arquivo train\_image\_classifier.py com alguns argumentos como o destino dos arquivos TFRecord, o conjunto de imagens para ser treinado, o modelo MobileNet\_v1\_1.0\_224 como modelo base, o método de treinamento e o número de etapas de treinamento. Os resultados do treinamento são apresentados na seção 4.1.2.

Depois de concluir o treinamento e a avaliação da MobileNet, se preparou a implementação no celular. Para isso, primeiro foi criado um gráfico de inferência da nova MobileNet. Como último passo, se otimizou o gráfico para dispositivos móveis. Esta otimização reduziu o tamanho binário do gráfico removendo operações desnecessárias para classificação e arredondando os pesos fornecidos do modelo. Arredondar os pesos fornecidos levou a uma pequena perda de precisão, mas melhorou muito a duração da classificação do modelo, o que é muito importante para dispositivos móveis.

Com o gráfico otimizado para dispositivos móveis e totalmente funcional seguiu-se a implementação e integração com os outros módulos da UPI no aplicativo para *Android*.

## 3.8 Interface do usuário

A interface do usuário para configuração do sistema é apresentada na figura 3.23. A tela 1 permite ao usuário selecionar as pegadas disponíveis na prótese; a tela 2 permite a conexão mediante o *Bluetooth* com a prótese; e a tela 3 possibilita as configurações da câmera e os tipos de sugestões de pegada e áudio. Um vídeo com os testes desenvolvidos pode ser visualizado em <https://www.youtube.com/watch?v=RP2M5jrL5JU>.

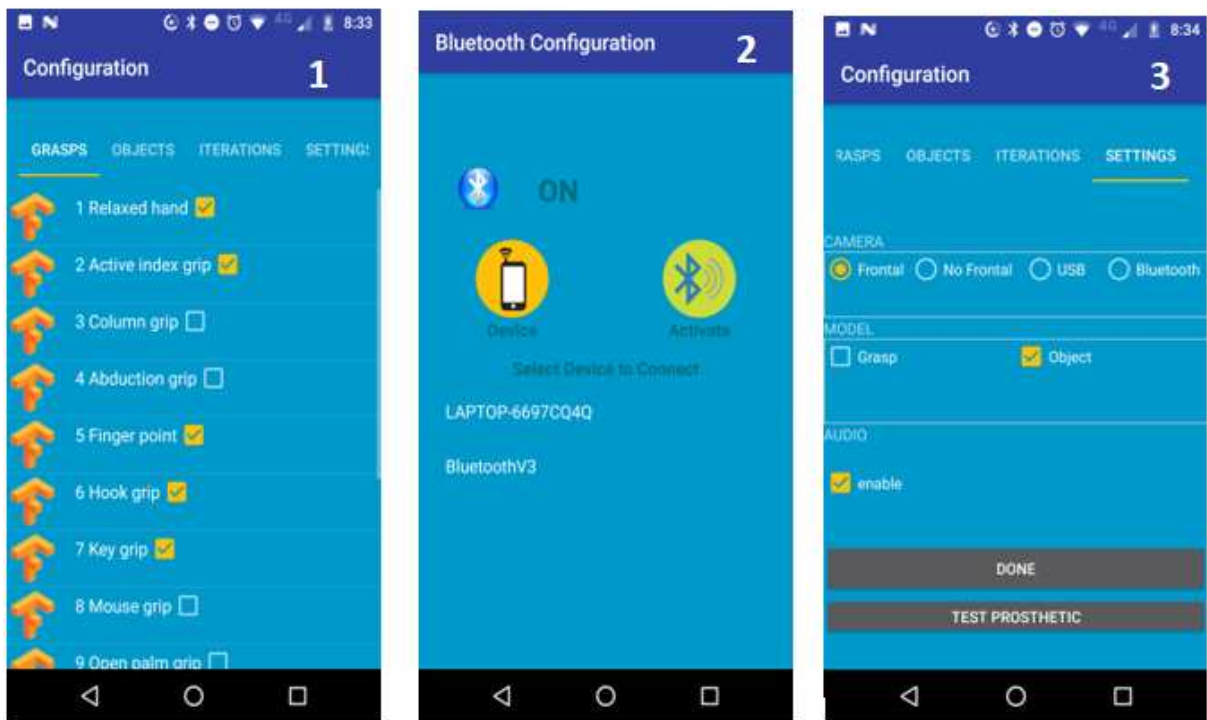


Figura 3.23: Interface de usuário - Configuração.

Na figura 3.24 se apresenta a interface de usuário para a seleção do tipo de pegada; inicialmente se faz a configuração da braçadeira Myo (1) e se apresenta uma tela principal onde o usuário pode determinar a intenção de pegada (2); quando o usuário pede para pegar um objeto se ativa uma câmera que captura a imagem e retorna um tipo de pegada provável (3,4); se o usuário aceitar, o nome do tipo de pegada é enviado para a prótese; se o usuário rejeitar outra pegada é sugerida e se cancelar o sistema volta ao estado inicial (2).



Figura 3.24: Interface de usuário -Funcionamento.

### 3.9 *Task Load Index* - TLX (Índice de Carga de Tarefa em tradução livre)

Para analisar a interface desenvolvida foi utilizado o procedimento de análise *Task Load Index* - TLX (Índice de Carga de Tarefa em tradução livre); esse método é um procedimento de classificação multidimensional baseado na média ponderada de seis escalas: Demanda Mental, Demanda Física, Demanda Temporal, Desempenho do Usuário, Esforço e Frustração. Essas definições são apresentadas na Tabela 3.4.

Tabela 3.4: Definições das 6 dimensões do método de mensuração NASA-TLX.

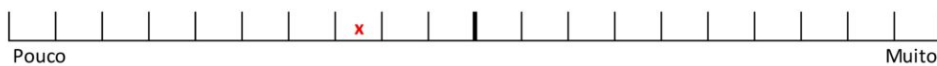
<b>Definições das 6 dimensões do método de mensuração NASA-TLX</b>	
<b>Dimensões</b>	<b>Definições</b>
Mental	Quantidade da atividade mental e perceptiva que a tarefa necessita (pensar, decidir, calcular, lembrar, olhar, procurar, etc.)
Física	Quantidade de atividade física que a tarefa necessita (puxar, empurrar, girar, deslizar, etc.)
Temporal	Nível de pressão temporal sentida. Razão entre o tempo necessário e o disponível.
Satisfação /Rendimento	Até que ponto o indivíduo se sente satisfeito com o nível de rendimento e desempenho no trabalho.
Esforço	Grau de esforço mental e físico que o sujeito tem que realizar para obter seu nível de rendimento.
Nível de frustração	Até que ponto o sujeito se sente inseguro, estressado, irritado, descontente, etc., durante a realização da atividade.

A média ponderada dessas escalas é calculada a partir da importância (subjetiva) de cada escala para os voluntários que testam o sistema, resultando na carga de trabalho global requerida pelo mesmo. Este método foi escolhido por fornecer dados para analisar quando uma interface requer menos esforço cognitivo para ser utilizado que outra.

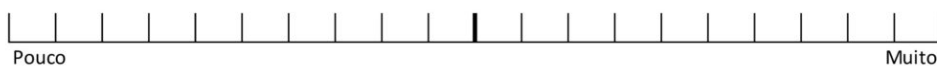
Essa avaliação funciona em duas etapas – a classificação bruta das escalas e a escolha da fonte de carga de trabalho. Na primeira parte, o objetivo é obter um valor numérico correspondente à magnitude de cada uma das escalas na tarefa que foi realizada. É apresentada ao operador uma régua, assim como na figura 3.25; a régua é dividida em 20 intervalos de 5 em 5 unidades, sendo os descritores de limite "Pouco"(0) e "Muito"(100). Os sujeitos devem marcar cada escala no local que acreditam ser o mais adequado para a tarefa que foram solicitados a executar.

**Marque na escala qual a sua opinião sobre o nível de influência dos fatores abaixo para a realização do seu mestrado**

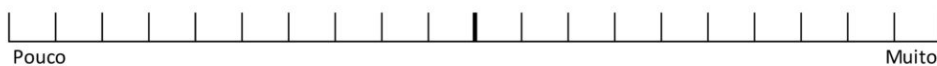
**EXEMPLO**



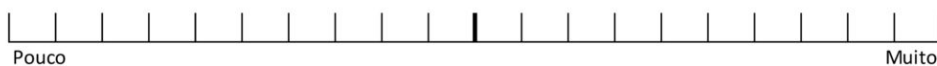
**Demanda Mental**



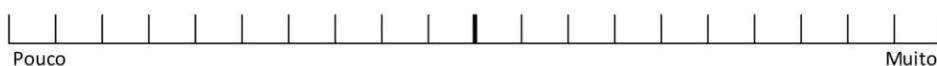
**Demanda Física**



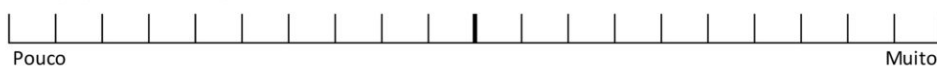
**Demanda Temporal**



**Performance**



**Esforço (Físico e Mental)**



**Nível de Frustração**

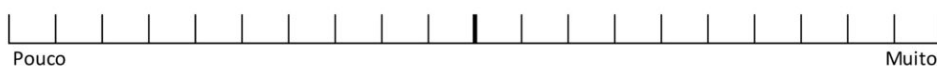


Figura 3.25: Exemplo de preenchimento de dados para medir as magnitudes.

A segunda etapa é onde os participantes irão avaliar a contribuição de cada uma das seis escalas para a carga de trabalho relacionada à tarefa realizada. Em outras palavras, é onde o peso de cada escala é calculado. Os participantes preenchem 15 perguntas onde as escalas são comparadas em pares; o número de vezes que uma escala é selecionada torna-se o peso da mesma, onde peso igual à 5 significa que a escala é a mais importante que qualquer outra e 0 significa que a escala não é relevante. Um exemplo de pergunta é apresentado na figura 3.26.

The figure displays three sequential screenshots of a web-based questionnaire for the NASA TLX scale evaluation. Each screenshot contains the following elements:

- Header:** "Clique no fator que representa a escala mais importante da carga de trabalho para a tarefa \*"
- Options:** Two radio button options with descriptive text in Portuguese.
- Buttons:** "Voltar" (Back) and "Próximo" (Next) buttons.

