UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

AVALIAÇÃO DAS TÉCNICAS DE SEGMENTAÇÃO, MODELAGEM E CLASSIFICAÇÃO PARA O RECONHECIMENTO AUTOMÁTICO DE GESTOS E PROPOSTA DE UMA SOLUÇÃO PARA CLASSIFICAR GESTOS DA LIBRAS EM TEMPO REAL

MAURO DOS SANTOS ANJO

ORIENTADOR: PROF. DR. EDNALDO BRIGANTE PIZZOLATO

São Carlos - SP
Setembro/2012

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

A599at

Anjo, Mauro dos Santos.
Avaliação das técnicas de segmentação, modelagem e classificação para o reconhecimento automático de gestos e proposta de uma solução para classificar gestos da libras em tempo real / Mauro dos Santos Anjo. -- São Carlos : UFSCar, 2013.
143 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2012.

1. Processamento de imagens. 2. Reconhecimento de gestos. 3. Linguagem brasileira por sinais. 4. Reconhecimento de padrões. I. Título.

CDD: 006.42 (20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"Avaliação das Técnicas de Segmentação, Modelagem e Classificação para o Reconhecimento Automático de Gestos e Proposta de uma Solução para Classificar Gestos da Libras em Tempo Real"

Mauro dos Santos Anjo

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação

Membros da Banca:

Prof. Dr. Ednaldo Brigante Pizzolato
(Orientador - DC/UFSCar)

Prof. Dr. Alexandre Luis Magalhães Levada
(DC/UFSCar)

Prof. Dr. Siome Klein Goldenstein
(UNICAMP)

São Carlos
Outubro/2012

RESUMO

Interfaces multimodais estão cada vez mais populares e buscam a interação natural como recurso para enriquecer a experiência do usuário. Dentre as formas de interação natural, estão a fala e os gestos. O reconhecimento de fala já está presente em nosso dia a dia em variadas aplicações, porém o reconhecimento de gestos apareceu recentemente como uma nova forma de interação.

A Linguagem Brasileira de Sinais (Libras) foi recentemente reconhecida como meio de comunicação e expressão através da Lei N˚10.436 de 24/04/2002, e também foi incluída como disciplina obrigatória em cursos de formação de professores e optativa em cursos de graduação através do Decreto N˚5.626 de 22/12/2005.

Neste contexto, esta dissertação apresenta um estudo sobre todas as etapas necessárias para a construção de um sistema para reconhecimento de Gestos Estáticos e Dinâmicos da Libras, sendo estas: Segmentação; Modelagem e Identificação; e Reconhecimento.

Resultados e soluções propostas serão apresentados para cada uma destas etapas, e o sistema será avaliado no reconhecimento em tempo real utilizando um conjunto finito de gestos estáticos e dinâmicos.

Todas as soluções apresentadas nesta dissertação foram encapsuladas no Software GestureUI, que tem por objetivo simplificar as pesquisas na área de reconhecimento de gestos permitindo a comunicação com interfaces multimodais através de um protocolo TCP/IP.

Palavras-chave: Reconhecimento de gestos, Linguagem Brasileira de Sinais, Reconhecimento de Padrões, Processamento de Imagens.

ABSTRACT

Multimodal interfaces are becoming popular and trying to enhance user experience through the use of natural forms of interaction. Among these forms we have speech and gestures inputs. Speech recognition is already a common feature in our daily basis but gesture recognition has just now being widely used as a new form of interaction.

The Brazilian Sign Language (Libras) was recently recognized as a legal way of communication since the Brazilian Government enacted the law N°10.436 on 04/24/2002, and also has recently became an obligatory subject in teachers education and an elective subject in undergraduate courses through the enactment N°5.626 on 12/22/2005.

In this context, this dissertation presents a study of all the steps that are necessary to achieve a complete system to recognize Static and Dynamic gestures of Libras, being these steps: Segmentation; Modeling and Interpretation; and Classification.

Results and proposed solutions will be presented for each one of these steps, and the system will be evaluated in the task of real-time recognition of static and dyamic gestures within a finite set of Libras gestures.

All the solutions presented in this dissertation were embedded in the software GestureUI, in which the main goal is to simplify the research in the field of gesture recognition allowing the communication with multimodal interfaces through a TCP/IP protocol.

Keywords: Gesture Recognition, Brazilian Sign Language, Pattern Recognition, Image Processing.

LISTA DE FIGURAS

Figura 1 - Etapas do sistema de reconhecimento de gestos. 5

Figura 2 - Kinect para XBOX 360 e sua segmentação de profundidade 7

Figura 3 - Ilustração conceitual da funcionalidade de interação com mapas através de gestos e realidade aumentada 7

Figura 4 - Exemplo de resultado da subtração simples, sendo a) o modelo de fundo, b) a imagem atual e c) o resultado do algoritmo 12

Figura 5 - Exemplo de correção de White Balance 18

Figura 6 - Exemplo de resultado da extração de regiões que contém cor de pele, sendo as imagens à esquerda as originais e as à direita o resultado obtido 25

Figura 7 - Exemplo de aplicação do operador de Sobel 27

Figura 8 - Exemplo de resultado de aplicação do algoritmo de Canny 29

Figura 9 - Exemplo de um Histograma para os canais R, G e B da imagem no canto inferior esquerdo 30

Figura 10 - a) Interior do Kinect mostrando os componentes; b) Dois exemplos de projeção do padrão gerado pelo projetor infravermelho do Kinect. 33

Figura 11 - Resultados obtidos com o experimento 1. a) modelo de fundo; b) imagem corrente; c) resultado final. 36

Figura 12 - Resultados obtidos com o experimento 2. a) modelo de fundo; b) imagem corrente; c) resultado final. 37

Figura 13 - Resultado obtidos pelo algoritmo com maior custo benefício *Tempo de processamento* X *Precisão.* 39

Figura 14 - Exemplos de replicação da posição humana em um robô, utilizando marcadores coloridos. 45

Figura 15 - Exemplo de Pictorial Structures 47

Figura 16 - Exemplo de Segmentação e decomposição da silhueta em partes. 49

Figura 17 - Exemplo de rotulação das partes segmentadas segundo o modelo hierárquico 50

Figura 18 - Exemplo de segmentação e identificação de partes visíveis e ocultas em imagem de baixa resolução e fundo complexo. 51

Figura 19 - Exemplo de identificação de partes sobrepostas ou não e correspondente modelagem em 3D. 54

Figura 20 - Problemas de estimação com o Kinect SDK. a) Mapa de profundidade; b) Estimação das partes do corpo; c) Imagem Colorida. 56

Figura 21 - Problemas de estimação utilizando o PrimeSense Nite. 57

Figura 22 - Problemas no rastreamento da face utilizando Camshift. a) estimação correta; b), c) e d) falhas de estimação após movimentar a(s) mão(s) sobre a face 59

Figura 23 - Máscara de pixels do usuário (amarelo) sem as partes móveis (branco) que ultrapassaram a barreira virtual. 61

Figura 24 - Gráfico da variação da coordenada Z do Centro de Massa (CM) do usuário com e sem partes móveis. 61

Figura 25 - Resultado da segmentação das mãos do usuário. Em azul a mão direita e em vermelho a mão esquerda. 63

Figura 26 - Pixels em azul correspondem a: a) área do blob; b) área convexa do blob. 65

Figura 27 - Detecção de Finger Tips. À esquerda exemplo de detecção com sucesso, à direita ampliação com detalhamento do K-Curvature 66

Figura 28 - Exemplo de problema de diferenciação do Descritor de Fourier invariável a rotação |FD|. Estas duas formas são erroneamente classificadas como similares. 68

Figura 29 - Gestos estáticos utilizados para avaliar o desempenho dos Descritores de Fourier. 69

Figura 30 - Gráfico de reconhecimento de cada gesto em cada experimento. 70

Figura 31 - Gráfico de média de reconhecimento por experimento. 71

Figura 32 - Exemplos de gestos efetuados com a luva verde. 76

Figura 33 - Exemplo do módulo de segmentação e reconhecimento de gestos utilizando luvas coloridas 76

Figura 34 - Exemplo de Configuração de Mãos para *Trabalhar* e *Xícara.* 81

Figura 35 - Exemplo de Localização da mão em *Brincar* e *Pensar.* 82

Figura 36 - Exemplo de Orientação em *Ajudar* e *Levantar.* 82

Figura 37 - Exemplos de Expressões em *Alegre* e *Moto*. 83

Figura 38 - Exemplos de posturas estáticas da Libras 86

Figura 39 - Exemplos de tipos de Segmentação do Gesture UI. a) Luvas coloridas; b) Câmera de profundidade. 100

Figura 40 - Exemplos de telas de treinamento de gestos. a) Treinamento da HMM; b) Avaliação de agrupamentos de HMMs previamente treinadas. 100

Figura 41 - Exemplo de forma de interação utilizando o Mint e o GestureUI. a) Interface web sendo controlada; b) gestos para navegação nos itens da interface.....104

Figura 42 - Aplicação de Realidade Aumentada (RA) utilizando Mint e GestureUI. a) Interface web com objetos disponíveis; b) cena de RA com os objetos selecionados. 104

Figura 43 - Gestos estáticos selecionados para testes de reconhecimento em tempo real. 108

Figura 44 - Passo a passo da preparação da imagem para processamento pela rede neural. a) Imagem original; b) Redimensionamento; c) Threshold e centralização. 108

Figura 45 - Exemplo dos problemas encontrados com a presença do braço na região de interesse a) e c) e a solução utilizando o algoritmo ARHCA. 109

Figura 46 - Arquitetura da MLP para reconhecimento dos dois grupos de gestos estáticos: (A, E, I, O, U) e (B, C, F, L, V). 112

Figura 47 - Gráfico das taxas de reconhecimento de cada um dos gestos dinâmicos para os conjuntos de Treinamento e Teste utilizando 16 clusters no K-Means. 119

Figura 48 - Gráfico das taxas de reconhecimento de cada um dos gestos dinâmicos para os conjuntos de Treinamento e Teste utilizando 32 clusters no K-Means. 120

Figura 49 - Gráfico das taxas de reconhecimento dos gestos dinâmicos utilizando somente a posição 3D e Fourier Descriptors no vetor de características. 121

Figura 50 - Gráfico das taxas de reconhecimento dos gestos dinâmicos utilizando descritores geométricos e a posição 3D no vetor de características, e 16 clusters no K-Means.....122

Figura 51 - Gráfico das taxas de reconhecimento dos 14 gestos dinâmicos utilizando descritores geométricos e a posição 3D no vetor de características, e 32 clusters no K-Means.....122

Figura 52 - Histograma de tempos de execução para Segmentação, Análise e Reconhecimento de Gestos Estáticos.....124

Figura 53 - Histograma de tempos de reconhecimento de gestos dinâmicos na fase final.....126

Figura 54 - Histograma de tempo de reconhecimento com 14 Gestos. 127

LISTA DE TABELAS

Tabela I - Valores ideais para o algoritmo de detecção de highlights, segundo Duque [DUQUE et al., 2005].....16

Tabela II - Tempos de execução do algoritmo de estimação de profundidade proposto por Hirschmüller. 38

Tabela III - Experimentos para avaliação da classificação de gestos com Descritores de Fourier. 70

Tabela IV - Exemplos da estrutura Tópico-Comentário da Libras. 84

Tabela V - Exemplos de possíveis arquiteturas para a HMM. 96

Tabela VI - Taxas de aprendizagem e Termum Momentum de cada uma das camadas do classificador MLP. 111

Tabela VII - Taxas de reconhecimento para os gestos estáticos A, E, I, O, U com e sem ARHCA. 113

Tabela VIII - Taxas de reconhecimento para os gestos estáticos B, C, L, F, V com e sem ARHCA. 113

Tabela IX - Relação de gestos de palavras da Libras escolhidos para avaliação do sistema de reconhecimento.....116

Tabela X - Relação dos sete gestos adicionais da Libras para avaliação do sistema de reconhecimento. 117

Tabela XI - Matriz de confusão na classificação dos gestos dinâmicos utilizando no vetor de características apenas Descritores de Fourier e a Posição 3D. 121

Tabela XII - Média e Desvio Padrão dos tempos de execução das etapas de Segmentação, Análise e Reconhecimento de Gestos Estáticos. 125

Tabela XIII - Média e Desvio Padrão dos tempos de reconhecimento de gestos dinâmicos da Libras. 126

SUMÁRIO

CAPÍTULO 1 - INTRODUÇÃO....4

1.1 Apresentação 4
1.2 Motivação e Relevância 6
1.3 Objetivo 8
1.4 Organização do Trabalho 8

CAPÍTULO 2 - SEGMENTAÇÃO....10

2.1 Considerações Iniciais....10
2.2 Subtração de Fundo (Background Subtraction) 11
2.2.1 Subtração simples....12
2.2.2 Running Gaussian Average....12
2.2.3 Temporal Median Filter....13
2.2.4 Mistura de Gaussianas (Mixture of Gaussians)....14
2.2.5 Principais problemas da detecção de movimento por Background Subtraction 15
2.2.5.1 Alterações de Brilho (Highlights) 15
2.2.5.2 Sombras....16
2.2.5.3 Fantasmas (Ghosts)....17
2.2.5.4 Câmeras de configuração automática....17
2.2.5.5 Regra de decisão para atualizar modelo de fundo 19
2.2.6 Técnicas de Pós-Processamento....19
2.2.6.1 Remoção de ruídos 19
2.2.6.2 Identificação de Blobs 20
2.2.6.3 Area thresholding 20
2.2.6.4 Morphological closing....20
2.2.6.5 Saliency test....21
2.2.6.6 Optical flow test....22
2.3 Segmentação por cor de pele 22
2.3.1 Principais fatores limitantes na segmentação por cor de pele....22

2.3.2 Algoritmos de detecção de cor de pele 23
2.3.2.1 Threshold explícito no espaço de cor 24
2.3.2.2 Gaussian mixture models (GMM) 24
2.3.2.3 Bayesian network (BN) 25
2.4 Edge Detection 26
2.4.1 Sobel 26
2.4.2 Canny 27
2.5 Edge Orientation Histogram 29
2.6 Segmentação de profundidade utilizando visão estéreo 31
2.7 Segmentação de profundidade utilizando infravermelho 31
2.8 Decisões de projeto e Resultados 34
2.8.1 Detecção de Movimento no Espectro Visual 35
2.8.2 Visão Estéreo no Espectro Visual 38
2.8.3 Solução Proposta: Virtual Wall 39

CAPÍTULO 3 - MODELAGEM E TRACKING DE PARTES DO CORPO HUMANO 41

3.1 Considerações Iniciais 41
3.2 Características dos métodos de modelagem e tracking 42
3.3 Marcadores Coloridos ou Refletivos 44
3.4 Sensores de posição e acelerômetros 45
3.5 Pictorial Structures 46
3.6 Modelagem através da silhueta 48
3.7 Modelagem utilizando imagens de profundidade 52
3.8 Decisões de Projeto e Resultados 55
3.8.1 Kinect SDK e PrimeSense Nite 55
3.8.2 Solução Proposta no escopo da Libras 57
3.8.3 Virtual Wall Dinâmico 58
3.8.3.1 Reconhecimento e rastreamento da face 58
3.8.3.2 Centro de Massa do usuário 60
3.8.4 Segmentação das mãos 62
3.8.5 Extração de Características 63
3.8.5.1 Descritores Geométricos 64
3.8.5.2 Descritor de trajetória 66
3.8.5.3 Descritores de Fourier 66

CAPÍTULO 4 - RECONHECIMENTO DE GESTOS.....72

4.1 Considerações Iniciais.....72
4.2 Projetos para Reconhecimento de Gestos Estáticos e Dinâmicos.....73
4.2.1 Reconhecimento de Gestos Estáticos.....73
4.2.2 Reconhecimento de Gestos Dinâmicos.....77
4.3 Projetos de reconhecimento de linguagens de sinais 78
4.4 Língua Brasileira de Sinais e seus desafios para segmentação e reconhecimento 80
4.4.1 Língua Brasileira de Sinais (Libras).....80
4.4.2 Parâmetros para formação de um sinal em Libras.....81
4.4.3 Estrutura Linguística da Libras.....84
4.4.4 Desafios para a segmentação e reconhecimento da Libras.....85
4.5 Decisões de Projeto e Resultados 87
4.6 K-Means.....87
4.6.1 Algoritmo de Lloyd.....88
4.6.2 Algoritmo Binary Split.....89
4.6.3 Normalização de Dados: Z-Scores.....90
4.7 Hidden Markov Models (HMM).....91
4.7.1 Problema de Avaliação.....92
4.7.2 Problema de Estimação ou Decodificação.....93
4.7.3 Problema de Treinamento.....94
4.7.4 Arquiteturas da HMM.....96

CAPÍTULO 5 - SOFTWARE GESTURE USER INTERFACE (GESTUREUI).....97

5.1 Considerações Iniciais.....97
5.2 Metodologia de desenvolvimento.....98
5.3 Funcionalidades.....99
5.3.1 Segmentação.....99
5.3.2 Reconhecimento de Gestos.....100
5.3.3 Coleta de Amostras.....101
5.3.3.1 Gestos Estáticos.....101
5.3.3.2 Gestos Dinâmicos.....102
5.3.4 Protocolo de Comunicação.....102
5.4 Aplicações.....103

CAPÍTULO 6 - RESULTADOS GERAIS.....106

6.1 Considerações Inicias 106
6.2 Reconhecimento de gestos estáticos em tempo real 107
6.2.1 Segmentação e Conjunto de Amostras 107
6.2.2 Classificador Multi-Layer Perceptron.....110
6.3 Reconhecimento de gestos dinâmicos em tempo real 113
6.3.1 Definição de escopo e objetivo.....114
6.3.2 Metodologia de Testes 117
6.3.3 Resultados de Reconhecimento de palavras da Libras 119
6.4 Avaliação da execução em tempo real.....123

CAPÍTULO 7 - CONCLUSÕES.....128

7.1 Contribuições 130
7.2 Trabalhos Futuros 132

REFERÊNCIAS.....133

GLOSSÁRIO.....142

Capítulo 1

INTRODUÇÃO

Este capítulo apresenta uma visão geral da área de pesquisa, assim como uma breve descrição dos objetivos e motivação do projeto.