**Screenshot 1 (Top):**

- Option 1: ☐ Demanda Mental ->Quantidade da atividade mental e perceptiva que a tarefa necessita (pensar, decidir, calcular, lembrar, olhar, procurar , etc.)
- Option 2: ☐ Esforço->Grau de esforço mental e físico que o sujeito tem que realizar para obter seu nível de rendimento.

**Screenshot 2 (Middle):**

- Option 1: ☐ Satisfação /Rendimento->Até que ponto o indivíduo se sente satisfeito com o nível de rendimento e desempenho no trabalho.
- Option 2: ☐ Demanda Mental ->Quantidade da atividade mental e perceptiva que a tarefa necessita (pensar, decidir, calcular, lembrar, olhar, procurar , etc.)

**Screenshot 3 (Bottom):**

- Option 1: ☐ Satisfação /Rendimento->Até que ponto o indivíduo se sente satisfeito com o nível de rendimento e desempenho no trabalho.
- Option 2: ☐ Demanda Física->Quantidade de atividade física que a tarefa necessita\ (puxar, empurrar, girar, deslizar, etc.)

Figura 3.26: Exemplo de perguntas para avaliação das escalas NASA TLX.

O link com o formulário das perguntas para a avaliação das escalas pode ser visualizado em [61].

Depois que toda a informação das duas etapas de avaliação é reunida, a carga de trabalho global da tarefa pode ser calculada multiplicando-se os pesos das escalas por sua respectiva contribuição bruta e dividindo esse valor por 15 (o número de comparações realizadas na etapa 2).

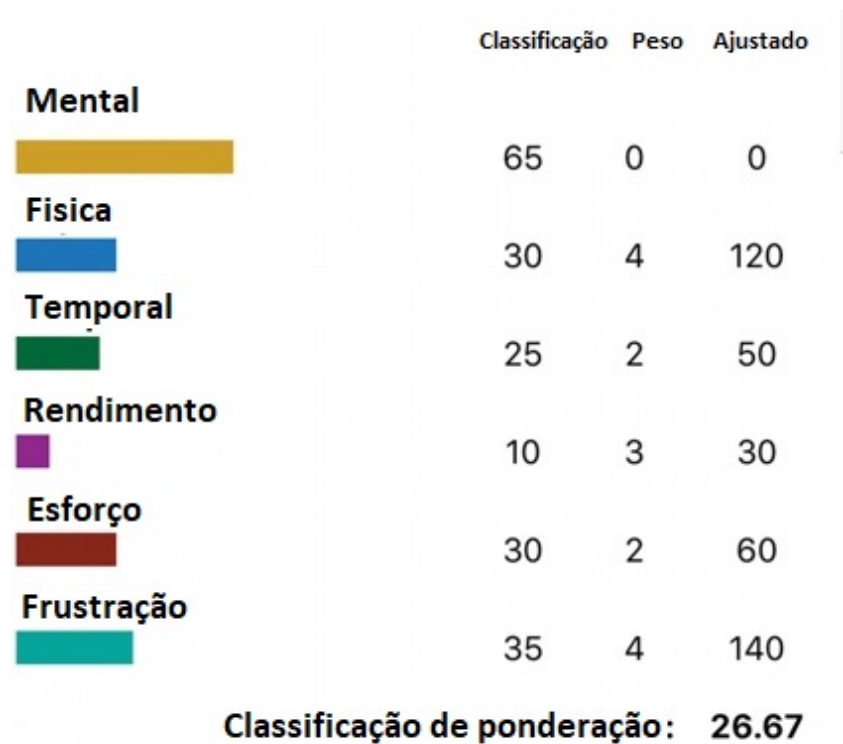


Figura 3.27: Exemplo de resultado para avaliação das escalas NASA TLX.

A Seção 4.1.3 apresenta os resultados de carga de trabalho para a interface que foi testada por 20 voluntários hígidos e um amputado. Todos os testes foram realizados em laboratório, seguindo um *script* e sob as mesmas condições para tornar os resultados reprodutíveis.

### 3.10 Resumo do capítulo

Este capítulo apresentou a metodologia utilizada para desenvolver e avaliar a interface para controlar uma mão protética. Primeiro a arquitetura proposta foi descrita; em seguida, os módulos também foram detalhados para garantir que outros possam reproduzir este trabalho. Finalmente, foi apresentado o procedimento experimental completo usado para testar e comparar a interface desenvolvida.



# Capítulo 4

## Resultados e análises

### 4.1 Resultados

#### 4.1.1 Resultados para o módulo sugestor de pegada baseado no reconhecimento de objetos

##### Resultados do treinamento

Como foi falado na seção 3.7.1, para obter um modelo pré-treinado que permitisse uma classificação de objetos com maior precisão, tamanho pequeno e boa velocidade, foram feitos experimentos com diversos modelos de visão computacional específicos para aplicativos móveis. Criou-se uma base de dados com imagens dos objetos mais usados no cotidiano e várias configurações de modelos foram treinadas para encontrar a rede que atingirá os melhores parâmetros de precisão, tamanho e rendimento.

Se treinou o modelo referência executando o *Inception-V3*, seguido das diferentes configurações das *MobileNet* iniciando pelo mais amplo: o *MobileNet (1,0 ; 128)*. O modelo resultante foi atualizado no aplicativo *Android* para medir o rendimento da CPU do celular.

O modelo gerado com *Inception-V3* teve uma precisão de 91% e gerou um arquivo de 86 Mb. Ele roda a cerca de 4 *frames* por segundo. *Inception-v3* é um ótimo modelo, porém é lento e pesado para dispositivos móveis, ocupa muito espaço e memória (quase

Modelo	Precisão	Milhões de multi-adições	Milhões de parâmetros	Tamanho do arquivo
Inception V3	91%	5000	23.2	86MB
1.0 MobileNet-224	87.3%	569	3.3	17MB
0.75 MobileNet-224	85.9%	325	1.9	10MB
1.0 MobileNet-192	83.9%	418	3.3	2MB
0.75 MobileNet-192	80.2%	239	1.9	93KB

Tabela 4.1: Comparação de resultados de detecção de objetos usando diferentes estruturas e arquiteturas de rede.

100 Mb). Além disso, o tempo de processamento para processar uma imagem de entrada  $224 \times 224$  em um telefone *Nexus 5* leva de 200 a 300 ms.

Usando o maior *MobileNet* (1,0; 224) se conseguiu alcançar 87,3% de precisão com apenas 5 minutos de treinamento. O tamanho do modelo resultante foi de apenas 17 Mb para quem mantém a pontuação, 7 vezes mais rápido e com apenas 4% de perda de precisão.

E a menor *MobileNet* (0,75; 192) usando pesos quantizados atingiu apenas 80,2% precisão. Mas o modelo ocupa apenas 930 Kb de memória e usa cerca de 40% da CPU do celular.

O melhor modelo que cumpre os requerimentos de precisão, tamanho e rendimento para nosso projeto é o modelo (0,75; 224), modelo este que foi usado como classificador de objetos dentro da interface UPI.

Com esse modelo, obteve-se (85,9%) de precisão com o conjunto de dados de treinamento. Foi validado com (400) iterações e obteve uma precisão de (84,60%) com o conjunto de dados de validação.

A figura 4.1 mostra a perda de entropia cruzada dos processos de treinamento e validação. A curva de cor azul representa o processo de treinamento e a curva de cor vermelha representa o processo de validação. Essa medida fornece *feedback* sobre o desempenho de nosso modelo de classificação antes de ser testado em um cenário real.

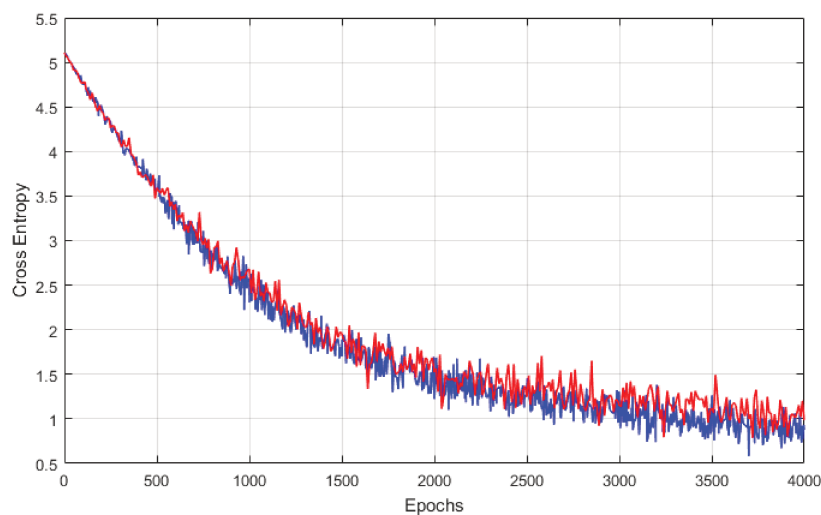


Figura 4.1: O erro de entropia cruzada do treinamento e validação definidos em cor azul e vermelho, respetivamente.

Os resultados apresentados na figura. 4.2 foram obtidos a partir da classificação dos objetos da vida cotidiana. Durante os testes, cinco objetos diferentes da mesma classe, foram escolhidos e apresentados aleatoriamente para ser classificados pelo celular.

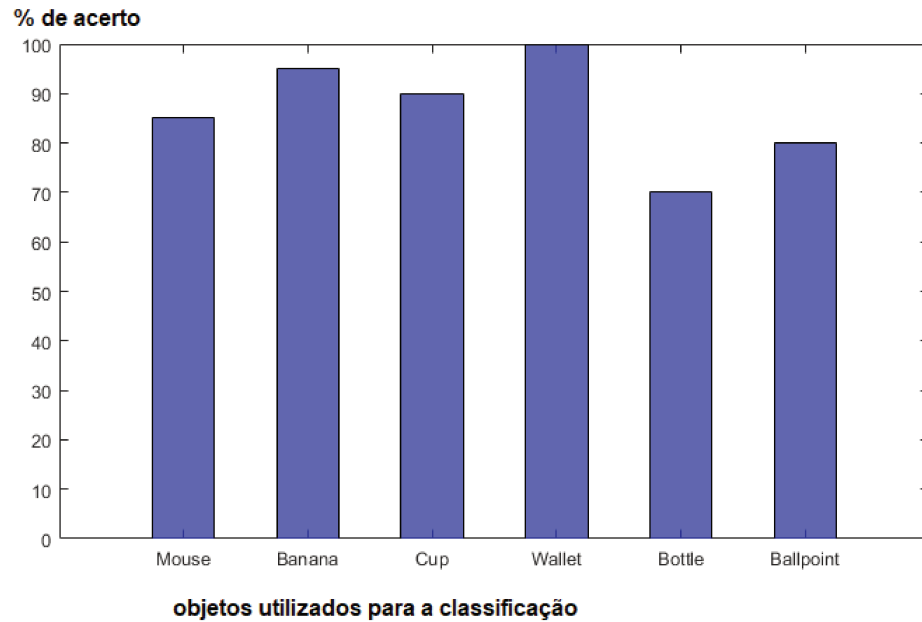


Figura 4.2: Precisão da classificação de um subconjunto de objetos da vida cotidiana.