1.1 Apresentação

Reconhecimento de Gestos é uma área de pesquisa que vem sendo abordada com frequência no meio acadêmico. Por ser uma tarefa de alta complexidade, ela envolve diversos tópicos como Interação Humano-Computador (IHC), Visão Computacional, Reconhecimento de Padrões, entre outros.

A possibilidade de se interagir com o computador por meio de gestos pode enriquecer substancialmente as interfaces IHC, aumentando a variabilidade de tarefas que podem ser executadas simultaneamente, para assim, unindo-se aos periféricos hoje existentes e ao reconhecimento de voz, verdadeiramente, construírem-se interfaces multimodais.

O gesto não só enriqueceria interfaces multimodais como também facilitaria a comunicação, aprendizado e interação de pessoas com deficiência auditiva ou mudas, pois elas utilizam como forma de comunicação uma língua de sinais, que no Brasil, é denominada Língua Brasileira de Sinais (Libras). Esta língua é composta por diversos gestos, que são utilizados como unidade básica de construção de sentenças ou expressões da linguagem.

A grande maioria dos trabalhos desenvolvidos até hoje visa o reconhecimento de interações gestuais através de posturas estáticas, ou através de gestos que são

geralmente modelados como posturas estáticas ao longo do tempo, com uma trajetória e velocidade.

Porém, o reconhecimento de gestos de uma língua de sinais como a Libras, exige a interpretação e modelagem de um contexto da língua além de outras variáveis que devem ser levadas em consideração, como por exemplo, identificação dos membros do corpo, posicionamento relativo entre estes, e muitas vezes até expressões faciais.

Neste contexto, um sistema de reconhecimento de gestos para a língua de sinais pode ser subdividido em quatro fases, como indicado na Figura 1.

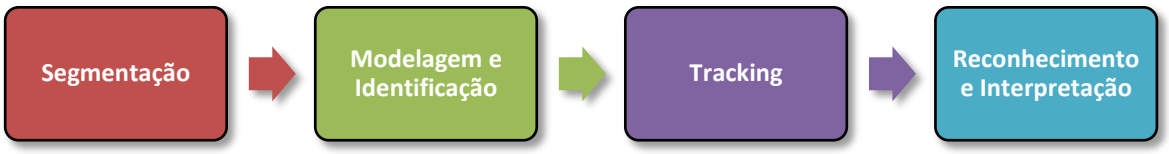


Figura 1 - Etapas do sistema de reconhecimento de gestos.

Segmentação corresponde à fase de modelagem da cena no campo de visão da câmera, buscando separar as partes do corpo humano dos objetos ou regiões pertencentes ao fundo.

Modelagem e Identificação é a fase em que os blobs (regiões na imagem que são agrupamentos de pixels segmentados a partir de uma característica comum, como cor de pele, por exemplo) obtidos na fase anterior são agrupados e identificados de acordo com um modelo pré-determinado de partes do corpo humano.

Tracking corresponde à tarefa de rastreamento de movimento de regiões de interesse detectadas através de modelos estatísticos para previsão de próximas posições. É uma fase opcional e depende muito da definição do método utilizado na fase de *Modelagem e Identificação*. Esta tarefa consiste na observação da segmentação de duas imagens consecutivas (em uma sequência de captura ou vídeo) para correlacionar as regiões rotuladas nas mesmas, estimando assim a possível movimentação de blobs.

Reconhecimento e Interpretação é a última etapa que consiste em fornecer um resultado final para o sistema. Este resultado é o gesto da língua de sinais que foi

reconhecido. Para se obter tal resultado, várias técnicas de reconhecimento podem ser utilizadas, como por exemplo, Redes Neurais. Uma gramática também é necessária para definir a língua de sinais como uma linguagem sensível ao contexto, podendo assim permitir a interação natural através da Libras.

Neste trabalho, todas estas etapas foram abordadas com foco no reconhecimento de algumas palavras da Libras em tempo real.

1.2 Motivação e Relevância

A comunicação através dos gestos é uma das formas mais naturais de interação dos humanos. Reconhecer e interpretar esta forma de interação mantendo seu caráter natural é o foco principal desta área de pesquisa.

A naturalidade da comunicação através dos gestos deve ser obtida sem utilização de quaisquer aparatos extra, que são geralmente utilizados para facilitar a tarefa de segmentação, modelagem e reconhecimento.

Eliminar marcadores coloridos, luvas de sensoriamento 3D, giroscópios, ou qualquer outro dispositivo que facilite a fase de segmentação e estimação do posicionamento das partes do corpo humano é necessário para manter a naturalidade. Mas isto exige um estudo aprofundado de algoritmos que sejam capazes de lidar com a informação utilizando apenas as dicas físicas visuais obtidas através de uma câmera. Portanto, podemos afirmar que quanto mais natural for a forma de interação por gestos com a máquina, mais complexa será a tarefa de segmentação e interpretação pelo software implementado.

A busca por esta solução de segmentação de gestos de forma natural é evidenciada por produtos atualmente presentes no mercado, que visam o reconhecimento e interpretação de movimentos para aplicações como interação com jogos, estatísticas para análise de marketing, interação com interfaces, aplicações militares, entre vários outros exemplos.

O Kinect, recentemente desenvolvido pela Microsoft, é um exemplo de arranjo de software e hardware que está presente no mercado com o objetivo de mapear a movimentação de partes do corpo humano. Sua principal funcionalidade está na simplificação da segmentação das imagens, utilizando uma tecnologia de estimação

de profundidade exemplificada na Figura 2. Sua tecnologia será abordada neste trabalho.

Figura 2 - Kinect para XBOX 360 e sua segmentação de profundidade

Um exemplo de aplicação militar é o projeto iARM (Intelligent Augmented Reality Mode), que está em desenvolvimento pela empresa Tanagram Partners[1] em Chicago - IL nos Estados Unidos. Este projeto combina tecnologias de segmentação e detecção de movimento com realidade aumentada, para auxiliar na tomada de decisão em campo de batalha, como exemplificado na Figura 3. A interface, em realidade aumentada, de interação do soldado com o sistema é projetada em um óculos e permite controle por gestos e voz.

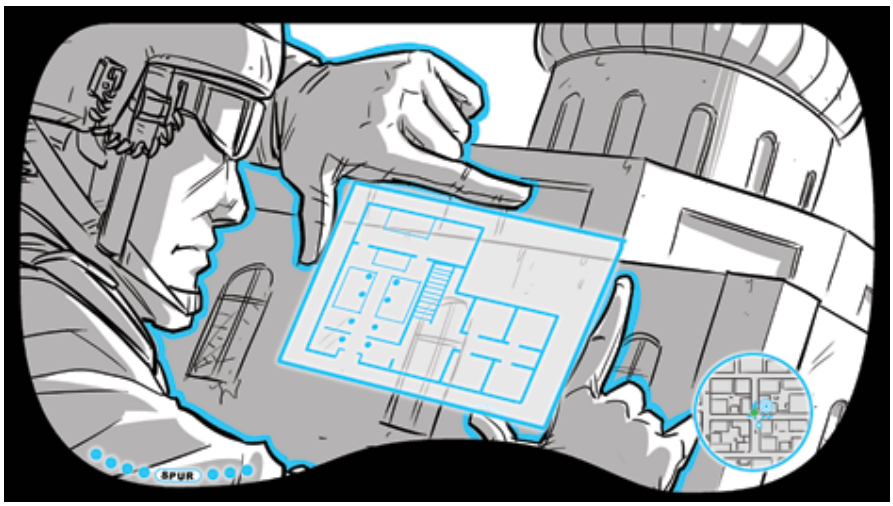


Figura 3 - Ilustração conceitual da funcionalidade de interação com mapas através de gestos e realidade aumentada

[1] TANAGRAM PARTNERS. Empresa de inovação tecnológica em variadas áreas. Disponível em: < http://tanagrampartners.com/ >. Acessado em: 03/03/2011.

A comunicação natural por meio de gestos é recente e ainda carente de aplicações voltadas para os deficientes auditivos ou mudos que se utilizam da língua de sinais para se comunicar. Com esta tecnologia, novas aplicações voltadas ao aprendizado da língua de sinais podem ser desenvolvidas, aprimorando assim o processo pedagógico de pessoas com tal deficiência, e também possibilitando a elas a interação com interfaces multimodais, que têm hoje como uma das formas de entrada natural o reconhecimento de comandos de voz.

1.3 Objetivo

O trabalho tem por objetivo o estudo de um sistema capaz de segmentar objetos que estão se movendo diante da(s) câmera(s), e dentre estes objetos identificar a figura humana e modelar as partes superiores do corpo humano necessárias para posteriormente reconhecer um conjunto finito de gestos da Língua Brasileira de Sinais (Libras), sempre levando em consideração o requisito de execução em tempo real.

Neste trabalho optamos por considerar satisfatório o tempo de 40 milissegundos para processamento de cada quadro capturado pela(s) câmera(s), sendo este nosso requisito de execução em tempo real.

Além disso, este trabalho deverá detalhar um estudo comparativo das soluções existentes de todas as etapas que compõe a tarefa de reconhecimento de gestos, sendo elas: *Segmentação; Modelagem e Identificação*; T*racking;* e *Reconhecimento*.

1.4 Organização do Trabalho

Subdividimos o projeto em três etapas organizando-as em três capítulos: O capítulo 2 apresenta o contexto de Segmentação e nossas soluções propostas; O capítulo 3 aborda a revisão bibliográfica e nossas decisões de projeto quanto a Modelagem e Tracking; Já o capítulo 4 apresenta o estado da arte para Reconhecimento de Gestos e também descreve nossas opções de projeto.

Devido a grande quantidade de informações, os resultados obtidos em cada uma das fases são apresentados ao final de cada capítulo.

No capítulo 5 o software GestureUI será apresentado. Posteriormente no capítulo 6 os resultados gerais de reconhecimento e avaliação de execução em tempo real do sistema serão relatados. E finalmente no capítulo 7 serão apresentadas as conclusões, trabalhos futuros e contribuições.

Neste trabalho optamos por manter os nomes originais em inglês dos métodos apresentados e também por não traduzir alguns termos técnicos da área de pesquisa. Um conjunto de termos amplamente utilizados nesta dissertação e suas definições serão apresentados na seção Glossário no final deste documento.

Capítulo 2

SEGMENTAÇÃO

Neste capítulo será apresentada a revisão bibliográfica da área de Segmentação no escopo do projeto, abrangendo basicamente a dicotomização da imagem em pixels de fundo e pixels de regiões em movimento. Posteriormente serão apresentados resultados obtidos nesta etapa e quais métodos foram adotados pelo sistema desenvolvido.

2.1 Considerações Iniciais

Segmentação é a tarefa de subdividir a imagem em regiões para facilitar a interpretação da informação contida na mesma. Esta subdivisão é feita através do agrupamento de pixels utilizando características (*features*) comuns a este conjunto. Estas propriedades podem ser cor, textura, intensidade ou continuidade.

São várias as técnicas de segmentação existentes, como por exemplo: Segmentação através de algoritmos de clusterização como o K-Means [LLOYD, 1982]; Métodos baseados em Histogramas [SHAPIRO et al., 2010]; Métodos baseados em compressão de imagens [MOBAHI et al., 2010]; Edge Detection [CANNY, 1986]; Region Growing [SHAPIRO et al., 2010]; Segmentação através de Grafos [GRADY, 2006]; Thresholding [GONZALEZ et al., 2007]; Segmentação utilizando Partial Differential Equations [OSHER et al., 2003]; Watershed transformation [BEUCHER et al., 1993]; Segmentação em múltiplas escalas [LINDEBERG, 1993]; Segmentação através de modelos pré-estabelecidos [MESTER et al., 1988]; Segmentação de regiões através da utilização de redes neurais como Multi-Layer Perceptron (MLP) [HAYKIN, 1998], Kohonen map [KOHONEN, 2001], Pulse-Coupled Neural Networks (PCNN) [JOHNSON et al., 1999], entre outras.

Porém, restringimos nossos estudos aos algoritmos passíveis de execução em tempo real. Logo, nas próximas seções serão abordados métodos de detecção de movimento por diferenças (Background Subtraction), Segmentação por cor de pele, Detecção de Bordas, Segmentação de profundidade e suas variações. Posteriormente, será apresentada a conclusão sobre os algoritmos estudados e a solução proposta para este projeto.

2.2 Subtração de Fundo (Background Subtraction)

Comumente utilizado para detecção de movimento através de diferenças, este método consiste na dicotomização da imagem em pixels pertencentes ao fundo e pixels referentes aos objetos de interesse ou em movimento na cena.

Esta tarefa é efetuada através da modelagem de fundo que consiste em uma modelagem estatística através de observações de uma sequência de imagens de uma cena para tentar estimar quais regiões da imagem pertencem ao fundo.

Para obter os pixels de interesse, que compõe os objetos em movimento, é efetuada uma simples subtração do modelo de fundo da imagem atual, seguido de um threshold.

Segundo Piccardi [PICCARDI, 2004] são vários os métodos de Background Subtraction, onde as principais variações estão na forma de estimação do modelo de fundo e no método de comparação deste modelo com a imagem atual. Alguns métodos citados por Piccardi, como Kernel Densisty Estimation (KDE), Sequential KD Approximation, Eigen Backgrounds entre outros, não possibilitam a implementação em tempo real.

Neste trabalho focamos nos métodos passíveis de utilização em tempo real, sendo estes: Subtração simples; Running Gaussian Average; Temporal Median Filter; e Mixture of Gaussians. Estes métodos serão descritos brevemente nas próximas seções. Posteriormente na seção 2.2.5 apresentaremos um estudo sobre os principais problemas dos métodos de Background Subtraction, e por último na seção 2.2.6 apresentaremos técnicas de Pós-Processamento para obter melhores resultados após a aplicação dos algoritmos de Background Subtraction.