### 4.1.2 Resultados para o módulo sugestor de pegada baseado no reconhecimento de pegada

Na seção 3.7.2 foi descrito o módulo sugestor de pegada, que é capaz de prever a forma desejada do usuário para pegar um objeto. Foi criada e analisada uma base de dados que contém imagens classificadas manualmente por pegada e foram avaliados vários modelos usando visão computacional que permitissem a classificação das imagens nos dispositivos móveis.

#### Análise da base de dados

Na figura 4.3 é apresentada uma comparação da precisão de treino e validação de todos os classificadores com a base de dados sem analisar; como resultado todos os seis classificadores ficaram com uma precisão abaixo dos 45%, o que significa que inicialmente essa base de dados não é muito boa.

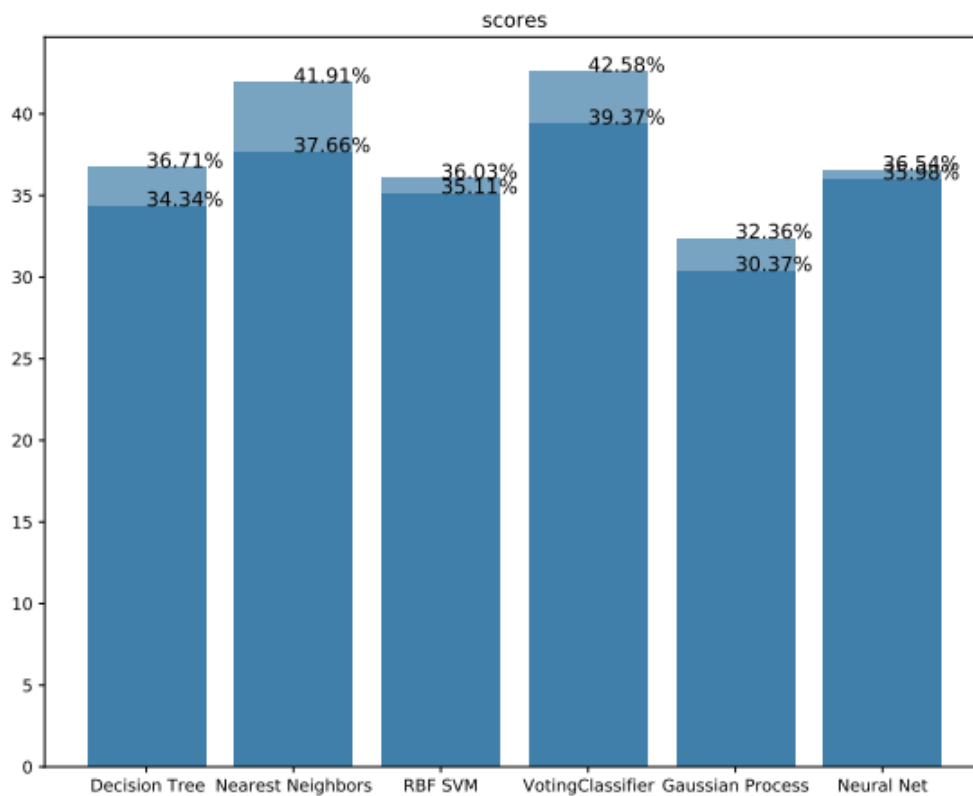


Figura 4.3: Comparação da precisão de treino e validação de seis classificadores sem a análise.

Para cada algoritmo de treinamento foi feita a análise da precisão de classificação por cada pegada, com o fim de identificar qual categoria possivelmente está gerando conflito e não permite uma boa classificação geral.

Para o algoritmo Neural Net as pegadas *Hook grip*, *Active index* e *Power grip* foram as que apresentaram a mais baixa qualidade de classificação, segundo a figura 4.4.

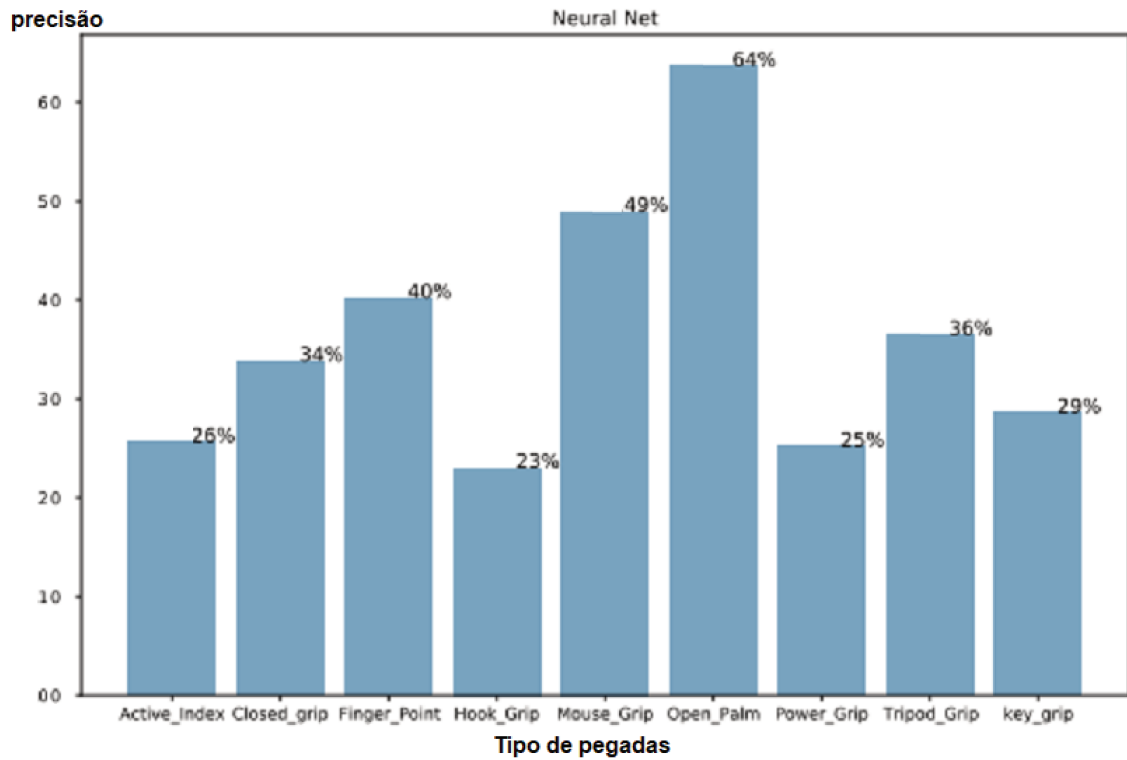


Figura 4.4: Precisão da classificação por pegada para o algoritmo Neural Net.

A figura 4.5 apresenta a comparação da precisão após a análise; a base de dados melhorou significativamente para todos os classificadores.

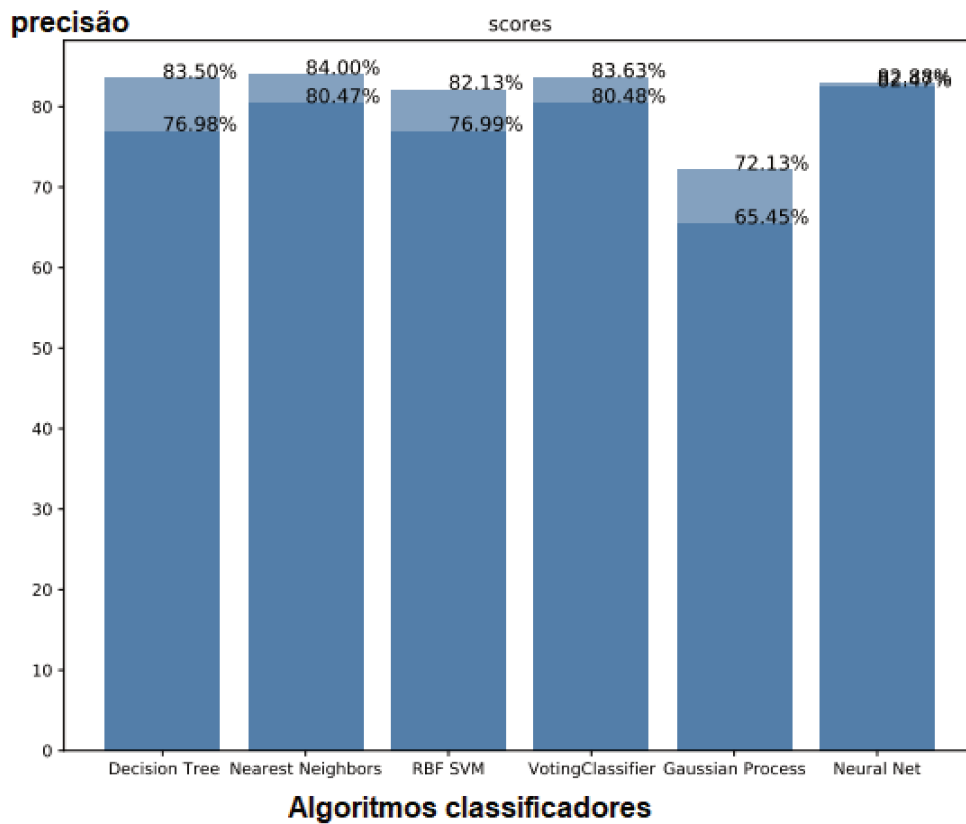


Figura 4.5: Comparação da precisão de treino e validação de seis classificadores após a análise.

## Resultados do treinamento usando o TensorFlow

Depois de criado e avaliado o *dataset*, foi realizado o primeiro teste treinando a rede com o modelo *Inception-v3* e com o resultado com esforços mínimos, se conseguiu atingir uma precisão de 95%, o que não é tão ruim para uma tarefa de classificação com nove (9) categorias. Este resultado foi alcançado sem uma otimização extensiva dos parâmetros da rede neural convolucional e também sem qualquer forma de regularização, o que indica que a análise feita com os classificadores descritos anteriormente ajudou na obtenção de uma base de dados eficiente.

Para melhorar os desempenhos, as redes foram treinadas depois usando os modelos da *MobileNet*. Como resultado dos testes avaliando a rede com o uso de vários modelos, concluiu-se que o modelo *MobileNet (0,75; 192)* atinge melhor os requisitos de precisão, rendimento e tamanho do arquivo. Esse modelo obteve uma precisão muito boa: 92% das amostras de classificação corretas após 300 épocas.

A figura 4.6 mostra a interseção de precisão de treino e precisão de validação. A precisão da validação garante a capacidade do modelo de generalizar para novos dados. O conjunto de dados de validação contém apenas os dados que o modelo nunca vê durante o treinamento e, portanto, não pode simplesmente memorizar. A precisão da classificação nos testes off-line atingiu 89% de precisão. Em seguida, a estrutura proposta foi implementada em tempo real em um *smartphone* padrão e se alcançou uma pontuação geral de 84% na classificação de um conjunto de objetos novos e vistos, mas rotacionados aleatoriamente.

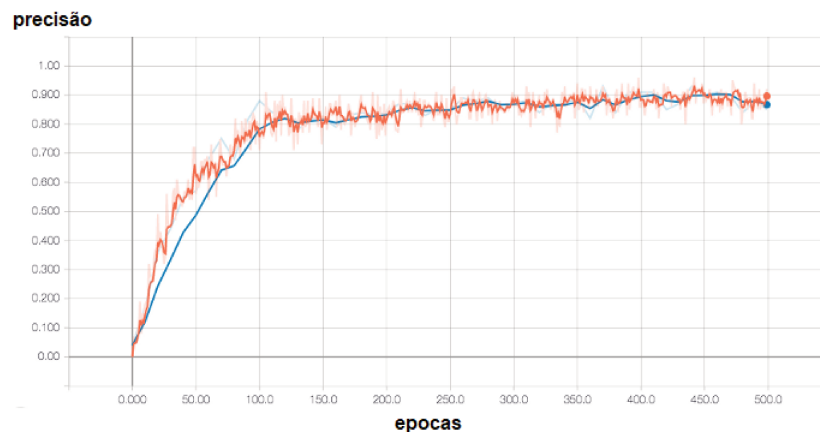


Figura 4.6: Gráfico de precisão para o modelo treinado com as imagens classificadas por pegada.

Para avaliar o módulo sugestor de pegada, baseado no reconhecimento de pegada, se comparou os resultados obtidos com os dados de treinamento de vários modelos usando um conjunto de dados denominado GraspSeg desenvolvido em [62]. Os resultados desses experimentos são relatados na tabela 4.2, que mostra que o classificador UPI desenvolvido neste trabalho teve uma precisão de pegada maior que todos os modelos treinados com o dataset GraspSeg. Essa melhoria pode ser atribuída a análise feita para a base de dados criada. O tamanho do modelo gerado pela UPI foi só maior do que o modelo ENet. O modelo Mobilnet treinado com a base de imagens de pegada UPI teve maior precisão, foi muito mais rápido e ocupou um tamanho menor de espaço que o modelo Mobilnet treinado com a base de dados GraspSeg. O Modelo Graspnet foi duas vezes mais rápido, teve uso de memória similar e uma precisão quase 2% menor do que o módulo implementado com a UPI.

Dataset	Modelo	Precisão de pegada (%)	Parâmetros (millions)	Tamanho do modelo (MB)	Memória (MB)	Time ms
GraspSeg	MobileNet	66.6	3.21	12.4	384	118
GraspSeg	AlexNet	67.8	56.8	22.7	451	25
GraspSeg	SegNet	77.7	29.4	112.3	699	74
GraspSeg	ResNet50	80.5	23.5	90.0	601	62
GraspSeg	ENet	81.7	0.41	1.4	523	97
GraspSeg	GraspNet (a - 0.5,8 - 8)	87.3	3.71	7.2	425	19
UPI dataset	MobilNet	89	3.8	2	420	40

Tabela 4.2: Comparação do modelo sugestor de pegada UPI e outros modelos CNN em termos de precisão, número de parâmetros de modelo, tamanho do modelo, consumo de memória do modelo e tempo de inferência do modelo com conjunto de dados GraspSeg.

### 4.1.3 Resultados para o teste da Interface UPI integrada com a prótese

Para validar e testar a funcionalidade da interface desenvolvida foram avaliados alguns requisitos como precisão de classificação, carga de trabalho, conjunto de preensões.

Exemplos do desempenho satisfatório dos tipos de pegadas são mostrados na figura 4.9. No entanto, alguns dos objetos precisam ser colocados em um local específico para serem agarrados pela prótese.



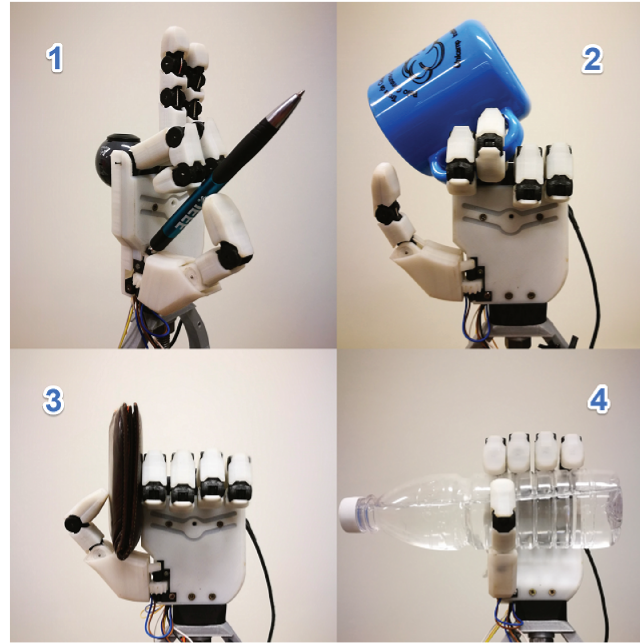


Figura 4.7: Apertos realizados no teste: (1) *Precision grasp*. (2) *Hook grasp*. (3) *Lateral grasp*. (4) *Power grasp*.

#### 4.1.4 Resultados da carga de trabalho usando o método de avaliação NASA TLX

Foram convidados 21 voluntários para avaliar a carga de trabalho da interface desenvolvida para o controle da prótese de mão apresentada na seção 3. Foram escolhidas várias classes de objetos da vida cotidiana: mouse, banana, caneca de café, carteira, garrafa e caneta esferográfica. Os voluntários interagiram com os objetos executando quatro apertos básicos definidos na taxonomia de Cutkosky [48]: *Tripod grip*, *Hook grip*, *Key grip* e *Power grip*.

Desses voluntários, 20 são pessoas híginas e um voluntário amputado. Os voluntários híginos tem entre 20 e 55 anos de idade e todos possuem pelo menos o ensino médio completo – 90% dos voluntários possuem ensino superior completo. O voluntário amputado tem 51 anos de idade e há 16 anos sofreu um acidente que causou a amputação do seu braço esquerdo por esmagamento. Entretanto, este lado não era o lado dominante do voluntário. Vale também ressaltar que o voluntário amputado já fez uso de uma prótese mioelétrica.

Para atingir reprodutibilidade dos testes, todos seguiram um roteiro pré-definido. Primeiro, os participantes forneceram consentimento por escrito para participar do experimento, que foi aprovado pelo Comitê de Ética da Universidade Estadual de Campinas (Brasil) sob o número CAAE 58592916.9.1001.5404 (consulte o Apêndice A). Em seguida, o pesquisador deveria preencher o formulário de participação do voluntário, com questões sobre o voluntário, como idade e gênero. Depois disso, o pesquisador deveria explicar como a interface a ser testada iria trabalhar e quais testes o usuário iria fazer. Após a explicação,

os experimentos começaram. Primeiramente, foi realizada a calibração da braçadeira. A calibração foi feita usando o *software* de braçadeira Myo. Após a calibração da braçadeira, foi realizado um teste quantitativo. Neste teste, o objetivo foi analisar se a orientação do braço teve alguma influência no resultado da classificação das contrações de confirmação e cancelamento. Os voluntários tiveram que realizar contrações de cancelamento e confirmação em 5 (cinco) diferentes orientações com o braço estendido e não estendido. Em seguida, foi apresentada a tarefa que os voluntários deveriam realizar, bem como instruções sobre como selecionar cada interação necessária para a realização dos testes.

Conforme indicado na seção 3.9, a avaliação do Índice de Carga de Tarefas da NASA foi realizada para avaliar a interface desenvolvida. Recapitulando, essa avaliação leva em consideração seis escalas: Demanda mental, Demanda física, Demanda temporal, Rendimento, Esforço e Frustração.

Para evitar a fadiga muscular, a maioria dos voluntários testaram as interfaces no mesmo dia, mas com intervalo de 5 minutos entre os testes para não sobrecarregar os músculos.

A avaliação NASA TLX foi feita só para a UPI usando o módulo reconhecedor de pegada baseado no reconhecimento de objetos. Os resultados do NASA TLX serão descritos em duas partes. A primeira se refere exclusivamente aos resultados obtidos com voluntários hígidos, enquanto a segunda parte se refere aos resultados obtidos com o voluntário amputado.

#### 4.1.5 Resultados Obtidos com os Testes dos Voluntários Hígidos

A tabela 4.3 apresenta os resultados obtidos dos testes com os vinte voluntários hígidos, depois que toda a informação das duas etapas de avaliação foi reunida, a carga de trabalho global da tarefa foi calculada multiplicando-se os pesos das escalas por sua respectiva contribuição bruta e dividindo esse valor por 15.