2.2.1 Subtração simples

Método mais simples que consiste em subtrair pixel a pixel ($I_{x,y}$) a imagem anterior ($Frame_{i-1}$) da imagem atual ($Frame_i$). Em seguida, é aplicado um processo de uma binarização utilizando um limiar (σ) previamente estabelecido, ilustrado na Figura 4 e indicado na fórmula a seguir:

$$I_{x,y} = \begin{cases} 1 \; se \; |Frame_i(x,y) - \; Frame_{i-1}(x,y)| > \; \sigma \\ 0 \; se \; |Frame_i(x,y) - \; Frame_{i-1}(x,y)| < \; \sigma \end{cases}$$

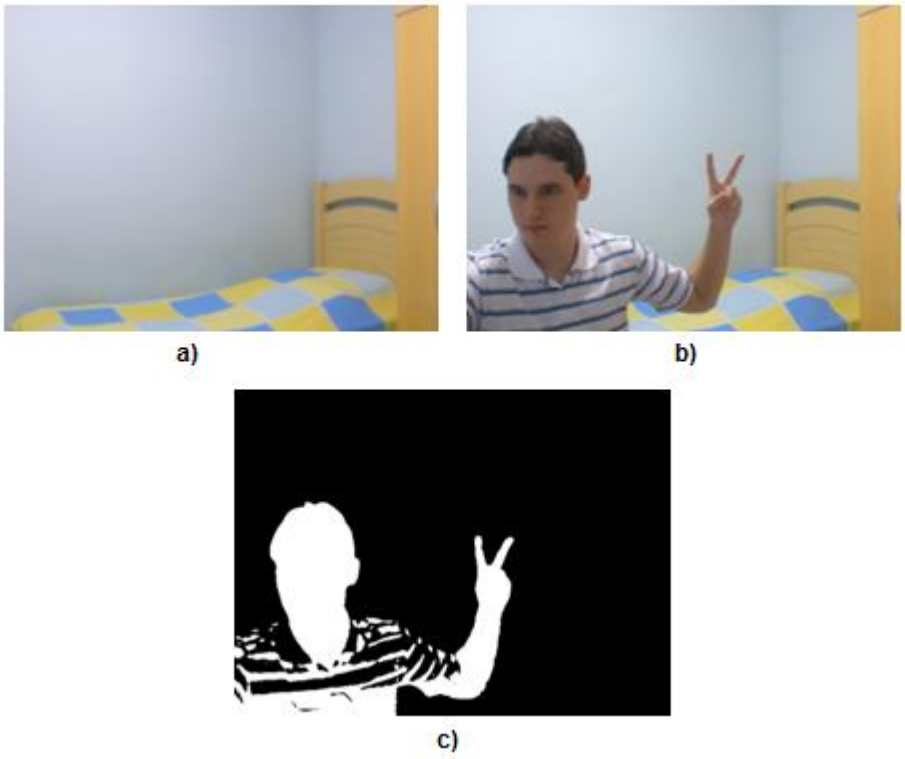


Figura 4 - Exemplo de resultado da subtração simples, sendo a) o modelo de fundo, b) a imagem atual e c) o resultado do algoritmo

2.2.2 Running Gaussian Average

Tomar a decisão entre pixels que sofreram alteração em relação ao modelo de fundo e os que são similares ao fundo, não pode simplesmente se basear na imagem anterior para a maioria das aplicações. Neste contexto, Christopher Wren [WREN et al., 1997] propôs um método no qual à medida que novas imagens são observadas, o modelo estatístico do fundo é modificado para se adaptar às pequenas alterações detectadas.

Mas esta modificação do modelo de fundo não pode permitir que as alterações sejam absorvidas pelo modelo rapidamente, ou seja, as alterações devem ser mínimas para adaptar o modelo lentamente, ao longo do tempo, a pequenas variações no fundo da cena.

Para isto, o modelo de fundo atual (μ_t) será calculado através da soma ponderada (α) entre a imagem recentemente observada (I_i) e o modelo de fundo anterior (μ_{t-1}), como na fórmula a seguir:

$$\mu_t = \alpha I_i + (1 - \alpha)\mu_{t-1}$$

Com o modelo de fundo, uma subtração simples é efetuada e seguida de uma binarização como no método anterior.

A alteração do modelo de fundo também pode ser efetuada de uma forma seletiva, ou seja, caso o pixel da imagem atual corresponda a um pixel de uma região em movimento (diferença detectada), ele não será utilizado para atualizar o modelo de fundo.

Para isso, é inserida na fórmula anterior a variável M que assume valor 1 quando o pixel em questão é uma diferença detectada e não influenciará no modelo de fundo e zero caso contrário, como indicado na fórmula a seguir:

$$\mu_t = M\mu_{t-1} + (1 - M)(\alpha I_i + (1 - \alpha)\mu_{t-1})$$

Esta operação pixel a pixel é simples e passível de implementação em tempo real.

2.2.3 Temporal Median Filter

Outra forma de se modelar o fundo da cena, considerada mais estável (gera menos ruído como pixels de regiões em movimento) por alguns pesquisadores, é armazenar um buffer de frames para depois calcular a média e obter um modelo de fundo. A maior desvantagem deste método é o armazenamento em memória deste buffer de imagens, e também a definição do tamanho do mesmo que determinará quanto tempo será necessário para que um objeto estático passe a fazer parte do fundo.

2.2.4 Mistura de Gaussianas (Mixture of Gaussians)

Todos os métodos anteriores definem para cada pixel um único valor no modelo de fundo. Em alguns casos, pode-se ter mudanças no fundo que não devem ser detectadas como diferenças, mas que acontecem em um intervalo de tempo menor que o da atualização do modelo.

Um exemplo desta situação seria uma árvore que cobre parcialmente um prédio e se movimenta com o vento. Um mesmo pixel pode assumir o valor do prédio, de um galho ou de uma folha.

Para solucionar este problema Stauffer e Grimson [STAUFFER et al., 1999] modelaram cada pixel da imagem com um conjunto de Gaussianas, que seriam responsáveis pela modelagem de cada uma das situações do fundo que não devem ser detectadas como diferenças.

Segundo Stauffer e Grimson [STAUFFER et al., 1999], a probabilidade de observar um certo valor x de um pixel no tempo t utilizando um conjunto (K) de gaussianas multivariadas (η) é dado pela fórmula a seguir:

$$P(x_t) = \sum_{i=1}^{K} w_{i,t}\eta(x_t - \mu_{i,t}, \sum_{i,t}) \text{ sendo } \sum_{i=1}^{K} w_{i,t} = 1$$

Mas para que esta fórmula possa modelar o fundo, um critério deve ser definido para diferenciar entre as distribuições gaussianas referentes ao fundo e as distribuições gaussianas das diferenças. Para isto, as distribuições são ordenadas segundo relação entre a amplitude e o desvio padrão, e as primeiras são definidas como fundo. Por exemplo, se cada pixel for modelado com três gaussianas, pode-se estabelecer que as duas primeiras colocadas serão definidas como fundo e a restante será correspondente as diferenças.

Usualmente o número de gaussianas utilizadas (K) varia entre três e cinco. Quanto maior o número de gaussianas maior é o tempo de processamento, dificultando a sua utilização em aplicações de tempo real.

A cada novo frame observado, os parâmetros das gaussianas que modelaram cada um dos pixels identificados devem ser atualizados. Inicialmente o sistema está em fase de adaptação, modelando o ambiente. Após alguns frames ele já é capaz de classificar os pixels.

2.2.5 Principais problemas da detecção de movimento por Background Subtraction

Apenas modelar o fundo e aplicar subtração para detectar as diferenças não é suficiente para garantir um bom resultado na segmentação. Essas diferenças detectadas podem ser highlights, ghosts ou sombras, que não deveriam fazer parte do resultado final. Estes problemas, juntamente com possíveis soluções, serão apresentados nesta seção.

Outro ponto importante que será abordado são regras de decisão para atualizar o modelo de fundo e câmeras com configurações automáticas que definem como o sistema irá se comportar nas atualizações do modelo que envolvem pixels detectados como diferença.

2.2.5.1 Alterações de Brilho (Highlights)

Este tipo de variação acontece com mudanças repentinas na iluminação da cena que está sendo observada. Esta variação não é facilmente detectada e modelada no espaço de cor RGB (Red, Green e Blue), por exemplo.

Por isso, Duque e seus colegas [DUQUE et al., 2005] propuseram uma solução utilizando o espaço de cor HSV (Hue, Saturation e Value) para modelar as variações de cromacidade dos objetos que seriam classificadas como Highlights.

Este método tem como entrada a imagem atual I_n^{HSV} e a imagem do modelo de fundo B_n^{HSV} no tempo n, e retorna como resultado uma máscara binarizada obtida através da seguinte fórmula:

$$HM_n(x,y) = \begin{cases} \mathbf{1}, & se\ \frac{1}{\beta} \leq \frac{I_n^V(x,y)}{B_n^V(x,y)} \leq \frac{1}{\alpha}\ \wedge \\ & |I_n^S(x,y) - B_n^S(x,y)| < \tau_S\ \wedge \\ & |I_n^H(x,y) - B_n^H(x,y)| \leq \tau_H \\ \mathbf{0}, & caso\ contrário \end{cases}$$

O parâmetro α depende da fonte de iluminação e das propriedades de reflectância e irradiação dos objetos na cena. Fontes de iluminação intensas e objetos com altas taxas de irradiação e reflectância implicam em baixos valores de α.

Reflectância é a proporção entre o fluxo de raios luminosos que incidem sobre o objeto e o fluxo de raios que é refletido. Já a irradiação corresponde à emissão de

ondas do espectro visível ou não por objetos, podendo ser fontes luminosas ou fontes de calor, por exemplo, porém este tipo de característica só irá influenciar caso a câmera sendo utilizada seja sensível a este espectro de ondas.

O parâmetro β é utilizado para evitar a classificação de ruídos como highlights. Já os limiares τ_S e τ_H são a máxima variação permitida para os componentes Saturação e Hue do espaço de cor.

Duque e seus colegas [DUQUE et al., 2005] definiram valores ideais para um bom funcionamento do algoritmo, como indicado na Tabela I.

Parâmetro	Valor
α	0,60 - 0,80
β	0,90 - 0,95
τ_S	15%
τ_H	60 graus

Tabela I - Valores ideais para o algoritmo de detecção de highlights, segundo Duque [DUQUE et al., 2005]

2.2.5.2 Sombras

A presença de sombras dos objetos, que são projetadas pela fonte de luz na cena sendo observada, pode modificar a forma do objeto, resultando em uma falha na segmentação. Para evitar este problema Duque [DUQUE et al., 2005] também propôs uma solução similar a detecção de Highlights com a seguinte fórmula:

$$SM_n(x,y) = \begin{cases} \mathbf{1}, & se\ \alpha \leq \dfrac{I_n^V(x,y)}{B_n^V(x,y)} \leq \beta\ \wedge \\ & |I_n^S(x,y) - B_n^S(x,y)| < \tau_S\ \wedge \\ & |I_n^H(x,y) - B_n^H(x,y)| \leq \tau_H \\ \mathbf{0}, & caso\ contrário \end{cases}$$

que retorna uma máscara binarizada contendo os pixels classificados como sombra. Seus parâmetros devem ter os mesmos valores utilizados na fórmula de detecção de highlights, indicados na Tabela I.

2.2.5.3 Fantasmas (Ghosts)

Os ghosts são regiões (blobs) erroneamente classificadas como diferenças que foram geradas pela movimentação de um objeto que fazia parte do modelo de fundo. Este tipo de erro de classificação impede o ajuste do modelo de fundo naquela região, em caso de atualização seletiva do modelo, gerando um deadlock [CUCCHIARA et al.,2003].

Os trabalhos publicados para solução do problema podem ser divididos em duas categorias: decisão baseada nos valores dos pixels e decisão baseada nas regiões ou blobs. Ou seja, uma solução foca seu escopo em uma decisão pontual e local, e a outra toma a decisão em um escopo global.

Rita [CUCCHIARA et al.,2003] propôs uma solução de escopo global que identifica blobs detectados como diferenças e classifica-os como objetos em movimento, sombra de objetos em movimento, ghosts e sombra de ghosts. Para diferenciar um blob de um objeto em movimento e um ghost, é utilizado o método proposto por [BAINBRIDGE-SMITH, 1997] para o cálculo aproximado do Fluxo Ótico (optical flow) através de equações diferenciais. Este fluxo ótico quantifica a tendência de movimento de um blob ao longo do tempo. Então ghosts tendem a permanecer imóveis diferenciando assim ghosts de objetos em movimento.

Duque [DUQUE et al., 2005] propôs outra solução de escopo global, utilizando o que ele chamou de Moving Edge Detection (MED). Este algoritmo consiste em aplicar detecção de bordas na subtração do frame atual com o frame anterior, obtendo assim as bordas dos objetos que estão em movimento. Na imagem obtida do algoritmo de background subtraction, ele efetua detecção de blobs e faz uma comparação de perímetro dos blobs detectados com as bordas da imagem obtida no MED. Caso a diferença supere 90% o blob será classificado como ghost.

2.2.5.4 Câmeras de configuração automática

Os algoritmos citados até agora para Background Subtraction se baseiam nos valores dos componentes do espaço de cor da imagem para tomar uma decisão, e é a câmera que discretiza esses valores contidos nos pixels da imagem.

As webcams que vêm embutidas em notebooks, ou mesmo as externas via conexão USB, são geralmente soluções de baixo custo que tentam se adaptar às mudanças do ambiente (iluminação e cor) para discretizar imagens digitais. Esta

adaptação é prejudicial para os algoritmos citados anteriormente, pois geram variações repentinas que podem ser classificadas como diferenças.

As principais configurações que se adaptam automaticamente nas câmeras são White Balance, Exposure Time e Automatic Gain Control que serão definidas aqui segundo Mchugh[2].

White Balance é o processo de remoção de predominância de cor irreal em uma imagem digitalizada, para que objetos da cor branca sejam digitalizados na cor branca. O algoritmo de White Balance da câmera deve levar em consideração a temperatura da fonte de luz que ilumina a cena para obter um bom resultado. Um exemplo de correção é indicado na Figura 5.

Figura 5 - Exemplo de correção de White Balance

Exposure Time, também conhecido como Shutter Speed, define o intervalo de tempo que o sensor da câmera ficará exposto à luz para digitalizar uma imagem. Isto permite que a câmera se adapte a diferentes condições de iluminação, tentando manter um bom contraste na imagem.

Automatic Gain Control é utilizado para melhorar a visualização da imagem quando o Exposure Time não conseguiu uma imagem nítida. É um algoritmo de multiplicação dos valores da imagem, na tentativa de aumentar o contraste.

[2] MCHUGH, Sean. Understanding White-Balance. Disponível em: < http://www.cambridgeincolour.com/tutorials/white-balance.htm >. Acessado em: 25/08/2012.

2.2.5.5 Regra de decisão para atualizar modelo de fundo

Após obter como resultado as diferenças entre o frame atual e o modelo de fundo, todos os algoritmos de Background Subtraction citados anteriormente precisam atualizar de alguma forma o modelo.

Na atualização existem duas opções:

- ***Atualização direta:*** Atualizar o modelo de fundo sem utilização da informação contida no resultado do Background Subtraction. Assim, todo o frame corrente é utilizado na atualização.
- ***Atualização condicional:*** Atualizar o modelo de fundo utilizando informações contidas no resultado do Background Subtraction. Os blobs desta imagem podem indicar objetos em movimento que não devem influenciar no modelo de fundo.

Segundo [PARKS et al., 2008] a atualização condicional aumenta as chances de detecção de diferenças com precisão devido ao fato de objetos em movimento não poluírem o modelo de fundo. Ela pode também solucionar problemas como Ghosts, validando primeiro os blobs antes da atualização. Porém um blob mal classificado pode se tornar um problema, pois continuará sendo identificado como diferença e nunca será incorporado ao modelo de fundo.

Portanto, no caso de utilização de uma metodologia de atualização condicional, deve-se ter o cuidado de garantir a classificação correta dos blobs, ou incorporar um algoritmo de verificação para eliminar blobs que permaneçam no resultado da segmentação por um período de tempo, evitando assim deadlocks.

2.2.6 Técnicas de Pós-Processamento

Parks [PARKS et al., 2008] identificou em seu estudo dos principais algoritmos de Background Subtraction, algumas técnicas de pós-processamento que costumam ser utilizadas para tentar garantir a morfologia dos objetos segmentados e eliminar ruídos indesejáveis. Algumas destas técnicas serão descritas a seguir.

2.2.6.1 Remoção de ruídos

Ruídos são geralmente gerados pela câmera e devem ser eliminados o quanto antes para não interferir em etapas posteriores. Os mesmos podem ser

eliminados através da convolução de uma máscara de média na imagem, seguida de um threshold simples.

2.2.6.2 Identificação de Blobs

Como o resultado dos algoritmos de Background Subtraction geralmente é uma máscara binária, pode-se aplicar algoritmos de identificação de blobs. Estes algoritmos têm por objetivo a identificação de regiões conexas na imagem, rotulando as mesmas com um identificador único.

Este procedimento permite calcular características de cada um desses blobs e utilizar essas informações para tentar modelar e classificar o que está sendo observado pela câmera.

Chang [CHANG et al., 2004] propôs um método, que funciona em tempo linear, para rotular blobs em uma imagem binária. Percorrendo apenas uma vez a imagem, o algoritmo é capaz de rotular os blobs utilizando seus contornos internos e externos. Este algoritmo pode ser adaptado para também rotular os “buracos” contidos nos blobs através dos contornos internos.