Escala/ Sujeito	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	Total	%
<b>Mental</b>	60	45	0	0	10	15	20	20	15	0	120	65	300	0	40	30	35	90	0	0	43,3	11.8
<b>Física</b>	0	100	40	105	50	0	40	15	0	60	10	180	150	50	60	120	75	50	60	0	58,3	15.9
<b>Temporal</b>	60	120	40	90	0	50	0	60	40	30	0	50	150	140	100	160	25	195	225	120	87,8	23.9
<b>Desempenho</b>	45	40	40	60	15	40	30	100	240	200	120	255	0	90	40	80	180	60	45	60	87,0	23.7
<b>Esforço</b>	40	60	0	90	20	40	0	30	0	40	0	55	80	0	225	120	75	0	150	160	59,3	16.2
<b>Frustração</b>	0	40	80	20	20	0	90	15	20	0	0	40	10	30	0	0	75	65	75	40	31,0	8.5
<b>Ponderação</b>	13,7	26,0	13,0	18,3	5,7	7,7	8,0	15,3	21,0	22,0	16,7	43,0	46,0	20,7	31,0	34,0	31,0	30,7	37,0	25,3	<b>24.4</b>	

Tabela 4.3: UPI usando o módulo sugestor de pegada baseado no reconhecimento de objetos - magnitude de cada uma das escalas de avaliação.

A figura 4.8 ilustra o resultado da classificação bruta para a UPI usando o módulo sugestor de pegada baseado no reconhecimento de objetos. Em média, pouca demanda Mental é exigida dos usuários –11,8. Isso implica que os voluntários se sentiram

confortáveis para realizar as tarefas utilizando esse módulo e que estas não foram difíceis de serem realizadas. Poucos voluntários acharam que o módulo exigiu muito fisicamente mesmo sem que houvesse fadiga muscular; a média para demanda Física ficou em 15,9.

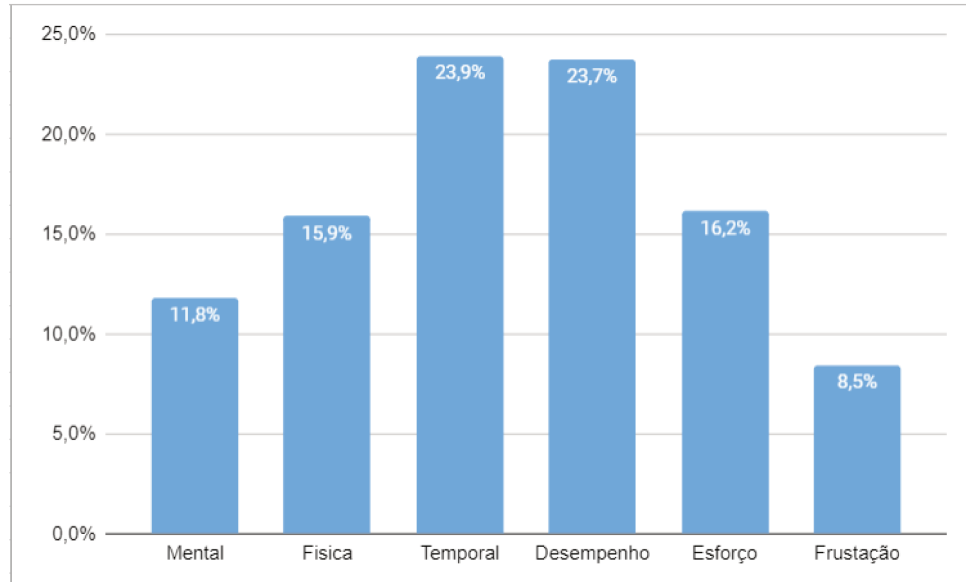


Figura 4.8: UPI usando o módulo sugestor de pegada baseado no reconhecimento de objetos: média das escalas de avaliação.

O comentário mais frequente sobre a demanda Temporal foi que, em alguns dos testes realizados, o tipo de preensão que os usuários desejavam selecionar não vinha como a primeira opção (seja porque o objeto de interesse era novo para o sistema, seja pelo fato de o banco de dados ter outras prioridades de preensão já treinadas). Consequentemente, nos casos que o usuário queira selecionar tipos de preensão que o sistema ainda não tinha conhecimento de que poderiam ser usados com o objeto de interesse, os usuários perderam algum tempo escolhendo a preensão desejada. Contudo, mesmo com essa observação, a UPI teve baixa demanda Temporal: 23.9 em média.

Quanto ao Desempenho, a maioria dos usuários tiveram alto desempenho com pouco Esforço. Isto mostra que essa interface de controle é fácil de ser utilizada mesmo quando os usuários a utilizam pela primeira vez.

Foi feita uma comparação entre a UPI e os módulos RFID e Movimento desenvolvidos numa pesquisa anterior e descrita em [38], foi feita. No Módulo de RFID, os movimentos dos dedos da prótese são definidos por leituras de marcadores de RFID que o sistema realiza e por contrações que o usuário realiza para confirmar ou para cancelar comandos. A prótese possui um leitor de RFID. Quando o usuário aproxima a prótese de um objeto que possui o marcador de RFID a leitura deste marcador é realizada e o sistema associa a leitura com uma preensão. Posteriormente, o usuário precisa confirmar se quer ou não interagir com aquele objeto, realizando uma contração de confirmação. Quando a confirmação é enviada, a prótese interage com o objeto como é possível visualizar no segundo passo da figura. Caso o usuário envie uma contração de cancelamento ao invés da contração de confirmação, a prótese continua no estado de repouso.

O módulo de movimento é uma Interface Homem-Máquina (IHM) para definir os movimentos dos dedos em uma prótese de mão utilizando uma combinação de contrações musculares (sinais EMG) e orientações. Ambos os sinais são captados pela pulseira Myo, que contém eletrodos e uma IMU que, além de captar os sinais biométricos e de movimento, os enviam para processamento. Com a fusão de sensores, o número de contrações musculares necessárias para o usuário controlar a prótese se reduz a duas – uma para a seleção de preensão (com o auxílio da orientação) e uma para cancelar as operações; o que torna essa interface diferente das próteses mioelétricas disponíveis no mercado, que exigem um tipo de contração diferente para cada tipo de preensão que a prótese de mão pode realizar. Esse fato reduz a carga cognitiva e, consequentemente, o tempo que o usuário deve treinar para controlar a interface.

O mesmo roteiro pré-definido e as mesmas condições de ambiente de teste foram feitas para a avaliação. A carga de trabalho global para os voluntários hígidos é apresentada na Tabela 4.4.

O Módulo de RFID exigiu menos carga cognitiva dos usuários; isso significa que ele foi o mais fácil de ser utilizado considerando que nenhum dos voluntários tiveram contato anteriormente com essa interface. Seguido pela interface UPI com média de carga

de trabalho de 24,45 e pelo Módulo de Movimento, que se mostrou o Módulo que mais exige esforço cognitivo dos usuários. A dificuldade em controlar o Módulo de Movimento vem principalmente das demandas Mental e Temporal exigidas por esse módulo. O módulo RFID teve uma carga de trabalho de 0.62% menor do que a UPI, mas tem a desvantagem de não ser apropriado para ser utilizado em ambientes não controlados, como o de laboratório.

Tabela 4.4: Média e Desvio Padrão (DP) da Carga de Trabalho para cada Módulo

<b>Módulo</b>	<b>Média <math>\pm</math> DP</b>
RFID	23.758 $\pm$ 12.963
Movimento	57.485 $\pm$ 8.928
UPI baseado rec objetos	24.45 $\pm$ 12.199

#### 4.1.6 Resultados Obtidos com os Testes do Voluntário Amputado

Para o voluntário amputado, ele precisaria mais que 10 minutos para controlar a UPI de maneira intuitiva – 10 minutos foi o tempo dado para que ele treinasse e se tornasse familiarizado antes de realizar a avaliação. A Figura 4.9 ilustra o resultado da classificação bruta para a UPI. Mais uma vez, o nível de Desempenho ao realizar as tarefas foi alto, o que contribuiu para que o nível de Frustração do voluntário se mantivesse baixo. Adicionalmente, o nível de Demanda Mental e Temporal se mantiveram baixos. As sugestões de preensões oferecidas pelo sistema seguem uma lista pré-programada e em algumas interações do teste, aconteceu de as primeiras sugestões do sistema não serem as interações que o usuário estava buscando, forçando-o a realizar várias contrações até encontrar a interação que ele desejava realizar. Isso contribuiu para o alto nível de Demanda Mental. Diferentemente do que foi observado para os testes com voluntários hígidos, o nível de Demanda Física foi muito alta. De acordo com a percepção do usuário, esse Módulo se mostrou desconfortável de ser usado devido ao número de contrações que ele precisou realizar para selecionar alguns tipos de preensões. Foi constatado um esforço grande por parte do usuário que demonstrou um cansaço físico evidente, devido ao fato do teste ter sido realizado com uma prótese mioelétrica.

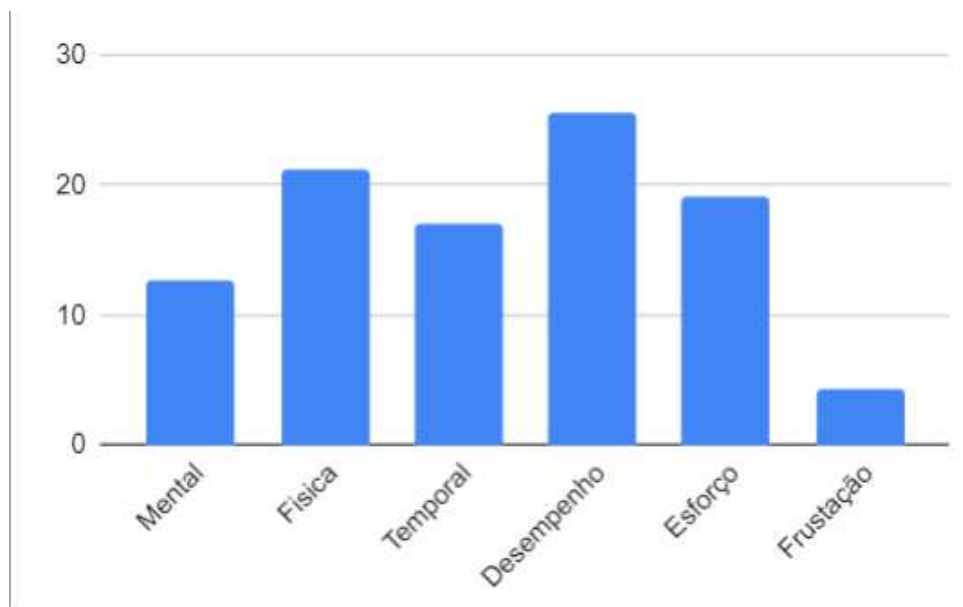


Figura 4.9: Módulo de visão: dados brutos das escalas de avaliação

## 4.2 Discussão geral

A maioria das próteses requer intenso treinamento do usuário para sua aprendizagem de controle, porém sem garantia de sucesso, levando os usuários a ficarem frustrados.

Existem muitos trabalhos na literatura que descrevem diferentes tipos de eletromiografia com interfaces homem-máquina baseadas no controle de dispositivos protéticos. No entanto, eles não avaliam a usabilidade, conforto, facilidade de controle e tempo de treinamento exigido dos usuários para o controle total da interface.

As métricas apresentadas geralmente estão relacionadas à precisão da classificação dos algoritmos que executam a interface. Além disso, existem produtos comerciais sofisticados, mãos protéticas que oferecem vantagens para amputados com um grande conjunto de preensões e suavidade de movimentos; no entanto, seu custo torna-os inacessíveis para a maioria das pessoas que poderiam beneficiar-se deles.

Neste trabalho, um protótipo de interface homem-máquina baseado em Visão Computacional foi desenvolvido e avaliado qualitativamente usando o índice de carga de tarefas da NASA para acessar a carga de trabalho necessária para controlá-los.

Este trabalho é o resultado do estudo em andamento sobre interfaces híbridas homem-máquina para mãos protéticas; a abordagem do problema de seleção de interação nesses dispositivos não se limita a agarrar. Dadas as características do problema mencionado, há contribuições quanto à definição das ações que o amputado pode realizar ao usar um dispositivo protético e também melhorar a heurística usada para realizar as ações desejadas. Na primeira questão, o sistema não limita os amputados a segurar objetos, mas permitem interações com o ambiente ao seu redor.

Os resultados mostraram que a interface UPI teve boa aceitação durante os

testes, exigindo baixo esforço cognitivo do usuário para controlar as interfaces dentro de 10 minutos após o início da utilização. Com menos esforço para selecionar e acionar uma compreensão, os usuários aprendem mais rápido como interagir com o sistema, o que os leva a uma melhor experiência.

A interface UPI foi desenvolvida em um *smartphone* dadas suas vantagens, custo e funcionalidades, permitindo a portabilidade sem uso do computador e da internet, o que acontece nas outras interfaces que usam visão computacional. Porém, o celular pode se tornar pesado dependendo do lugar onde ele for colocado no corpo; por esse motivo se recomenda, para trabalhos futuros, embutir o sistema implementado em um microcontrolador.

### 4.3 Resumo do capítulo

Este capítulo apresentou a avaliação da interface desenvolvida para controlar uma mão protética. Primeiro foram apresentados os resultados de treinamento para o módulo sugestor de pegada baseado em objetos; em seguida se apresentou os resultados de análise da base de dados que permite sugerir automaticamente o tipo de pegada e o resultado de treinamento usando as ferramentas de *TensorFlow*. Finalmente os resultados de avaliação do Índice de Carga de tarefas da NASA foram apresentados.

## Capítulo 5

### Conclusão

- Uma interface para possibilitar a interação entre o usuário e as próteses de mão que possuem funcionalidades motoras foi implementado.
- Neste trabalho a abordagem do problema de seleção de interação não se limita a agarrar, o usuário pode realizar outras ações desejadas permitindo interações com o ambiente ao seu redor.
- Se implementou um módulo que permitir fazer a previsão do tipo de pegada de duas maneiras diferentes: Por reconhecimento de objetos Por reconhecimento do tipo de pegada. O metodo de sugestão de pegada mediante reconhecimento de objetos é uma nova proposta a qual permite ao usuario customizar os tipos de pegada para cada objeto.
- Ficou evidenciado que a aprendizagem profunda baseada sistemas de visão por computador podem melhorar consideravelmente a funcionalidade de aderência das mãos mioelétricas.
- Foram avaliadas diferentes arquitecturas de classificação para obter um modelo com boa presição, tamanho e performance. Foi criada uma base de dados com imagens para a classificação de 9 tipos de pegada.
- Foi criado um algoritmo que permite biaxar imagens do buscador da google e customizar a base de dados.
- Foi avaliado qualitativamente usando o índice de carga de tarefas da NASA para acessar a carga de trabalho necessária para controlá-los.
- Foi digitalizado o formulario de avaliação NASA TLX que geralmente é feito usando papel.



- Os resultados mostraram que a interface UPI teve boa aceitação durante os testes, exigindo baixo esforço cognitivo do usuário para controlar as interfaces dentro de 10 minutos após o início da utilização.
- Com menos esforço para selecionar e acionar uma compreensão, os usuários aprendem mais rápido como interagir com o sistema, o que os leva a uma melhor experiência.

# Bibliografia

- [1] J. Fajardo, V. Ferman, A. Lemus e E. Rohmer, “An Affordable open-source multi-functional upper-limb prosthesis with intrinsic actuation”, em *Advanced Robotics and its Social Impacts (ARSO), 2017 IEEE Workshop on*, IEEE, 2017, pp. 1–6.
- [2] D. T. Andrade, A. Ishikawa, A. D. Munoz e E. Rohmer, “A hybrid approach for the actuation of upper limb prostheses based on computer vision”, em *Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), 2017 Latin American*, IEEE, 2017, pp. 1–6.
- [3] Bebionic, *The Hand*, [http://bebionic.com/the\\_hand](http://bebionic.com/the_hand), (Accessed in 29/08/2016).
- [4] T. Bionics, *Our Products: i-limb quantum*, <http://www.touchbionics.com/products>, (Accessed in 30/08/2016).
- [5] A. A. Dynamics, *Michelangelo Hand*, <http://armdynamics.com/pages/michelangelo>, (Accessed in 30/08/2016).
- [6] P. Erik Scheme MSc e P. Kevin Englehart PhD, “Electromyogram pattern recognition for control of powered upper-limb prostheses: State of the art and challenges for clinical use”, *Journal of rehabilitation research and development*, v. 48, n. 6, p. 643, 2011.
- [7] E. A. Biddiss e T. T. Chau, “Upper limb prosthesis use and abandonment: a survey of the last 25 years”, *Prosthetics and orthotics international*, v. 31, n. 3, pp. 236–257, 2007.
- [8] W. H. Organization et al., *World report on disability: World Health Organization; 2011*.
- [9] D. Cummings, “Prosthetics in the developing world: a review of the literature”, *Prosthetics and orthotics international*, v. 20, n. 1, pp. 51–60, 1996.
- [10] P. Slade, A. Akhtar, M. Nguyen e T. Bretl, “Tact: Design and performance of an open-source, affordable, myoelectric prosthetic hand”, em *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 6451–6456.
- [11] A. Akhtar, K. Y. Choi, M. Fatina, J. Cornman, E. Wu, J. Sombeck, C. Yim, P. Slade, J. Lee, J. Moore et al., “A Low-Cost, Open-Source, Compliant Hand for Enabling Sensorimotor Control for People with Transradial Amputations”,

- 
- [12] G. P. Kontoudis, M. V. Liarokapis, A. G. Zisimatos, C. I. Mavrogiannis e K. J. Kyriakopoulos, “Open-source, anthropomorphic, underactuated robot hands with a selectively lockable differential mechanism: Towards affordable prostheses”, em *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, IEEE, 2015, pp. 5857–5862.
  - [13] A. Fougner, Ø. Stavdahl, P. J. Kyberd, Y. G. Losier, P. Parker et al., “Control of upper limb prostheses: terminology and proportional myoelectric control a review”, *Transactions on Neural Systems and Rehabilitation Engineering*, v. 20, n. 5, pp. 663–677, 2012.
  - [14] J. Barnes, M. Dyson e K. Nazarpour, “Comparison of hand and forearm muscle pairs in controlling of a novel myoelectric interface”, em *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*, IEEE, 2016, pp. 002 846–002 849.
  - [15] J. Fajardo, A. Lemus e E. Rohmer, “Galileo Bionic Hand: sEMG activated approaches for a multifunction upper-limb prosthetic”, em *2015 IEEE Thirty Fifth Central American and Panama Convention (CONCAPAN XXXV)*, IEEE, 2015, pp. 1–6.
  - [16] R. LAW, *The Physical Characteristics of Humans*; 1996.
  - [17] T. RHEE e J. P. NEUMANN U.and LEWIS, “Human hand modeling from surface anatomy”, em *ACM. Proceedings of the 2006 symposium on Interactive 3D graphics and games*, 2015, pp. 27–34.
  - [18] R. Michel Thomine, Evelin Mackin e Elisabeth Schlegl, *Examination of the hand and wrist*, 5.05. 1996. endereço: [https://books.google.com.br/books?id=G1gWHR1\\_J9UC&printsec=frontcover&hl=pt-BR&source=gbs\\_atb#v=onepage&q&f=false](https://books.google.com.br/books?id=G1gWHR1_J9UC&printsec=frontcover&hl=pt-BR&source=gbs_atb#v=onepage&q&f=false) (acesso em 04/02/2020).
  - [19] G. SCHLESINGER, “Der Mechanische Aufbau der Kunstlichen Glieder. Ersatzglieder und Arbeitshilfen, part II”, 1919, pp. 27–34.
  - [20] J. Napier, “The prehensile movements of the human hand”, *The Journal of Bone e Joint Surgery*, 38b(4), 1956, pp. 902–913.
  - [21] A. Keller, T. C. e Z. V., “Studies to determinate the functional requeriments for hand and arms prostheses”, Department of Engineering, Los Angeles CA: University of California, 1947, pp. 902–913.
  - [22] I. T, B. G e A. MA, “Opposition space as a structuring concept for the analysis of skilled hand movements”, *IEEE Transactions on robotics and automation*, pp. 158–173, 1986.
  - [23] A. M. DOLLAR, “Classifying Human Hand Use and the Activities of Daily Living”, 2016, pp. 203–206.