2.2.6.3 Area thresholding

Area Thresholding consiste na eliminação de blobs que possuam área inferior a β. Pode ser considerado também um algoritmo de eliminação de ruído, mas o tempo de execução será maior. Isto se deve ao fato da necessidade de identificação (rotulação) de blobs seguida de Area Thresholding que são tarefas que têm tempo de processamento variável dependendo da quantidade de blobs na imagem.

Segundo [PARKS et al., 2008] para garantir um bom desempenho β deve ser bem abaixo da área do menor objeto de interesse, por exemplo, 25% da mesma.

2.2.6.4 Morphological closing

Antes de explicar a operação Morphological Closing, é necessária a apresentação de três conceitos comuns em processamento de imagens, sendo eles: Structuring Element; o operador Dilatação ($\oplus$); o operador Erosão ($\ominus$).

O structuring element é similar a uma máscara de convolução, que é utilizado como elemento principal das operações morfológicas em processamento de

imagens, e é convolucionado ou deslocado pela imagem para efetuar a operação desejada.

O operador de Dilatação ($\oplus$) consiste basicamente na convolução de um structuring element em uma imagem binária marcando todos os pontos por onde passa desde que ao menos um ponto da máscara se sobreponha a um pixel previamente marcado na imagem.

Já o operador de Erosão ($\ominus$) consiste basicamente também na convolução de um structuring element em uma imagem binária, mas só irá marcar os pontos em que a máscara esteja totalmente contida em uma região de pixels previamente marcada.

Finalmente, a operação Morphological Closing, ou operação de fechamento, é indicada pelo operador •, onde A é a imagem binária e B é o chamado structuring element ou kernel que definirá a forma de interação do algoritmo com a imagem. Esta operação é efetuada como indicado na fórmula a seguir:

$$A \bullet B = (A \oplus B) \ominus B$$

Onde $\oplus$ é o operador de dilatação e $\ominus$ é o operador de erosão. Portanto, a operação de fechamento é a erosão da dilatação da imagem A por um kernel B.

Parks [PARKS et al., 2008] detectou em seu estudo comparativo, que todos os algoritmos de Background Subtraction melhoraram seus desempenhos quando a operação de Morphological closing é utilizada.

2.2.6.5 Saliency test

Segundo [PARKS et al., 2008] o *Saliency test* ou *Teste de "saliência"* é utilizado para eliminar blobs em que $\gamma\%$ dos seus pixels não são salientes o suficiente. Um pixel é saliente se ele continuar sendo classificado como diferença após o limiar para threshold ser multiplicado por um fator α. Geralmente o fator α é igual a 2.

Portanto, este teste tenta garantir que o blob seja suficientemente diferente do modelo de fundo, tentando evitar falsos positivos na classificação de blobs de objetos que estão em movimento.

2.2.6.6 Optical flow test

Este teste é utilizado para solucionar o problema de ghosts descrito anteriormente. Consiste no cálculo da média da tendência de movimento dos pixels do blob. Caso a média se aproxime de zero este blob é um ghost e deve ser eliminado.

2.3 Segmentação por cor de pele

Segundo [KAKUMANU et al., 2007] a segmentação por cor de pele está presente em vários tipos de aplicações como detecção de faces, tracking de faces, reconhecimento de gestos, sistemas de busca de imagens orientados a conteúdo, e várias outras aplicações no ramo de interação humano-computador.

A segmentação para interação natural por gestos pode ser simplificada pela utilização de uma característica comum a regiões de interesse como mãos e braços, que é a cor da pele. Em um cenário ideal, todos os membros com pele exposta seriam corretamente segmentados e utilizados para uma posterior modelagem e interpretação.

Porém, estes sistemas são limitados por vários fatores como variação de iluminação, variação de tonalidade de pele devido à variabilidade étnica, ocultação por roupas ou luvas, fundo, características da câmera, entre outros fatores.

Nesta seção os principais problemas na segmentação por cor de pele serão relatados, e alguns algoritmos de detecção serão apresentados.

2.3.1 Principais fatores limitantes na segmentação por cor de pele

Os principais fatores limitantes, segundo [KAKUMANU et al., 2007] são:

- **Iluminação:** Mudanças na fonte de luz ou em sua intensidade em ambientes internos ou externos, sombras, mudanças repentinas nas condições de iluminação são fatores que alteram a cor de pele na imagem. É considerado como o principal problema na detecção de cor de pele, que reduz consideravelmente o desempenho destes sistemas.

- **Características da câmera:** Uma mesma pessoa diante de câmeras diferentes pode ter uma diferença na cor da pele. Isto se deve ao tipo de sensor utilizado para digitalizar imagens, que pode ser CCD ou CMOS por exemplo, e sua sensibilidade e discretização do espectro de cor observado.
- **Variabilidade étnica:** A cor de pele varia entre as etnias, e também entre as regiões onde as pessoas vivem. Este tipo de variabilidade deve ser levada em consideração na detecção, e é outro fator que dificulta a precisão dos sistemas.
- **Características individuais:** São características como idade e sexo, e outras que são singulares a cada indivíduo, que podem alterar a tonalidade de cor da pele.
- **Outros fatores:** fatores diferentes como maquiagem, corte de cabelo, óculos, cor de fundo, sombras e movimento também influenciam na cor de pele.

2.3.2 Algoritmos de detecção de cor de pele

Um dos fatores cruciais antes de modelar um sistema de detecção de cor de pele é definir qual espaço de cor será utilizado. Segundo [VEZHNEVETS et al., 2003] e [KAKUMANU et al., 2007], em suas pesquisas comparativas, a maioria dos trabalhos publicados na área utilizaram os seguintes espaços de cor: RGB; RGB normalizado; CIE-XYZ; HSI; HSV; HSL; TSL; YCbCr; YIQ; YUV; YES; CIE-Lab; CIE-Luv. Lembrando que todos eles foram obtidos através de uma transformação linear ou não-linear do RGB.

Os trabalhos de Vezhnevets [VEZHNEVETS et al., 2003] e KAKUMANU [KAKUMANU et al., 2007] também apontaram que os algoritmos de detecção de cor de pele se baseiam na dicotomização entre pixel cor de pele e pixel não cor de pele. Eles dividiram estes algoritmos em categorias como: Threshold explícito no espaço de cor; Modelos de histogramas com classificadores ingênuos de Bayes; Classificadores Gaussianos; Elliptical boundary model; Classificador Multi-Layer Perceptron (MLP); Classificador Self Organizing Map (SOM); Classificador de máxima entropia; Classificador Bayesian Network (BN).

Para aplicações como reconhecimento de gestos, o tempo de processamento é um fator crucial, portanto algoritmos mais simples como os de Threshold explícito no espaço de cor são mais utilizados. Levando em consideração o requisito de tempo real e complexidade de otimização, alguns algoritmos mais simples e eficientes serão explicados a seguir.

2.3.2.1 Threshold explícito no espaço de cor

É o mais simples dos métodos, e consiste na observação da dispersão amostral dos valores de cor de pele em um conjunto de treinamento, e na determinação de um threshold para cada canal de cor.

Este algoritmo é utilizado em vários espaços de cor. Chai [CHAI et al., 1999], por exemplo, propôs um sistema de detecção de faces utilizando detecção de cor de pele no espaço de cor YCbCr, eliminando a componente de iluminação Y e definindo um threshold fixo para os canais Cb e Cr: $R_{Cb} = [77,127]$ e $R_{Cr} = [133,173]$.

Wang [WANG et al., 2001] utilizou os espaços de cor RGB normalizado e HSV simultâneamente, utilizando também um threshold fixo para os canais R, G, H, S e V: $R_r = [0.36,0.465]$; $R_g = [0.28,0.363]$; $R_H = [0,50]$;$R_S = [0.20,0.68]$; $R_V = [0.35,1.0]$.

Os resultados obtidos através de threshold explícito são geralmente satisfatórios para aplicações que não exigem muita precisão na classificação dos pixels. Os resultados deste método podem conter ruídos e sua capacidade de generalização é limitada ao espaço amostral utilizado para escolher os limiares.

2.3.2.2 Gaussian mixture models (GMM)

Segundo [KAKUMANU et al., 2007], vários trabalhos publicados na área propuseram soluções de modelagem estatística da cor de pele utilizando n-Gaussianas. A vantagem deste tipo de solução é que as gaussianas podem generalizar bem na classificação, utilizando poucas amostras de treinamento.

Uma única Gaussiana não seria capaz de absorver a variabilidade amostral da cor de pele. Portanto vários pesquisadores optaram por utilizar um conjunto de gaussianas como na fórmula a seguir:

$$P(p) = \sum_{i=1}^{n} w_i \eta(p - \mu_i, \sum_i)$$

Onde n corresponde ao número de pixels e $P(p)$corresponde a probabilidade de p ser um pixel cor de pele.

Lee [LEE et al., 2002] utilizou seis gaussianas para detecção de cor de pele, treinando com o Compaq Dataset e obteve 90% de acerto no reconhecimento. Já Thu [THU et al., 2002] utilizou quatro gaussianas em outro conjunto de dados menos complexo mas não apresentou resultados gerais, apenas alguns exemplos de aplicação de seu algoritmo, como representado na Figura 6. Ou seja, o número de gaussianas depende da variabilidade de seu conjunto de dados não só em etnia, mas também em iluminação.

Figura 6 - Exemplo de resultado da extração de regiões que contém cor de pele, sendo as imagens à esquerda as originais e as à direita o resultado obtido

2.3.2.3 Bayesian network (BN)

Segundo [KAKUMANU et al., 2007], Bayesian networks são grafos acíclicos dirigidos que permitem uma representação eficiente de funções densidade de probabilidade condicionais.

Sebe [SEBE et al., 2004] utilizou uma BN para modelagem e classificação de pixels com cor de pele. Um dos maiores problemas com reconhecimento de padrões é a disponibilidade de sets de teste e treinamento. Neste projeto, Sebe utilizou o algoritmo Stochastic Structure Search (SSS), para treinar a estrutura da BN e melhorar os resultados de generalização com um menor set de treinamento.

Sebe utilizou 60 mil amostras de pixels extraídas de imagens aleatórias para compor um espaço amostral com pixels de cor de pele e pixels com quaisquer outras cores. Neste conjunto de amostras 600 estavam rotuladas e foram utilizadas para treinamento do classificador. A avaliação do classificador através de pixels não rotulados obteve taxas de acerto de 95,82% a 98,32% e taxas de falsos positivos de 5% a 10%.

2.4 Edge Detection

Segundo [GONZALEZ et al., 2007], intuitivamente, edges ou bordas são pixels conectados que estão no limite entre duas regiões. Mais especificamente, edges são pontos de descontinuidade na imagem, ou seja, pixels onde há mudança brusca na intensidade em relação a sua vizinhança. Este é um dos métodos mais utilizados para tentar detectar estruturas significativas em uma imagem.

Os métodos existentes para detecção de edges utilizam aproximações das derivadas de primeira e segunda ordem para detectar as descontinuidades da imagem.

Em um cenário ideal, ao aplicar um algoritmo de detecção de bordas em uma imagem, o resultado seria as bordas de todos os objetos contidos no cenário, e também detalhes internos destes objetos. Porém, o resultado é afetado por sombras, borrões, falta de contraste, entre vários outros fatores.

Nesta seção serão abordados brevemente dois dos métodos de detecção de bordas: (a) o operador de Sobel para aproximação de descontinuidades e (b) o algoritmo de Canny para detecção de bordas.

2.4.1 Sobel

O operador de Sobel é utilizado para calcular aproximadamente o gradiente dos valores de intensidade da imagem. Para isso, efetua-se uma convolução na imagem utilizando duas máscaras de tamanho 3x3, um para direção y e outra para a direção x, como indicado na fórmula a seguir:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A \ e \ G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A$$

Onde G_y é o gradiente aproximado na direção y, G_x é o gradiente aproximado na direção x, A é a imagem e $*$ é o operador de convolução. Com os valores dos gradientes aproximados, o gradiente de cada pixel é calculado como indicado na fórmula a seguir:

$$G = \sqrt{{G_x}^2 + {G_y}^2}$$

É possível também calcular a direção do gradiente utilizando os gradientes aproximados nas direções x e y como indicado na fórmula a seguir:

$$\Theta = \arctan\left(\frac{G_y}{G_x}\right)$$

Um exemplo do resultado da aplicação do operador de Sobel em uma imagem é demonstrado na Figura 7.

Figura 7 - Exemplo de aplicação do operador de Sobel

2.4.2 Canny

Canny [CANNY, 1986] propôs um método de detecção de bordas que é amplamente utilizado em sistemas de visão computacional. Canny tinha como objetivo formular uma solução ótima para a detecção de bordas. Para ele, um ótimo detector de bordas significa:

- **Good detection:** o algoritmo deve ser capaz de detectar a maior quantidade possível de bordas “reais” na imagem.
- **Good localization:** as bordas detectadas devem ser localizadas o mais próximo possível das bordas reais na imagem original.
- **Minimal response:** uma determinada borda deve ser marcada uma única vez, e se possível, ruídos não devem originar falsas bordas.

Green[3] sintetizou o algoritmo de Canny em seis etapas que serão descritas brevemente a seguir.

A primeira etapa consiste na *Redução de ruídos,* onde uma máscara Gaussiana é convolucionada na imagem antes de iniciar o processo de detecção de ruídos. Esta transformação na imagem (I_i) é efetuada como na fórmula a seguir:

$$I_f = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * I_i$$

A segunda etapa é a aplicação do operador de Sobel para estimar a magnitude do gradiente em cada pixel, como já explicado na seção anterior.

A terceira etapa consiste no cálculo da direção do edge (Θ) de cada pixel, que é efetuada da mesma forma descrita na seção anterior:

$$\Theta = \arctan\left(\frac{G_y}{G_x}\right)$$

A quarta etapa consiste na classificação da direção dos edges de cada pixel em quatro categorias identificadas por cores, sendo elas: Amarela, [0 – 22,5] e[157,5 – 180]; Verde, [22,5 - 67,5]; Azul, [67,5 – 112,5]; Vermelha, [112,5 to 157,5].

A quinta etapa é chamada de *Non-maximum suppression* e consiste na detecção dos pontos que seus gradientes são máximos locais em suas direções. Estes pontos serão marcados em uma imagem binária.

A sexta e última etapa aplica um método chamado *Hysteresis thresholding* para eliminar edges detectados que não correspondem a edges na imagem original. Este método consiste na validação dos edges através de dois thresholds na magnitude dos gradientes, um alto e um baixo, utilizando o resultado do threhsold

[3] GREEN, B. Canny Edge Detection Tutorial. Disponível em: < http://dasl.mem.drexel.edu/alumni/bGreen/www.pages.drexel.edu/_weg22/can_tut.html >. Acessado em: 25/08/2012.

alto como ponto de partida para percorrer os edges e eliminar os que são considerados falsas detecções.

Um exemplo do resultado do algoritmo de Canny aplicado na mesma imagem da figura pode ser visto na Figura 8.

Figura 8 - Exemplo de resultado de aplicação do algoritmo de Canny

2.5 Edge Orientation Histogram

Histogramas são comumente utilizados para representar imagens. Eles organizam a informação nos chamados bins, que são as barras utilizadas para indicar a quantidade de ocorrências de um determinado valor na imagem, e formam uma representação gráfica dos dados como apresentado na Figura 9.

Histogramas também podem ser utilizados para descrever objetos ou cenas, e através de algoritmos de comparação de histogramas, definir se um histograma é semelhante ou equivalente a outro.

Histogramas podem ser construídos utilizando vários tipos de características como intensidade de canais de cor (RGB, por exemplo), orientação do gradiente calculado para detecção de Edges, ou qualquer outra transformação da imagem que possa ser quantificada e subdivida em grupos (bins).

Muitos autores utilizaram Edge Orientation Histograms para identificar objetos, auxiliar na subtração de fundo, ou detectar objetos em movimento. Alguns trabalhos serão exemplificados a seguir.

Mason [MASON et al., 2001] propôs um método de modelagem de fundo utilizando Edge Orientation Histograms, que subdivide a imagem em regiões quadriculadas, que são utilizadas para computar os histogramas que compõem o modelo de fundo. Para descobrir se a região contém algum objeto em movimento os histogramas da imagem atual com o modelo de fundo são comparados e se a diferença for maior que um limiar, esta região é marcada como região com movimento.