- 
- [24] D. KUMAR, T. BASTOS e S. ARJUNAN, “Devices for Mobility and Manipulation for People with Reduced Abilities”, 2014, pp. 13–23.
  - [25] A. Fougner, E. Scheme, A. D. Chan, K. Englehart e Ø. Stavdahl, “Resolving the limb position effect in myoelectric pattern recognition”, *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v. 19, n. 6, pp. 644–651, 2011.
  - [26] E. Fujiwara, Y. T. Wu, C. Suzuki, D. Andrade, A. Ribas e E. Rohmer, “Optical Fiber Force Myography Sensor for Applications in Prosthetic Hand Control”, em *2018 IEEE Fifteenth International Workshop on Advanced Motion Control*, IEEE, 2018, pp. 1–6.
  - [27] M. S. Trachtenberg, G. Singhal, R. Kaliki, R. J. Smith e N. V. Thakor, “Radio frequency identification—an innovative solution to guide dexterous prosthetic hands”, em *Engineering in Medicine and Biology Society, EMBC, 2011 annual international conference of the IEEE*, IEEE, 2011, pp. 3511–3514.
  - [28] G. Hotson, D. P. McMullen, M. S. Fifer, M. S. Johannes, K. D. Katyal, M. P. Para, R. Armiger, W. S. Anderson, N. V. Thakor, B. A. Wester et al., “Individual finger control of a modular prosthetic limb using high-density electrocorticography in a human subject”, *Journal of neural engineering*, v. 13, n. 2, p. 026 017, 2016.
  - [29] C. M. Oppus, J. R. R. Prado, J. C. Escobar, J. A. G. Mariñas e R. S. Reyes, “Brain-computer interface and voice-controlled 3D printed prosthetic hand”, em *Region 10 Conference (TENCON), 2016 IEEE*, IEEE, 2016, pp. 2689–2693.
  - [30] D. P. McMullen, G. Hotson, K. D. Katyal, B. A. Wester, M. S. Fifer, T. G. McGee, A. Harris, M. S. Johannes, R. J. Vogelstein, A. D. Ravitz et al., “Demonstration of a semi-autonomous hybrid brain-machine interface using human intracranial EEG, eye tracking, and computer vision to control a robotic upper limb prosthetic”, *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v. 22, n. 4, pp. 784–796, 2014.
  - [31] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis e J. L. Wyatt, “One-shot learning and generation of dexterous grasps for novel objects”, *The International Journal of Robotics Research*, v. 35, n. 8, pp. 959–976, 2016.
  - [32] M. Markovic, S. Dosen, C. Cipriani, D. Popovic e D. Farina, “Stereovision and augmented reality for closed-loop control of grasping in hand prostheses”, *Journal of neural engineering*, v. 11, n. 4, p. 046 001, 2014.
  - [33] G. Ghazaei, A. Alameer, P. Degenaar, G. Morgan e K. Nazarpour, “Deep learning-based artificial vision for grasp classification in myoelectric hands”, *Journal of neural engineering*, v. 14, n. 3, p. 036 025, 2017.

- 
- [34] N. Bu, Y. Bandou, O. Fukuda, H. Okumura e K. Arai, “A semi-automatic control method for myoelectric prosthetic hand based on image information of objects”, em *Intelligent Informatics and Biomedical Sciences (ICIIBMS), 2017 International Conference on*, IEEE, 2017, pp. 23–28.
  - [35] S. Kumra e C. Kanan, “Robotic Grasp Detection using Deep Convolutional Neural Networks”, *Sensors*, 2017.
  - [36] A. A. Joseph Redmon, “Real-Time Grasp Detection Using Convolutional Neural Networks”, *ICRA*, 2015.
  - [37] J. T. Umar Asif e S. Harrer, “EnsembleNet: Improving Grasp Detection using an Ensemble of Convolutional Neural Networks”, 2018.
  - [38] D. T. Andrade, A. Ribas, G. Pereira e E. Rohmer, “Human Prosthetic Interaction: Integration of Several Techniques”, em *XIII Simposio Brasileiro de Automação Inteligente (SBAI2017)*, IEEE, 2017, pp. 1–6.
  - [39] T. M. Amund Tveit e T. B. Røst, *DeepLearningKit - an Open Source Deep Learning Framework for Apple’s iOS, OS X and tvOS developed in Metal and Swift*, Online. endereço: <https://arxiv.org/abs/1605.04614>.
  - [40] S. S. Latifi Oskouei, H. Golestani, M. Hashemi e S. Ghiasi, “CNNdroid: GPU-Accelerated Execution of Trained Deep Convolutional Neural Networks on Android”, em *Proceedings of the 2016 ACM on Multimedia Conference*, sér. MM ’16, Amsterdam, The Netherlands, 2016, pp. 1201–1205.
  - [41] S. Yao, S. Hu, Y. Zhao, A. Zhang e T. F. Abdelzaher, “DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing”, *CoRR*, v. abs/1611.01942, 2016. arXiv: [1611.01942](https://arxiv.org/abs/1611.01942). endereço: <http://arxiv.org/abs/1611.01942>.
  - [42] M. W. Moskeiwicz, F. N. Iandola e K. Keutzer, “Boda-RTC: Productive Generation of Portable, Efficient Code for Convolutional Neural Networks on Mobile Computing Platforms”, *CoRR*, v. abs/1606.00094, 2016. arXiv: [1606.00094](https://arxiv.org/abs/1606.00094). endereço: <http://arxiv.org/abs/1606.00094>.
  - [43] TensorFlow, *TensorFlow Lite guide*, Online. endereço: <https://www.tensorflow.org/lite/guide>.
  - [44] Apple, *Core ML*, Online. endereço: <https://developer.apple.com/documentation/coreml>.
  - [45] Z. Ji, “HG-Caffe: Mobile and Embedded Neural Network GPU (OpenCL) Inference Engine with FP16 Supporting”, *CoRR*, v. abs/1901.00858, 2019. arXiv: [1901.00858](https://arxiv.org/abs/1901.00858). endereço: <http://arxiv.org/abs/1901.00858>.

- 
- [46] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang e Z. Zhang, “MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems”, *CoRR*, v. abs/1512.01274, 2015. arXiv: [1512.01274](https://arxiv.org/abs/1512.01274). endereço: <http://arxiv.org/abs/1512.01274>.
  - [47] E. Rohmer, S. P. Singh e M. Freese, “V-REP: A versatile and scalable robot simulation framework”, em *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2013, pp. 1321–1326.
  - [48] M. R. Cutkosky, “On grasp choice, grasp models, and the design of hands for manufacturing tasks”, *IEEE Transactions on robotics and automation*, v. 5, n. 3, pp. 269–279, 1989.
  - [49] C. Medynski e B. Rattray, “Bebionic prosthetic design”, Myoelectric Symposium, 2011.
  - [50] J. S. Kutafina E Laukamp D, *Wearable Sensors in Medical Education: Supporting Hand Hygiene Training with a Forearm EMG*. *Stud Health Technol Inform*, <https://github.com/d4rken/myolib>, (Accessed in 30/08/2018).
  - [51] S. Shafarenka, *TinyMachine API in a nutshell*, <https://github.com/beworker/tinymachine>, (Accessed in 30/08/2018).
  - [52] N. Jermain, *Using Mobile Devices for Deep Learning*, <https://opendatascience.com/using-mobile-devices-for-deep-learning/>, (Accessed in 06/04/2019).
  - [53] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens e Z. Wojna, “Rethinking the Inception Architecture for Computer Vision”, jun. de 2016. DOI: [10.1109/CVPR.2016.308](https://doi.org/10.1109/CVPR.2016.308).
  - [54] A. G. Howard, M. Z. B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto e H. Adam, *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, <https://arxiv.org/pdf/1704.04861.pdf>, (Accessed in 07/12/2019).
  - [55] H. Zulkifli, *Understanding Learning Rates and How It Improves Performance in Deep Learning*, <https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10>, (Accessed in 03/02/2020).
  - [56] scikit-learn, *A decision tree classifier*, <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>, (Accessed in 30/08/2020).
  - [57] —, *Nearest Neighbors*, <https://scikit-learn.org/stable/modules/neighbors.html>, (Accessed in 30/08/2020).
  - [58] —, *C-Support Vector Classification*, <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>, (Accessed in 30/08/2020).
  - [59] —, *A decision tree classifier*, [VotingClassifier](#), (Accessed in 30/08/2020).

- [60] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang e C. Liu, “A Survey on Deep Transfer Learning”, *CoRR*, v. abs/1808.01974, 2018. arXiv: [1808.01974](https://arxiv.org/abs/1808.01974). endereço: <http://arxiv.org/abs/1808.01974>.
- [61] D. Amparo, *Formulario de avaliação da NASA TLX*, Online. endereço: <https://form.jotform.com/201148780701046>.
- [62] U. Asif, J. Tang e S. Harrer, “GraspNet: An Efficient Convolutional Neural Network for Real-time Grasp Detection for Low-powered Devices”, em *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, International Joint Conferences on Artificial Intelligence Organization, jul. de 2018, pp. 4875–4882. DOI: [10.24963/ijcai.2018/677](https://doi.org/10.24963/ijcai.2018/677). endereço: <https://doi.org/10.24963/ijcai.2018/677>.

## Apêndice A

### Aprovação do Comitê de Ética



**TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO – Linha de pesquisa IV (Anexo VI)**

**Projeto XTReMe: Experiências de Tecnologias para Reabilitação em Medicina**  
**Subprojeto: Estudo e desenvolvimento de próteses de mão robóticas**  
**Pesquisadores responsáveis: Eric Rohmer**  
**CAAE: 58592916.9.1001.5404**

Você está sendo convidado a participar como voluntário da pesquisa “Estudo e desenvolvimento de próteses de mão robóticas”. Este documento, chamado Termo de Consentimento Livre e Esclarecido, visa assegurar seus direitos como participante e é elaborado em duas vias, uma que deverá ficar com você e outra com o pesquisador.

Por favor, leia com atenção e calma, aproveitando para esclarecer suas dúvidas. Se houver perguntas antes ou mesmo depois de assiná-lo, você poderá esclarecê-las com o pesquisador. Se preferir, pode levar este Termo para casa e consultar seus familiares ou outras pessoas antes de decidir participar. Não haverá nenhum tipo de penalização ou prejuízo se você não aceitar participar ou retirar sua autorização em qualquer momento.

**Justificativa e objetivos:**

Próteses e órteses inteligentes de mão de alta tecnologia oferecem a possibilidade de escolher vários tipos de ações para os dedos, indo além do simples abrir e fechar de mão. Por exemplo, algumas próteses importadas oferecem dezenas de ações possíveis como apontar para digitar, segurar objetos pequenos com dois dedos ou até a pegada para utilizar um mouse. Porém, possuem um custo extremamente alto, chegando a milhares de dólares ou podem ainda nem estar disponíveis para o mercado. De fato, deixando o acesso à essa tecnologia indisponível a centenas de brasileiros.