Miller [MILLER et al., 2008] propôs a utilização de Edge Histograms como característica de treinamento para um classificador possibilitando a segmentação de pessoas e veículos. Esta aplicação foi utilizada em sistemas de vigilância.

Chen [CHEN et al., 2008] propôs um método de detecção e segmentação de regiões em uma imagem que contém pessoas utilizando Haar-like features e Edge Orientation Histograms para treinar uma rede neural com o algoritmo Real AdaBoost.

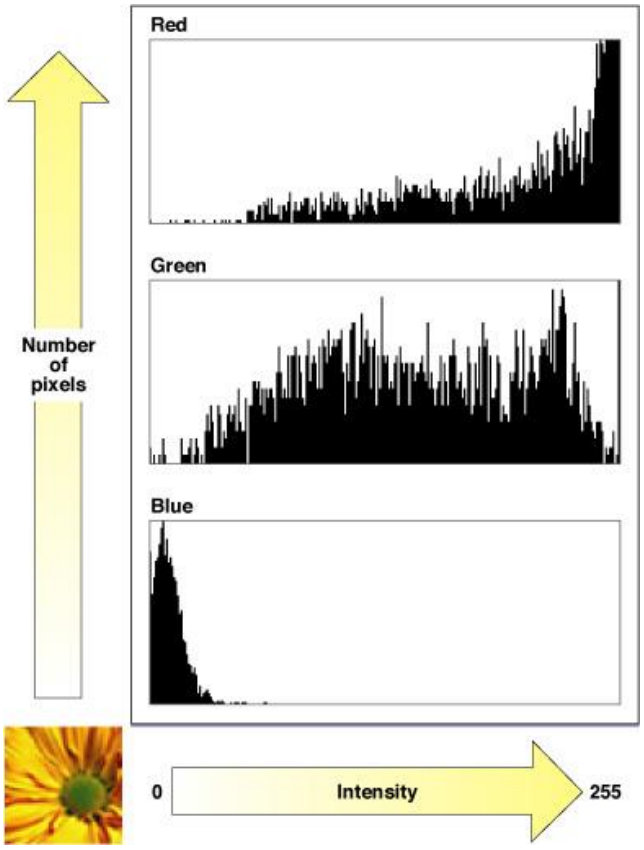


Figura 9 - Exemplo de um Histograma para os canais R, G e B da imagem no canto inferior esquerdo

2.6 Segmentação de profundidade utilizando visão estéreo

Segundo Otuyama[4], visão estéreo é o ramo da visão computacional que analisa o problema da reconstrução da informação tridimensional de objetos a partir de um par de imagens capturadas simultaneamente, mas com um pequeno deslocamento lateral.

O princípio básico foi obtido do exemplo do par de olhos humanos, que estão a uma distância conhecida entre eles e possuem duas perspectivas ligeiramente diferentes da cena sendo observada. Esta diferença pode ser utilizada para estimar a profundidade de objetos a partir da comparação das duas imagens.

Existem duas formas de correlacionar as imagens e aproximar uma medida de profundidade: uma é através da intensidade luminosa das imagens, ou seja, comparar as regiões pelos valores de cada pixel e a outra forma é através de características como bordas, formas geométricas conhecidas, corners (cantos), entre outras propriedades que podem ser utilizadas para aproximar a profundidade através das diferenças de perspectiva.

Scharstein [SCHARSTEIN et al., 2002] fez uma estudo comparativo dos métodos mais conhecidos, avaliando todos com uma extensa base de dados e comparando eficiência e eficácia. Após seu trabalho, novos métodos surgiram como Tjandranegara [TJANDRANEGARA, 2005] que propôs um novo método para câmeras estáticas sem conhecimento prévio de distância e ângulo entre as câmeras, e Yang [YANG et al., 2007] que propôs um método de pós-processamento para otimizar a qualidade das imagens de profundidade obtidas de câmeras de profundidade, utilizando visão estéreo de duas câmeras coloridas de alta resolução.

2.7 Segmentação de profundidade utilizando infravermelho

As chamadas câmeras de profundidade são dispositivos que retornam como um de seus resultados um mapa de distâncias da cena sendo observada. Para

[4] OTUYAMA, J. M. Visão Computacional - Visão Estéreo. Disponível em: < http://www.inf.ufsc.br/~visao/1998/otuyama/index.html >. Acessado em: 25/08/2012.

sistemas de detecção e interpretação de movimentos, como reconhecimento de gestos, esta é uma solução que simplifica a fase de segmentação.

No mercado as câmeras mais comuns baseavam-se em sistemas binoculares com câmeras em arranjo estéreo para estimar profundidade. Esta solução tem um custo alto, e uma precisão não muito aceitável para sistemas que exigem riqueza de detalhes.

Recentemente a Microsoft em pareceria com a empresa israelense PrimeSense, desenvolveu o Kinect[5]. Este é um dispositivo adicional para a plataforma de jogos Xbox 360, que possibilita a interação através dos movimentos de seu corpo.

Ele foi projetado para ser utilizado em ambientes fechados e para eliminar os problemas com a luz visível que já foram descritos neste capítulo. Propõe, para tanto, uma solução no espectro Infravermelho tornando o sistema robusto as condições do ambiente onde for instalado e até possibilitando sua utilização em locais com iluminação precária ou ausente.

Basicamente, este dispositivo funciona utilizando: um projetor de raios infravermelho classe 1 com comprimento de onda de 830nm; uma câmera com filtro infravermelho; e uma câmera VGA - RGB tradicional. Estes componentes são indicados na Figura 10 – a). Com este arranjo de hardware, o Kinect consegue estimar a posição do usuário desde que o mesmo esteja a pelo menos 80cm do dispositivo ou no máximo a 4m de distância. Fora destes limites os resultados de estimação são imprevisíveis.

O projetor de raios infra e a câmera com filtro são montados em um arranjo estéreo, possibilitando que um sistema embarcado correlacione o padrão projetado pelo projetor de raios infra e a imagem capturada pela câmera. O resultado final é um mapa de profundidade com boa precisão.

O padrão infravermelho projetado pelo Kinect é exemplificado na Figura 10 – b), onde na imagem superior o padrão pode ser visto em uma parede plana evidenciando uma separação em quadrantes e alguns pontos de referência para auxiliar na estimação. Já na imagem inferior o padrão é projetado em uma sala com alguns móveis evidenciando a área de interação.

[5] Microsoft. Kinect for Xbox360. Disponível em: < http://www.xbox.com/pt-br/kinect >. Acessado em: 25/08/2012.

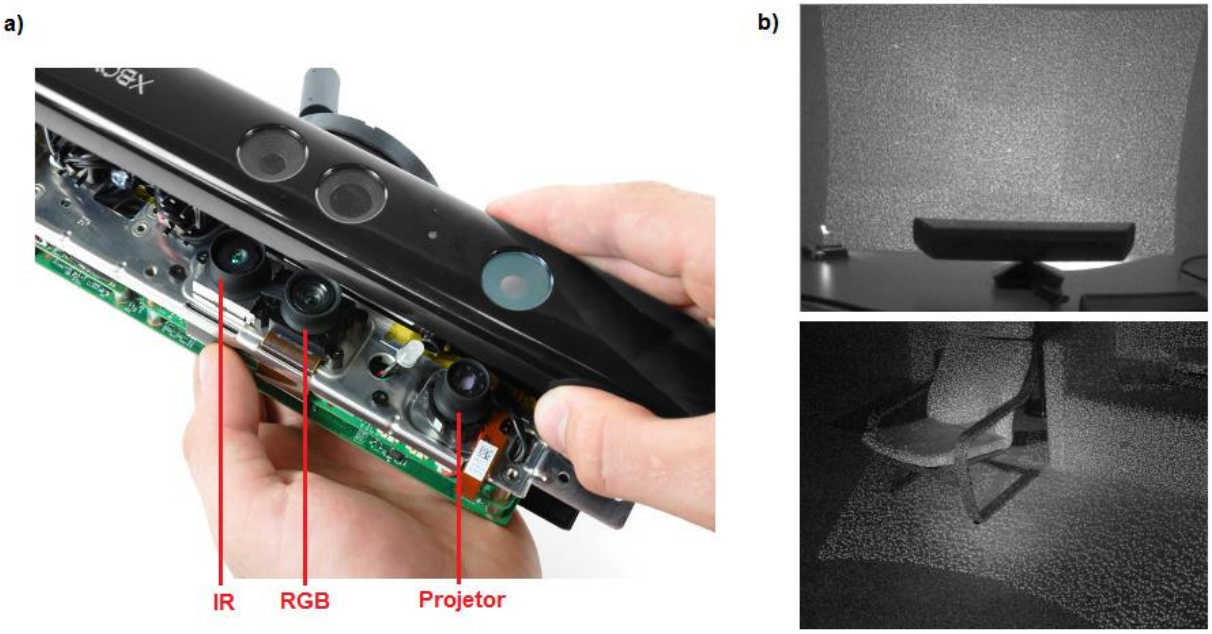


Figura 10 - a) Interior do Kinect mostrando os componentes; b) Dois exemplos de projeção do padrão gerado pelo projetor infravermelho do Kinect.

No Kinect, a câmera de vídeo VGA RGB é utilizada para obter características de cor e textura dos objetos segmentados no mapa de profundidade. Devido a disposição das câmeras e projetor, o mapa de profundidade e a imagem colorida de uma cena tem um deslocamento lateral referente à distância entre a câmera infra e a câmera RGB. Para solucionar este problema, o sistema embarcado da PrimeSense registra esta imagem colorida com o mapa de profundidade, permitindo assim que os pixels de ambas sejam correlacionados com pequena margem de erro.

A PrimeSense, que detém a patente desta tecnologia de estimação de profundidade, também comercializa uma câmera de profundidade de baixo custo chamada *PrimeSensor™ Reference Design*[6] que utiliza apenas um projetor de raios infravermelho classe 1 e uma câmera infravermelho.

Devido a excelente resposta do mercado ao Kinect, a PrimeSense se juntou a ASUS para o desenvolvimento de uma versão do dispositivo para computadores. O projeto se chama Xtion e foi lançado no segundo trimestre de 2011, também com o foco em jogos, porém desta vez disponibilizando uma biblioteca para programação do dispositivo e criação de aplicações diversas para desktops.

[6] Primesense - Reference Design. Disponível em: < http://www.primesense.com/?p=514 >. Acessado em: 25/08/2012.

O Kinect despertou interesse não só dos usuários de jogos, mas também da comunidade científica da área de visão computacional e processamento de imagens. Rapidamente, bibliotecas foram desenvolvidas para controlar o Kinect através do computador. O objetivo é obter acesso à imagem de profundidade, à imagem da câmera VGA e controlar o motor de movimento do sensor. Um exemplo é o OpenKinect[7] um projeto colaborativo não só para desenvolvimento do driver e API de programação para o Kinect, mas também para desenvolvimento de projetos de interação através de movimentos.

No segundo semestre de 2011 a Microsoft se rendeu ao grande interesse da comunidade de desenvolvedores, e lançou uma SDK[8] oficial para computadores, onde o programador pode controlar todas as funcionalidades do dispositivo incluindo a obtenção da modelagem de partes do corpo humano dos usuários se movendo em frente ao dispositivo.

2.8 Decisões de projeto e Resultados

A fase de segmentação no contexto de reconhecimento de Libras para este projeto deve atender aos requisitos a seguir:

- Deve ser capaz de distinguir entre o fundo da cena e o usuário;
- Os objetos segmentados (partes do corpo humano neste contexto) devem ter suas propriedades morfológicas mantidas;
- O sistema deve ser capaz de lidar com oclusões;
- Deve ser passível de execução em tempo real.

Nas próximas subseções serão apresentados os problemas das técnicas utilizando câmeras de espectro visual no contexto de reconhecimento de Libras em tempo real, e posteriormente as soluções adotadas serão apresentadas.

[7] OpenKinect. Disponível em: < http://openkinect.org/wiki/Main_Page >. Acessado em: 25/08/2012.

[8] KINECT SDK. Disponível em: < http://www.microsoft.com/en-us/kinectforwindows >. Acessado em: 15/08/2012.

2.8.1 Detecção de Movimento no Espectro Visual

Todas as técnicas de segmentação de diferenças para detecção de movimento, citadas nas seções anteriores deste capítulo, foram implementadas na busca por um resultado em que a morfologia dos objetos segmentados fosse mantida.

Observando que a maioria dos algoritmos se complementam para obtenção de melhores resultados, um algoritmo de detecção de diferenças foi desenvolvido utilizando uma composição dos algoritmos apresentados neste capítulo.

O algoritmo foi subdividido em dois módulos:

1. **Modelagem de Fundo:** Este módulo é responsável pelo aprendizado do modelo de fundo, sendo atualizado a cada frame capturado pela câmera. Aqui podemos optar no software por três métodos: Running Gaussian Average (Seção 2.2.2); Temporal Median Filter (Seção 2.2.3); Mixture of Gaussians (Seção 2.2.4).
2. **Cálculo do Mapa de Diferenças:** Nesta fase o Mapa de diferenças é obtido através da diferença entre o frame atual e o modelo de fundo atual.

O módulo responsável pelo cálculo do mapa de diferenças é aprimorado com: Detecção de cor de pele utilizando threshold explícito para manter o requisito de tempo real (Seção 2.3.2.1); Eliminação de sombras e highlights (Seção 2.2.5).

O algoritmo funciona da seguinte forma:

1. Os primeiros 60 frames são utilizados para adaptação de um modelo inicial do fundo sendo observado;
2. Captura novo frame, atualiza modelo de fundo e calcula máscara binária de diferenças utilizando um limiar fixo definido pelo usuário;
3. Calcula mapa de pixels com cor de pele na cena observada;
4. Calcula mapa de pixels que são classificados como sombra ou mudança repentina de iluminação (highlight);
5. Percorre simultaneamente as máscaras binárias de diferenças, cor de pele, sombra e highlights e classifica cada pixel como diferença na máscara binária final se ele atender as seguintes condições:
    a. Estar presente na máscara de movimento ou na máscara de cor de pele (excluindo pixels cor de pele que pertencem ao fundo);

b. Estar ausente nas máscaras de sombra e highlights.

Avaliamos a viabilidade de utilização deste algoritmo na etapa de segmentação do nosso sistema de reconhecimento de Libras através de vídeos exemplo. Estes vídeos tentam reproduzir alguns problemas que seriam confrontados em nossas aplicações como sombras, mudanças repentinas de iluminação (highlights), oclusões de partes do corpo humano.

Vamos avaliar aqui dois casos em especial:

- **Experimento 1:** o usuário não aparece no início da cena, aguardando o tempo necessário para o sistema aprender o fundo sem a presença do mesmo;
- **Experimento 2:** o usuário já está em frente à câmera no início da execução e efetua movimentos de oclusão de partes do corpo.

Os resultados do primeiro experimento podem ser observados na Figura 11. Neste primeiro exemplo, temos um fundo composto por móveis e o usuário incialmente posiciona apenas sua mão no campo de visão e depois aparece por completo sentado em uma cadeira. Podemos observar que na primeira cena onde o usuário posiciona a mão somente a frente do fundo homogêneo a segmentação é perfeita. Já na segunda cena podemos observar além do usuário temos a sombra do mesmo projetada no armário acompanhada por uma mudança repentina de iluminação (simulada por uma alteração brusca do tempo de exposição da câmera).

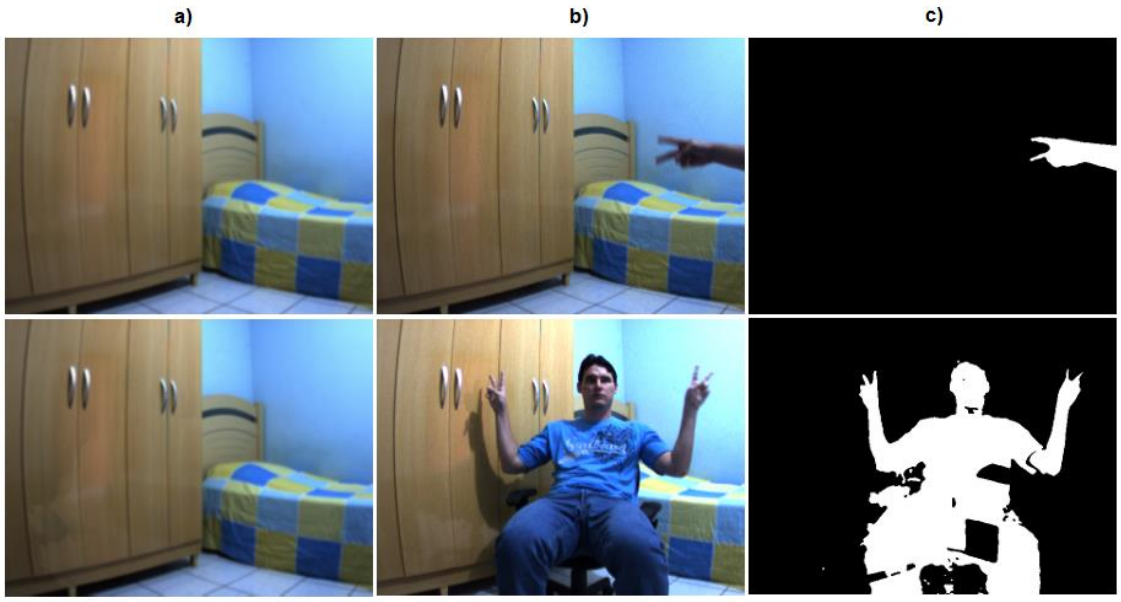


Figura 11 - Resultados obtidos com o experimento 1. a) modelo de fundo; b) imagem corrente; c) resultado final.

O resultado da segunda cena mostra que o algoritmo foi bem sucedido com a eliminação da sombra e não gerou ruído devido a mudança repentina de iluminação (highlight). Porém, este resultado evidencia a falta de precisão na segmentação da silhueta do usuário. No fundo temos uma colcha com quadrados azuis muito próximos da cor da roupa do usuário resultando em falhas na morfologia da silhueta segmentada. Outro problema é como o usuário não faz parte do fundo quando o braço for posicionado em frente ao tórax ou face ele fará parte da silhueta e não poderá mais ser segmentado com este algoritmo.

Analisando agora o experimento 2 podemos observar os resultados na Figura 12. Neste exemplo focamos na análise de dois problemas recorrentes no escopo deste projeto: tratamento de oclusão; e o usuário já estar posicionado em frente a câmera no início da execução do sistema.

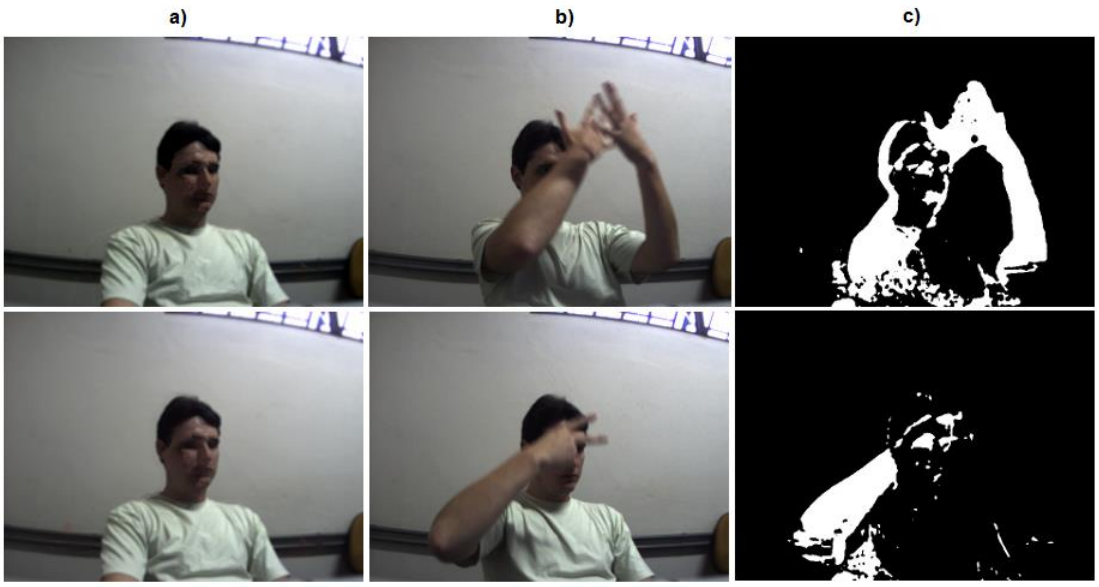


Figura 12 - Resultados obtidos com o experimento 2. a) modelo de fundo; b) imagem corrente; c) resultado final.

Neste exemplo o usuário faz parte do plano de fundo como observado na Figura 12 – a). E podemos ver que o algoritmo não obteve sucesso na segmentação de diferenças no caso de oclusão das mãos com a face. E também evidencia uma série de regiões com ruído devido a movimentação do usuário (efeito fantasma explicado na Seção 2.2.5.3).

Como citado anteriormente o contexto da aplicação de reconhecimento de gestos exige tratamento de oclusão e resposta em tempo real. Ao observar os resultados obtidos, pode-se concluir que não será possível a utilização desta solução sem impor uma série de restrições ao usuário, dificultando a ideia inicial de interação natural com o sistema.

2.8.2 Visão Estéreo no Espectro Visual

Uma solução promissora para tratamento de oclusões é a estimação de profundidade dos objetos observados em uma cena. Com esta informação seria possível separar o fundo dos objetos que realmente importam para a aplicação com um simples threshold.

Como explicado anteriormente na Seção 2.6 a visão estéreo é utilizada para estimar a profundidade utilizando múltiplas câmeras de espectro visual. Para isso esta técnica se baseia em características visuais como cor, bordas, cantos, entre outros.

O estudo comparativo feito por Scharstein [SCHARSTEIN et al., 2002] mostra com detalhes as diferenças entre vários métodos de estimação de profundidade, comparando os mesmos levando em consideração fatores como tempo de processamento e precisão.

Baseando-se nos requisitos de tempo real e segmentação de objetos com integridade morfológica, citados anteriormente, o melhor algoritmo estudado foi o proposto por Hirschmüller [HIRSCHMÜLLER, 2001] que, como exemplificado na Tabela II, tem tempo de processamento médio de 100ms em imagens estéreo com 384x288 de resolução. Uma estimação de execução em uma imagem de 640x480 pixels o algoritmo pode levar no mínimo 270ms para concluir a tarefa de estimação de um único mapa de profundidade de mesma resolução.

	Pixels	Tempo	Frequência
Teste Real	110592px	100ms	10Hz
Estimação	307200px	270ms	3.7Hz

Tabela II - Tempos de execução do algoritmo de estimação de profundidade proposto por Hirschmüller.

Para analisar a precisão do algoritmo de Hirschmüller, Scharstein utilizou três métricas de erro: Pixels em regiões sem oclusão (*B.O.*); Pixels em regiões sem textura (*B.T.*); Pixels em regiões de descontinuidade de profundidade (*B.D.*). Os resultados foram: *B.O. = 4,25%; B.T. = 4,47% e B.D. = 15,05%* (quanto mais próximo de zero forem as métricas, melhor e mais preciso é o resultado final). Um exemplo de resultado da estimação deste algoritmo pode ser visto na Figura 13 item c). Comparando com o resultado ideal no item d), podemos ver várias descontinuidades dos objetos segmentados, falhas morfológicas e ruídos de borda.

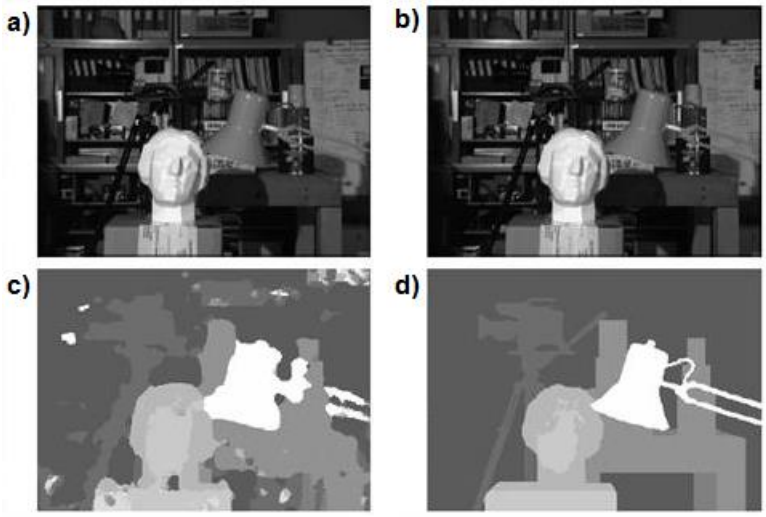


Figura 13 - Resultado obtidos pelo algoritmo com maior custo benefício *Tempo de processamento* X *Precisão.*

Portanto, levando em consideração o tempo de processamento que inviabiliza atender o requisito de tempo real e também a imprecisão do resultado da segmentação, a técnica de estimação de profundidade através do espectro visual não será utilizada em nossa solução final.

2.8.3 Solução Proposta: Virtual Wall

Como descrito e exemplificado ao longo deste capítulo, as soluções que utilizam câmeras do espectro visual não conseguem atender aos requisitos necessários para uma segmentação passível de utilização sem fatores limitantes, como controle da iluminação ambiente, fundo controlado, e principalmente oclusões entre partes do corpo humano.

As soluções utilizando câmeras do espectro visual focam na detecção de movimento e modelagem do fundo para extrair os objetos de interesse da cena. Já

nas soluções com câmeras de profundidade, isto não é mais necessário, visto que apenas com a informação de profundidade conseguimos eliminar o fundo.

Portanto, utilizando câmeras de profundidade descritas na Seção 2.7 podemos facilmente diferenciar entre o fundo da cena e o usuário apenas definindo regiões de interesse como barreiras virtuais.

Este simples, porém eficiente algoritmo foi chamado de Virtual Wall e consiste na definição de uma referência espacial (distância conhecida) para eliminar o fundo da cena como definido a seguir:

$$MP(d) = \begin{cases} fundo, & d \geq \tau \\ objetos, & d < \tau \end{cases}$$

Onde MP é o mapa de profundidade resultante, d é o valor de um pixel no mapa e τ é o limiar escolhido que definirá a posição da barreira virtual em relação a câmera de profundidade.

O limiar ou distância conhecida (τ) pode ser definido dependendo da aplicação, podendo ser um valor fixo ou um valor calculado em tempo real como a posição da face do usuário ou o centro de massa do mesmo, por exemplo.

Neste projeto optamos por utilizar o centro de massa do usuário como referência. No Capítulo 3, mais especificamente na Seção 3.8, a justificativa para esta escolha e o cálculo do centro de massa do usuário serão explicados com detalhes.

Capítulo 3

MODELAGEM E TRACKING DE PARTES DO CORPO HUMANO

Neste capítulo uma revisão bibliográfica sobre modelagem e tracking de partes do corpo humano é apresentada. Posteriormente será apresentada a solução proposta nesse trabalho para detecção e tracking das mãos juntamente com a modelagem das mesmas em um vetor de características.

3.1 Considerações Iniciais

Segundo [MOESLUND et al., 2006] a modelagem e reconhecimento do movimento humano é uma área de pesquisa muito ativa devido as várias possíveis aplicações. Esta área tenta resolver problemas de alta complexidade como estimar a posição e movimento de partes articuladas com alto grau de liberdade e que podem se sobrepor.

As aplicações desta área de pesquisa podem ser agrupadas em três categorias: Aplicações de Segurança, Aplicações de Controle e Aplicações de Análise.

Aplicações de segurança são as mais clássicas na análise de movimentos e consistem em sistemas de vigilância e controle de estacionamentos, estabelecimentos, rodovias, etc. Analisam fluxo de pessoas, carros, e mais recentemente até análise de comportamento para identificar atitudes suspeitas no ambiente em que o sistema está inserido.

Aplicações de controle são aplicações em que a modelagem e identificação da postura é utilizada para controlar interfaces humano-computador como jogos ou realidade-aumentada, um dispositivo eletromecânico como um atuador, ou até para modelagens reais de movimentação para efeitos especiais em filmes com computação gráfica.

Aplicações de análise utilizam a informação obtida para analisar, por exemplo, o desempenho de atletas e otimizar seus resultados através de relatórios detalhados de suas falhas e pontos fracos durante uma sessão de treinamento. Também são utilizadas para auxiliar em diagnósticos ortopédicos, e também para análise comercial para descobrir quantas pessoas se interessaram por determinado produto ou simplesmente o fluxo de pessoas em uma loja.

O foco deste capítulo são as aplicações de controle, e algumas soluções do estado da arte para modelagem de partes do corpo humano em movimento serão explicadas a seguir. Posteriormente serão apresentadas as decisões de projeto e a solução proposta para esta fase do sistema de reconhecimento de gestos.

3.2 Características dos métodos de modelagem e tracking

Moeslund [MOESLUND et al., 2001] definiu uma taxonomia das funções dos algoritmos de modelagem e tracking de humanos que será descrita a seguir:

- **Inicialização:** Garantir que o sistema inicie sua operação com uma correta interpretação do ambiente;
- **Tracking:** Segmentar e efetuar tracking de humanos em um ou mais frames capturados da câmera;
- **Estimação da Postura:** Estimar a posição e postura de humanos em um ou mais frames, ou seja, onde estão na imagem cada uma das partes do corpo como cabeça, tronco, braços e pernas.
- **Reconhecimento:** Reconhecer ações, comportamentos, e atividades de um ou mais indivíduos em um ou mais frames capturados.

A função de *Inicialização* consiste na definição e utilização de um modelo humanóide que define as características estruturais e de movimento. Muitos autores

se basearam na inicialização manual das estruturas, o que vem sendo revertido por pesquisas mais recentes que tentam obter uma inicialização automática com o mínimo de informação a priori possível. Esta informação a priori é o que define a fase de inicialização para facilitar as fases posteriores de tracking e estimação da postura. Tais informações podem ser uma estrutura cinemática, um modelo 3D, aparência de cor, modelo de postura mais provável, relações dimensionais entre as partes do corpo, entre outros.

Tracking foi mais comumente utilizado para aplicações de vigilância em ambientes internos e externos, para identificar humanos e determinar sua posição ao longo do tempo. Esta função pode ser subdividida em duas etapas, sendo elas a etapa de segmentação que consiste na separação dos humanos do fundo da cena, e a etapa de identificação temporal que consiste em rotular no frame atual o humano que estava presente no frame anterior e pode ter se movimentado, para assim obter o padrão de movimentação de cada humano previamente segmentado na etapa inicial.

Estimação da Postura é a fase onde a configuração do esqueleto ou estrutura cinemática é estimada. Este processo pode ser integralmente executado pela fase de tracking (que após a primeira estimação da configuração, pode ser capaz de rastrear o posicionamento da estrutura em frames posteriores) ou executado frame a frame. Os algoritmos de estimação de postura podem ser subdivididos em três categorias:

- **Sem modelo a priori:** São os algoritmos que não utilizam nenhum modelo explícito para estimação da postura;
- **Utilização indireta do modelo:** Estes algoritmos utilizam um modelo a priori como referência para guiar a estimação da postura. Geralmente são métodos 2D que rotulam as partes do corpo obtendo o chamado cardboard model (Estrutura definida por quadriláteros que indicam a região que contém as partes do corpo, e suas interconexões);
- **Utilização direta do modelo:** Estes algoritmos utilizam um modelo 3D explícito que define a estrutura cinemática e forma humana para efetuar a reconstrução da postura a partir dos dados lidos.

Reconhecimento é a etapa onde ocorre uma interpretação dos dados obtidos nas fases anteriores, para determinar as atividades ou ações que estão ocorrendo no

ambiente inspecionado. O reconhecimento e modelagem de ações é diretamente relacionado ao objetivo e tipo da aplicação. Várias aplicações podem utilizar reconhecimento de ações como vigilância, recuperação ortopédica, análise de desempenho de atletas, interação com jogos, reconhecimento de língua de sinais, entre outras.

3.3 Marcadores Coloridos ou Refletivos

A utilização de marcadores coloridos ou refletivos simplifica a segmentação, e também a modelagem, pois os pontos de interesse do corpo humano estarão identificados por marcadores (podendo ser diferenciados para cada parte) e a partir deles a estimação da postura se torna uma tarefa mais simples.

Porém, no meio acadêmico esta solução não é bem aceita, pois elimina a naturalidade da comunicação e restringe a interação do usuário. Mas muitos sistemas comerciais para o setor médico ou esportivo utilizam esta técnica por sua simplicidade e maior garantia de precisão.

Cole [COLE et al., 2007] utilizou marcadores coloridos diferenciados para cada parte ou pontos específicos do corpo humano, sendo que o foco de seu trabalho era o controle da movimentação de robôs para que a mesma seja similar a humana. Seu sistema estima a pose humana através da identificação dos marcadores coloridos, e envia comandos de manipulação dos atuadores de movimentação para modificar o posicionamento do robô como exemplificado na Figura 14.