Adicionalmente, com as próteses inteligentes importadas, a seleção da ação desejada pelo usuário é realizada por muitas contrações musculares diferentes e conseguir a escolha certa necessita de um treinamento cotidiano intensivo e muito fatigante com a prótese para um resultado que pode ser frustrante.

Visando reverter esse cenário, procura-se oferecer um produto brasileiro, de fácil manutenção, personalizado e semelhante em funcionalidades, mas com uma seleção de movimentos mais robusta e intuitiva por um preço muito mais acessível (usando a tecnologia de impressão 3D e componente de prateleiras).

Sendo assim, o objetivo desta pesquisa é testar os protótipos com o público alvo e refinar as técnicas de seleções das ações, bem como a confiabilidade e conforto de uso da prótese.

Neste documento, prótese se refere tanto a próteses quanto a órteses inteligentes (i.e. que tem interações que se adaptam às necessidades do usuário). Além disso, interfaces (homem-máquina) são definidas como um conjunto de sensores e as interações do usuário com os estes sensores.

As interfaces que selecionam as ações dos dedos consideradas nesta pesquisa são baseadas no uso combinado de eletromiografia (EMG) para avisar a prótese do começo do processo de seleção da ação. Em seguida, seleciona-se a ação por comando de voz, aplicação de celular, movimento, chips fixos em objetos ou outros. Finalmente, o EMG define o momento de realização da ação.

**Procedimentos:**

Rubrica do pesquisador: \_\_\_\_\_ Rubrica do participante: \_\_\_\_\_

Participando do estudo você está sendo convidado a utilizar uma interface para o controle da prótese de mão no simulador através de uma ou várias interfaces de interação, tais como comandos por voz, sensores de gestos e eletromiografia. Além disso, você também poderá ser convidado a utilizar uma prótese robótica de mão real e realizar as mesmas interações descritas anteriormente.

Em ambos os casos apresentados, um terceiro estará sempre acompanhando o procedimento podendo, eventualmente, pará-lo, se necessário. Você poderá ser convidado mais de uma vez para realizar os testes a fim de analisarmos a sua evolução no uso das interfaces e sua opinião sobre a usabilidade e ergonomia com relação a todas as interfaces apresentadas.

O local de realização dos testes será dentro do Laboratório de Computação e Automação da Faculdade de Engenharia Elétrica e Computação da UNICAMP. O tempo total estimado para realização dos testes não deve ser superior a 2 (duas) horas. Caso os resultados forem conclusivos e caso você tenha interesse e se sinta confortável poderemos pedir para que a prótese seja utilizada por você durante uma semana durante o seu cotidiano. Ao final da semana, você dará retorno sobre a sua experiência de uso.

Os procedimentos que serão realizados são descritos detalhadamente a seguir:

1. Demonstração: chegando ao local do teste, você será apresentado à prótese de mão (virtual ou real) e os sensores a serem utilizados. Os detalhes da prótese serão mostrados e o pesquisador responsável irá explicar como ela pode ser manipulada através das demonstrações. O tempo estimado para essa etapa é de 10 a 20 minutos, dependendo do número de interfaces que irão ser demonstradas.
2. Posicionamento da prótese no usuário: nesta etapa, o membro da equipe que fez a demonstração irá retirar os sensores que estava utilizando para fazer a demonstração e irá colocá-los em você. Se a prótese real estiver disponível, esta será usada ao invés da prótese virtual. Você será perguntado se se sente confortável ao utilizar o sistema e, caso não esteja, pode retirar a qualquer momento. Tempo estimado para essa etapa é de 5 a 10 minutos.
3. Uso da prótese via interface homem-máquina: você poderá ser convidado a: a) Utilizar uma interface para controlar a prótese (até 60 minutos de duração); b) Utilizar mais uma interface para controlar a prótese ou repetir o teste anterior com a mesma interface (até 30 minutos de duração); c) Caso esteja testando com a prótese real, apenas colocar a prótese para descrever se peso, tamanho e outras características do produto estão adequadas e sua opinião sobre a estética do mesmo (até 30 minutos de duração); O tipo de interface utilizada será definido pela equipe. Entretanto, você poderá desistir a qualquer momento do teste se, eventualmente, não se sentir confortável com as decisões da equipe.
4. Calibração: neste teste você estará com a prótese e tentará se familiarizar com os tipos de comandos que devem ser enviados a prótese. Você tentará repetir as interações demonstrados pela equipe para que o sistema se adeque à sua forma de interação.
5. Práticas com a interface: nesta etapa você tentará controlar os diferentes tipos de ações que são possíveis através do sistema. Você, por exemplo, tentará manusear um mouse, pegar uma caneta, pegar uma bola, tomar um copo com água, segurar

Rubrica do pesquisador: \_\_\_\_\_ Rubrica do participante: \_\_\_\_\_

uma maleta (leve e sem objetos dentro), apontar para algum objeto, fechar as mãos, segurar uma serrinha de unha, utilizar um borrifador de água, manusear talheres, abrir a mão, relaxar a mão, entre outros eventos do cotidiano.

6. Entrevista: você é a parte essencial no desenvolvimento desse projeto. Dessa forma, queremos saber a sua opinião sobre a sua experiência em utilizar a prótese de mão. Será realizada uma entrevista com áudio e vídeo onde será perguntado o que você achou da sua experiência, quais suas sugestões, quais as suas críticas e quais as suas expectativas. A entrevista será realizada durante o treinamento do uso da prótese.

**Desconfortos e riscos:**

Os riscos possíveis relacionados aos procedimentos descritos acima incluem desconforto muscular durante o uso da prótese.

**Benefícios:**

A sua participação nesta pesquisa não implicará em nenhum benefício pessoal e não é obrigatória.

**Acompanhamento e assistência:**

Caso queira, você poderá desistir da sua participação a qualquer momento, sem que isso lhe cause nenhum prejuízo. Você será acompanhado e assistido pelo pesquisador responsável e a sua equipe durante esses procedimentos, podendo fazer perguntas sobre qualquer dúvida que apareça durante todo o estudo.

**Sigilo e privacidade:**

Os dados coletados estarão sob o resguardo científico e o sigilo profissional, e contribuirão para o alcance dos objetivos deste trabalho e para posteriores publicações dos dados.

**Ressarcimento e indenização:**

Você não receberá nenhum pagamento por sua participação nesta pesquisa, mas caso venha a ter despesas de transporte ou alimentação para participar na pesquisa, será ressarcido.

**Métodos alternativos:**

Não há métodos alternativos.

**Contato:**

Para quaisquer dúvidas, você pode contatar os pesquisadores responsáveis: Dr. Eric Rohmer (tel. (19)352-10247, email: [eric@dca.fee.unicamp.br](mailto:eric@dca.fee.unicamp.br), endereço: Faculdade de Engenharia Elétrica e Computação - Av. Albert Einstein, 400, UNICAMP, CEP: 13083-859, Cidade Universitária, Campinas, SP).

Em caso de denúncias ou reclamações sobre sua participação e sobre questões éticas do estudo, você poderá entrar em contato com a secretaria do Comitê de Ética em Pesquisa (CEP) da UNICAMP das 08:30hs às 11:30hs e das 13:00hs às 17:00hs na Rua: Tessália Vieira de Camargo, 126; CEP 13083-887 Campinas – SP; telefone (19) 3521-8936 ou (19) 3521-7187; e-mail: [cep@fcm.unicamp.br](mailto:cep@fcm.unicamp.br).

Rubrica do pesquisador: \_\_\_\_\_ Rubrica do participante: \_\_\_\_\_

**O Comitê de Ética em Pesquisa (CEP).**

O papel do CEP é avaliar e acompanhar os aspectos éticos de todas as pesquisas envolvendo seres humanos. A Comissão Nacional de Ética em Pesquisa (CONEP), tem por objetivo desenvolver a regulamentação sobre proteção dos seres humanos envolvidos nas pesquisas. Desempenha um papel coordenador da rede de Comitês de Ética em Pesquisa (CEPs) das instituições, além de assumir a função de órgão consultor na área de ética em pesquisas

**Consentimento livre e esclarecido:**

Após ter recebido esclarecimentos sobre a natureza da pesquisa, seus objetivos, métodos, benefícios previstos, potenciais riscos e o incômodo que esta possa acarretar, aceito participar e declaro estar recebendo uma via original deste documento assinada pelo pesquisador e por mim, tendo todas as folhas por nós rubricadas:

Nome do (a) participante: \_\_\_\_\_

Contato telefônico: \_\_\_\_\_

e-mail (opcional): \_\_\_\_\_

\_\_\_\_\_  
Data: \_\_\_\_/\_\_\_\_/\_\_\_\_.  
(Assinatura do participante ou nome e assinatura do seu RESPONSÁVEL LEGAL)

**Responsabilidade do Pesquisador:**

Asseguro ter cumprido as exigências da resolução 466/2012 CNS/MS e complementares na elaboração do protocolo e na obtenção deste Termo de Consentimento Livre e Esclarecido. Asseguro, também, ter explicado e fornecido uma via deste documento ao participante. Informo que o estudo foi aprovado pelo CEP perante o qual o projeto foi apresentado. Comprometo-me a utilizar o material e os dados obtidos nesta pesquisa exclusivamente para as finalidades previstas neste documento ou conforme o consentimento dado pelo participante.

\_\_\_\_\_  
Data: \_\_\_\_/\_\_\_\_/\_\_\_\_.  
(Assinatura do pesquisador)

Rubrica do pesquisador: \_\_\_\_\_ Rubrica do participante: \_\_\_\_\_

# Apêndice B

## Código fonte

- Aplicação no Android: <https://gitlab.com/jadiazm2012/smarthand>
- Programa que permite a análise de bases de dados de imagens: [https://colab.research.google.com/drive/1xPZ8DjMXqUBQeAQSSEuIRNOF\\_A\\_FsurN#scrollTo=ZTRvmK\\_GB\\_Gx](https://colab.research.google.com/drive/1xPZ8DjMXqUBQeAQSSEuIRNOF_A_FsurN#scrollTo=ZTRvmK_GB_Gx)
- Software de treinamento para o modulo sugestor de pegada baseado no reconhecimento de pegada: <https://colab.research.google.com/drive/1xs4QGC9h0EPIsoU6Ge25UN2YkkVVZ-SI#scrollTo=WBtFK1h08Ks0>

## Apêndice C

### Formulário NASA-TLX

Url: <https://form.jotform.com/201148780701046>