A Vicon[9] possui uma linha de produtos, para análise e captura de movimentos, voltada para vários nichos de mercado como animação, medicina, esportes, entre outros. Dentre seus variados produtos o Motus[10] é uma ferramenta de análise de movimento que pode fazer tracking de patterns manualmente selecionadas quadro a quadro. O Motus também permite a utilização de um módulo

[9] VICON. Empresa fornecedora de software e hardware para Motion Analysis. Disponível em: < http://www.vicon.com >. Acessado em: 25/08/2012.

[10] MOTUS - VICON Motion Analysis Software. Disponível em: < http://www.vicon.com/products/motus.html >. Acessado em: 26/08/2012.

adicional de tracking automático de marcadores refletivos, utilizando múltiplas câmeras infravermelhas, para captação dos movimentos com precisão milimétrica.

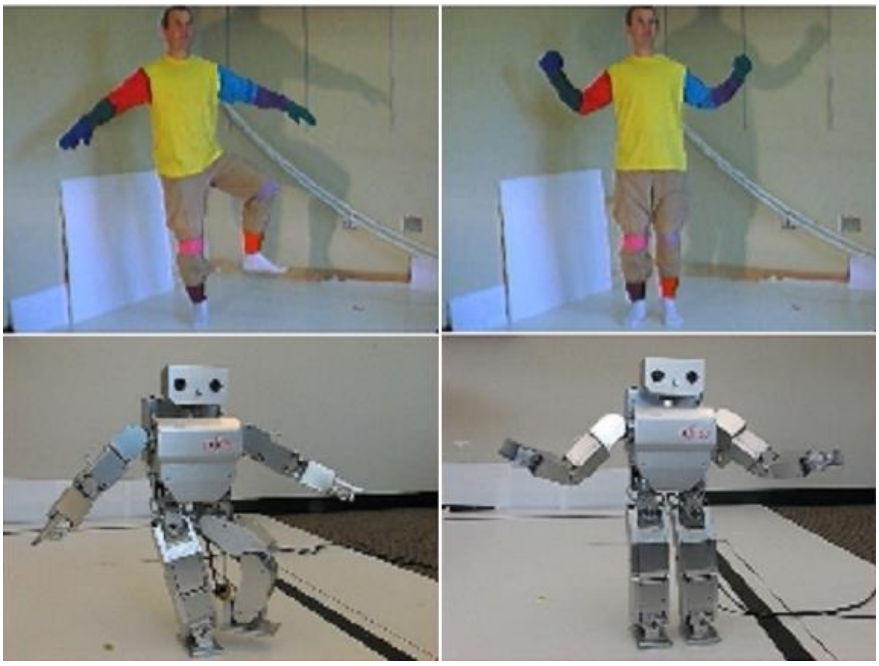


Figura 14 - Exemplos de replicação da posição humana em um robô, utilizando marcadores coloridos.

3.4 Sensores de posição e acelerômetros

Sensores de posição ou os chamados sensores 3D, juntamente com acelerômetros e giroscópios eletrônicos, são um tipo de hardware comumente utilizado para sistemas de alta precisão com objetivos como reconstrução de superfícies ou mapeamento de movimento. Estas soluções costumam ser de alto custo, possuem alta precisão, mas tem a desvantagem de ter de usar os sensores pelo corpo, eliminando assim a naturalidade na comunicação através de gestos ou movimentos corporais.

Allen [ALLEN et al., 2002] utilizou dados capturados de redes de sensores distribuídos pelo corpo para mapear as deformações de superfície na parte superior do corpo humano e obter um modelo 3D preciso da superfície do corpo utilizando como guia um modelo cinético pré-definido.

Yang [YANG et al., 2008] utilizou uma rede de sensores equipados com acelerômetros e giroscópios eletrônicos para modelar um sistema de

reconhecimento de 12 ações humanas distintas, utilizando oito destes sensores em partes do corpo humano. Com este arranjo de hardware e software o sistema foi capaz de reconhecer as 12 ações com uma taxa de acerto de 98.8%.

3.5 Pictorial Structures

Este é um método de modelagem estatística, onde um modelo do corpo humano é pré-definido através de treinamento, e utilizado para buscar nas imagens a configuração (postura) desta estrutura.

Segundo [IVANOVS, 2007] este modelo chamado de Pictorial Structures é definido através da subdivisão do objeto a ser modelado em partes deformáveis. Esta estrutura pode ser representada através de um grafo $G = (V, E)$, onde os vértices $V = \{v_0, \dots, v_{n-1}\}$ do grafo correspondem as n partes da estrutura, e para cada par de partes conectadas v_i e v_j existe uma aresta $(v_i, v_j) \in E$. Cada parte é defininda por uma configuração $l = (l_0, \dots, l_{n-1})$, onde l_i especifica a localização da parte v_i. Este vetor de localização pode ser definido como:

$$l_i = (x, y, s, \theta)$$

Onde x e y definem a posição da parte relativa à parte imediatamente superior na árvore, s define a escala ou dimensão da parte (como dito anteriormente a parte é deformável, e pode sofrer alteração em suas dimensões) e θ define o ângulo formado entre a parte e sua conexão com a parte superior.

Muitos métodos de modelagem se baseiam na identificação individual das partes do corpo antes de agrupá-las para definir a estrutura como um todo, e isso exige que o identificador das partes seja de alta complexidade. O modelo utilizando Pictorial Structures reduz essa complexidade utilizando o modelo de interconexão deformável entre pares de partes, ou seja, as partes não são buscadas individualmente, mas sim agrupadas aos pares com a característica de posicionamento e angulação relacional, simplificando assim os descritores de cada parte.

O problema de efetuar o matching de uma Pictorial Structure em uma imagem pode ser definido como um problema de minimização de uma função de energia. Considerando $m_i(l_i)$ uma função que calcula o grau de incompatibilidade quando a

parte v_i está na posição l_i na imagem, $d_{ij}(l_i, l_j)$ uma função que mede o grau de deformação entre as partes v_i e v_j quando elas estão localizadas nas posições l_i e l_j respectivamente, a função que calcula o matching ótimo para a imagem pode ser definida como:

$$l^* = \arg\min_{l} \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) \right)$$

Até agora foram mencionadas somente características de posicionamentos relacionais e estruturais do modelo, mas também devem ser levadas em consideração na tarefa de matching, as características internas das partes. Estas características podem ter qualquer tipo de informação desejada, desde que seja formatada uma função de custo para estimar regiões prováveis na imagem. A média de cor da região exemplifica um tipo de característica.

Um exemplo presente em [IVANOVS, 2007] de um par de partes de uma estrutura pode ser visto na Figura 15. A parte retangular poderia corresponder a um antebraço e a parte em forma de trapézio a mão. Em a) o círculo vermelho e azul completamente preenchidos indicam a possibilidade de conexão de cada parte a outras partes, e o círculo azul vazio indica que a parte retangular permite se conectar a uma parte "filha" na árvore estrutural. Em b) é mostrado um exemplo de conexão em ângulo entre as partes.

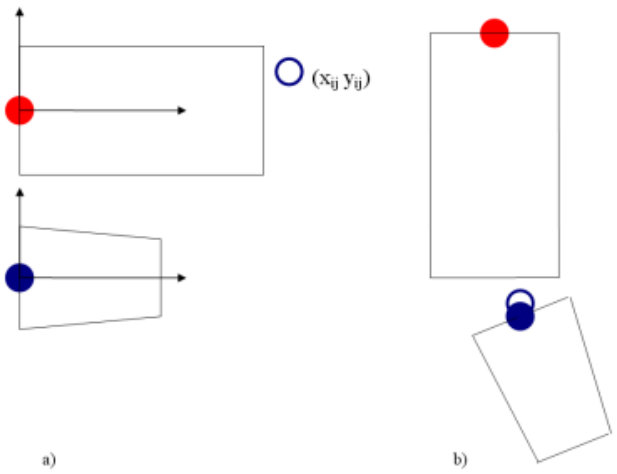


Figura 15 - Exemplo de Pictorial Structures

A principal desvantagem deste método é a complexidade do problema de minimização da função de energia. Mesmo limitando o grau de liberdade das

conexões entre as partes e simplificando o modelo de aparência das mesmas, estes sistemas podem não manter a mesma eficiência e capacidade de generalização sem perder a possibilidade de resposta em tempo real.

Felzenszwalb [FELZENSZWALB et al., 2005] utilizou Pictorial Structures para reconhecer faces, utilizando sete partes, e o corpo humano, utilizando dez partes (cabeça, tronco, duas partes para cada braço e duas partes para cada perna). Para formatar os modelos foi proposta uma forma de treinamento supervisionado (onde as partes eram marcadas manualmente nas imagens de treinamento) que obtém como resultado o modelo a ser buscado, não necessitando de uma formatação manual do mesmo. A estimação de pose foi executada no resultado obtido de um algoritmo de Background Subtraction.

Andriluka [ANDRILUKA et al., 2009] aplicou o conceito de Pictorial Structures para detectar pessoas e estimar a pose das mesmas em imagens de alta resolução. Em contraste com a pesquisa anterior, em que os modelos de aparência das partes eram simples descritores de forma aplicadas em imagens resultantes da subtração de fundo, foram utilizados modelos robustos de identificação de características com treinamento de classificadores AdaBoost para cada parte. O autor cita como desvantagem de seu método os casos de oclusão de partes nos quais seu sistema não consegue estimar com precisão a postura.

3.6 Modelagem através da silhueta

A silhueta humana pode ser obtida tanto de sistemas de visão estéreo para estimação da distância dos pontos da imagem a câmera, quanto de um simples sistema monocular utilizando Background Subtraction para segmentar regiões em movimento. Muitos trabalhos foram desenvolvidos utilizando as características de forma da silhueta humana para modelar e estimar a postura humana.

Zhao [ZHAO, 2001] propôs um método de modelagem da forma humana e estimação da postura utilizando um sistema de visão estéreo para segmentar as partes do corpo através da silhueta humana. O objetivo de seu trabalho foi desenvolver um sistema genérico capaz de superar os seguintes problemas: oclusão de partes; ser tolerante as variações na forma causadas pelas roupas que

atrapalham a tarefa de segmentação e identificação de partes; ser invariável a rotação, translação e escala; detectar articulações (joints) com alta precisão sem a utilização de marcadores.

Partindo do pressuposto que os contornos obtidos podem ter sido afetados por ruído, partes ocultas, roupas, entre outros fatores, Zhao desenvolveu um algoritmo chamado RCR, ou Recursive Context Reasoning, para solucionar os problemas de detecção da postura humana através da silhueta.

A primeira etapa de seu algoritmo corresponde à segmentação e decomposição do contorno como exemplificado na Figura 16, e consiste inicialmente na obtenção da silhueta (a), seguida de uma aproximação do contorno utilizando uma curva B-Spline para eliminação de ruídos (b) com cálculo dos pontos de curvatura negativa mínima (Negative Curvature Minima – NCM). Utilizando estes pontos de interesse e algumas regras quanto à proporção das partes ele define a regra do melhor “corte” que tenta subdividir a silhueta em partes significativas (c). E finalmente um algoritmo de refinamento para escolher a melhor subdivisão é aplicado (d).

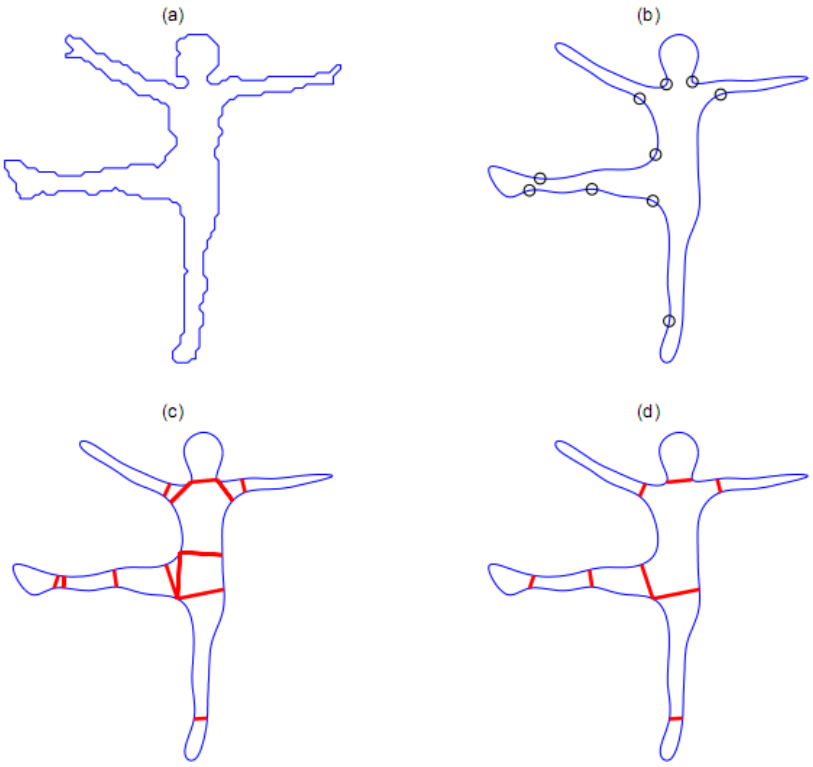


Figura 16 - Exemplo de Segmentação e decomposição da silhueta em partes.

A segunda etapa, ilustrada na Figura 17, consiste na rotulação das partes segmentadas de acordo com o modelo hierárquico do corpo humano previamente estabelecido (e). A partir do resultado da etapa anterior (a) eliminam-se subdivisões similares para obter um modelo com menos informações (b). A partir desta segmentação menos detalhada, as partes são rotuladas em um nível hierárquico mais alto no modelo do corpo humano (e), obtendo uma rotulação (c) menos granular. Finalmente, as partes rotuladas em (c) que são constituídas de sub-partes no modelo hierárquico, são rotuladas a partir do resultado de segmentação rico em detalhes (a). Esta estimação da postura é feita através de modelos probabilísticos que levam em consideração não só o modelo hierárquico que evidencia as interconexões entre as partes, como também aspectos de forma e proporção, o espaço de possibilidades é buscado com o objetivo de minimizar uma função de energia.

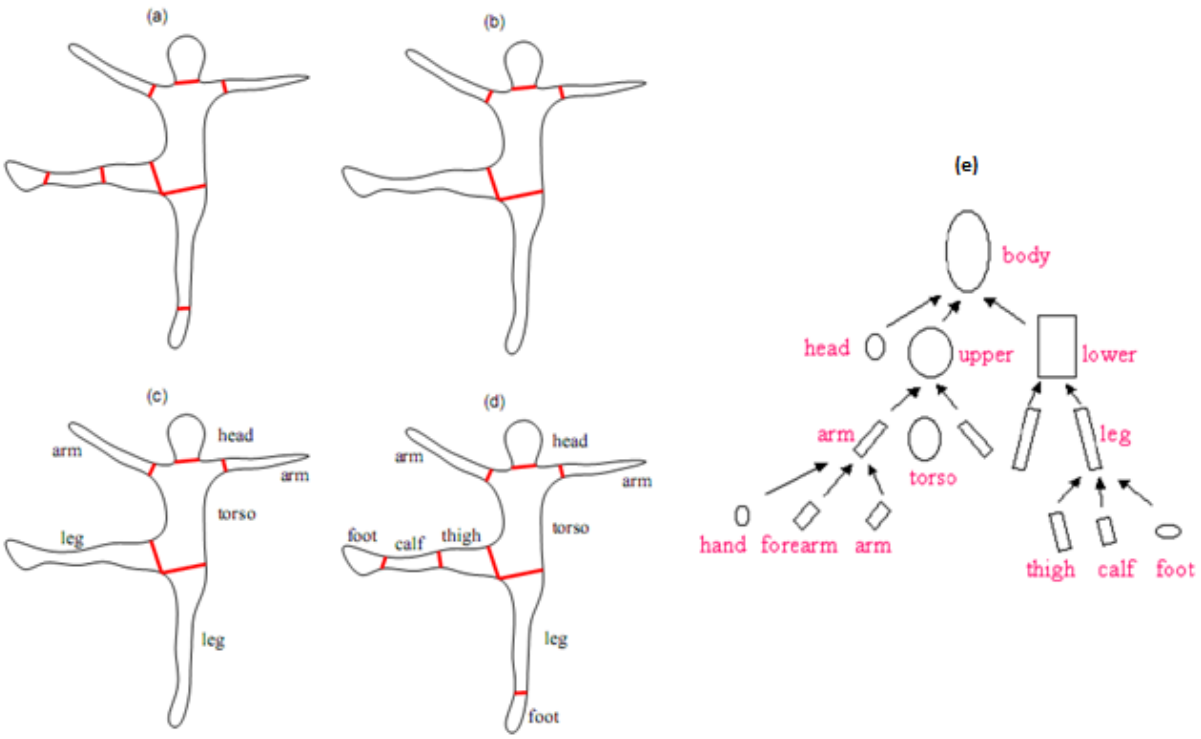


Figura 17 - Exemplo de rotulação das partes segmentadas segundo o modelo hierárquico.

A terceira etapa consiste na tentativa de adaptar o modelo estimado às partes detectadas, e estimar o posicionamento das partes ocultas. Esta etapa é exemplificada na Figura 18. Este exemplo foi executado em um ambiente com fundo complexo, e a partir de um sistema estéreo o contorno é extraído em (a). Pode-se

observar que o contorno de um dos braços não está completo, e que as pernas não foram segmentadas devido à baixa diferença de intensidade entre a cor da calça e o fundo. Deste contorno incompleto três partes são identificadas em (b). Em (c) o modelo de postura estimado é alinhado ao contorno, as partes ocultas (um braço e pernas) tem a mesma orientação do tórax (torso) e são posteriormente alinhadas em (d) utilizando como guia um algoritmo de Edge Detection.

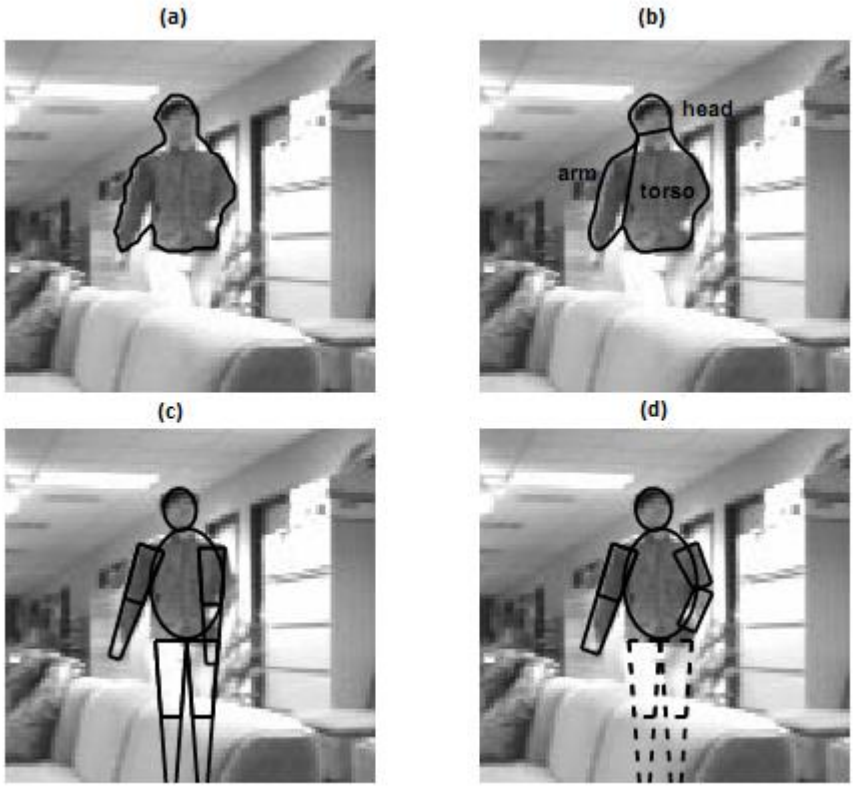


Figura 18 - Exemplo de segmentação e identificação de partes visíveis e ocultas em imagem de baixa resolução e fundo complexo.

Zhao testou o funcionamento de seu sistema na detecção de pedestres e suas atividades acoplando um sistema estéreo em um ônibus. Seu sistema foi capaz de reconhecer pedestres e suas posturas com uma velocidade de um a seis frames por segundo.

Hu [HU et al., 2000] propôs um método que utiliza um modelo paramétrico humano em 2D para estimar a postura humana. Esta tarefa foi feita em duas etapas, sendo a primeira a extração da silhueta em movimento através de um algoritmo de Background Subtraction. Na segunda etapa um algoritmo genético é utilizado para relacionar a silhueta segmentada e o modelo paramétrico humano. Para reduzir o

espaço de busca, uma modelagem hierárquica parecida com a proposta em [ZHAO, 2001] é utilizada. Os experimentos demonstraram a funcionalidade do sistema para extrair posturas humanas de vídeos.

Mittal [MITTAL et al., 2003] se baseou no trabalho de Zhao [ZHAO, 2001] para criar um sistema capaz de estimar posturas e fazer tracking das partes do corpo de várias pessoas em um ambiente fechado, utilizando múltiplas câmeras. Seu método tem como vantagens: Não necessita de inicialização manual; não precisa de especificação das dimensões da estrutura 3D do modelo humano; não exige a utilização de poses previamente modeladas ou padrões de atividade como guia de calibração; não é sensível a bordas e fundos complexos. O algoritmo foi capaz de segmentar múltiplas pessoas em situações de oclusão em uma velocidade de 10 frames por segundo.

Takahashi [TAKAHASHI et al., 2004] utilizou um algoritmo de Background Subtraction para segmentar a silhueta humana e Redes Neurais para o aprendizado das posturas. A rede neural tem como entrada as características que são extraídas da silhueta para formatar um feature vector. A saída desta rede neural são as posições de cada uma das partes do corpo humano que serão detectadas (cabeça, ombros, mãos, cotovelos, joelhos e pés). Após o processamento da rede neural, a postura é representada visualmente por um grafo. O tempo de processamento deste método foi de 20 frames por segundo.

3.7 Modelagem utilizando imagens de profundidade

Imagens de profundidade podem ser obtidas através de câmeras em arranjo estéreo ou câmeras de profundidade que utilizam luz não visível. Ambas as formas foram descritas no capítulo anterior. Câmeras em arranjo estéreo não possuem precisão suficiente para um sistema de modelagem e estimação de postura eficiente em tempo real, e como as câmeras de profundidade são uma tecnologia recente esta área de pesquisa ainda não possui muitas publicações.

Um mapa de profundidade facilita tanto na detecção de regiões da imagem com movimentação quanto na modelagem de partes do corpo humano. A principal vantagem que facilita a modelagem é quanto à sobreposição de partes do corpo,

como por exemplo, braço e tronco, que seria facilmente detectada e segmentada com a característica de profundidade. Mas problemas como oclusão de partes, que desaparecem do campo visual, ou partes mescladas (que podem ser exemplificadas por um braço próximo ao tórax se tornando uma região homogênea em relação à profundidade) continuam presentes nestes arranjos de hardware, e precisam ser tratados pelos algoritmos de estimação da postura humana.

Para modelar e estimar a postura pode-se utilizar a silhueta, obtida de um simples threshold do mapa de profundidade para segmentar as regiões mais próximas da câmera, e aplicar os algoritmos citados na seção anterior. A única vantagem deste procedimento é a simplificação da fase de segmentação, mas o hardware será subutilizado, pois a característica de profundidade de cada pixel não será aproveitada na modelagem.

Alguns exemplos recentes de modelagem que utilizam as informações de mapas de profundidade serão apresentados a seguir.

Zhu [ZHU et al., 2008] propôs um método de modelagem 3D a partir de imagens de profundidade obtidas de uma câmera MESA SR4000 que utiliza uma tecnologia do tipo *time of flight* que consiste basicamente na estimação do tempo entre a emissão da luz e a recepção da luz refletida pelo objeto.

O método de Zhu foi implementado com uma modelagem híbrida utilizando um modelo paramétrico previamente estabelecido e um modelo obtido através da aprendizagem para estimar a postura da parte superior do corpo humano. Seu sistema é capaz de estimar pontos de interesse, tentar se recuperar de problemas de oclusão ou múltiplos candidatos detectados para cada parte, evitar rotações não naturais das juntas e não sofrer sobreposição de partes (self-penetration problem) no mapeamento para um sistema de coordenadas 3D.

Em sua dissertação de doutorado, Zhu [ZHU, 2009] aprimorou seu sistema para a estimação de postura para todo o corpo humano utilizando heurísticas de recuperação de erros ocasionados por oclusão de membros, separadas para a parte superior e parte inferior do corpo, como exemplificado pela Figura 19. Este trabalho foi testado exaustivamente em variadas condições e novas heurísticas para recuperação de erro foram propostas em [ZHU et al., 2010].

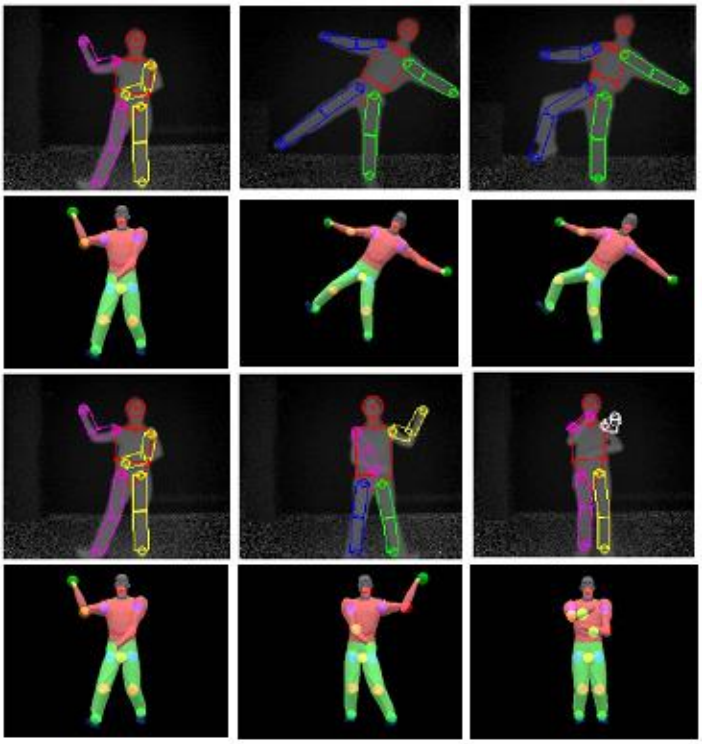


Figura 19 - Exemplo de identificação de partes sobrepostas ou não e correspondente modelagem em 3D.

Plagemann [PLAGEMANN et al., 2010] propôs um método livre de modelo, que foca na identificação de partes do corpo através do agrupamento de nuvens de pontos de profundidade em blobs e identificação de pontos de interesse. Estes pontos de interesse são descritos com um feature vector que é utilizado para estimar se a região em que está contido é uma parte do corpo ou não. Seu sistema é limitado ao tracking e identificação da cabeça, mãos e pés. Mas não trata oclusões ou restrições relacionadas ao posicionamento relativo entre as partes do corpo.

Jain [JAIN et al., 2010] propôs um modelo que utiliza múltiplos métodos para modelar o corpo humano. Seu sistema utiliza duas câmeras uma de profundidade e uma câmera digital. As imagens obtidas são registradas (alinhadas) e utilizadas para segmentação de objetos em movimento. Uma vez efetuada a segmentação, é utilizado um detector de objetos com *Haar-Like Features* que é treinado com algoritmo AdaBoost para detecção do tronco e cabeça. Após esta detecção os braços e suas subdivisões são detectados independentemente utilizando um método de esqueletização e um modelo paramétrico para detectar os pontos de interesse. O resultado final é um sistema capaz de modelar a parte superior do corpo de múltiplas pessoas em cenas sem casos de oclusão a 15 quadros por segundo.

3.8 Decisões de Projeto e Resultados

Para a fase de modelagem e tracking das partes do corpo humano no contexto da Libras os requisitos são:

- Identificar o usuário que esteja em frente ao equipamento;
- Identificar as mãos e obter as máscaras binárias correspondentes à região de interesse das mesmas;
- Não cometer erros de classificação das mãos, por exemplo, segmentando um objeto estranho ou ruído de segmentação como uma mão;
- Extrair características das mãos presentes;
- Ao ser acrescentada como etapa posterior a de Segmentação manter o sistema passível de execução em tempo real.

Os métodos apresentados anteriormente tanto de espectro visual quanto no espectro infravermelho não atendem ao requisito de tempo real, e principalmente não são capazes de atender ao requisito de não cometer erros de classificação. Apesar de alguns métodos como Zhu [ZHU et al., 2010] e Zao [ZHAO, 2001] apresentarem algoritmos para recuperação de erros de estimação, eles ainda são passíveis de erro e podem até não conseguir se recuperar de um erro de oclusão mais acentuada e propagar este mesmo erro no decorrer das estimações.

Nas próximas seções serão apresentadas alternativas, utilizando as câmeras de profundidade, para modelagem das partes do corpo humano, e também a solução proposta para este projeto.

3.8.1 Kinect SDK e PrimeSense Nite

A Microsoft disponibilizou recentemente uma SDK [KINECT SDK] para desenvolvimento de aplicativos para computadores utilizando o Kinect e Windows. Com esta SDK o desenvolvedor pode manipular diretamente as imagens geradas pelo dispositivo, capturar áudio através dos microfones embutidos e também usufruir dos algoritmos de segmentação de usuários e modelagem de partes do corpo humano.

O algoritmo para modelagem desenvolvido pela Microsoft não exige nenhum tipo de calibração inicial e consegue estimar 20 posições pré-definidas do corpo humano. Porém, este algoritmo foi inicialmente desenvolvido para uma estimação do corpo completo do usuário, o que ocasiona problemas quando o usuário se encontra próximo ao dispositivo e não tem seu corpo totalmente visível na área de interação.

Este problema é exemplificado na Figura 20 onde podemos observar dois casos de falha de estimação. A coluna a) mostra os mapas de profundidade com os pixels do usuário em destaque, a coluna b) mostra a falha de estimação da postura que pode ser vista na coluna c).

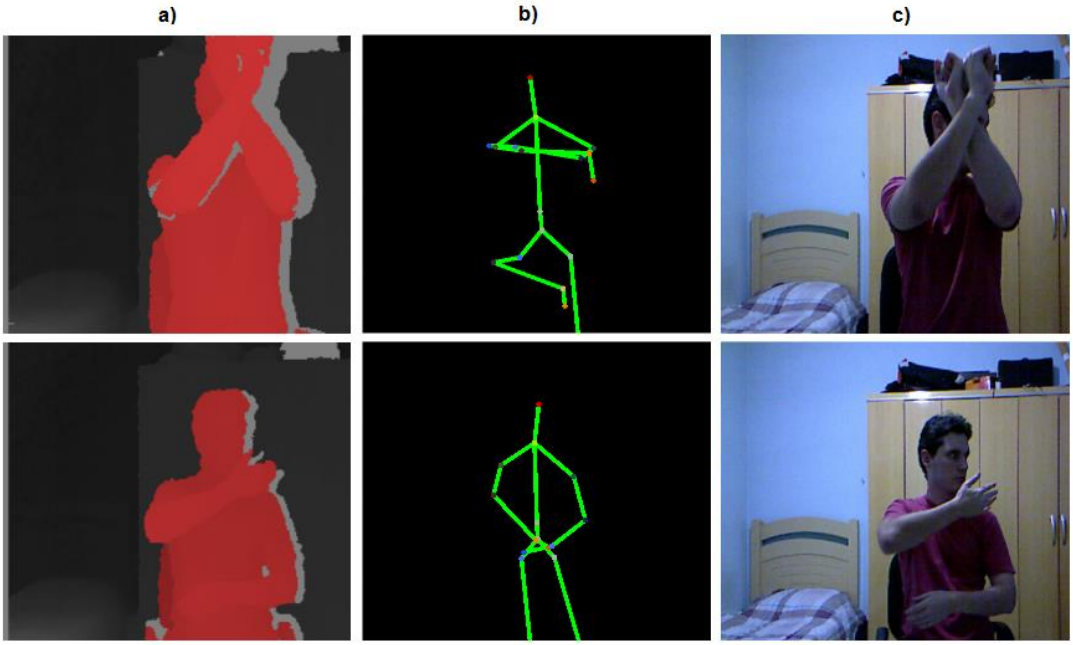


Figura 20 - Problemas de estimação com o Kinect SDK. a) Mapa de profundidade; b) Estimação das partes do corpo; c) Imagem Colorida.

A PrimeSense também oferece uma solução para desenvolvedores que é o Middleware chamado Nite. Este middleware é multiplataforma e permite detecção de usuários na cena e obtenção de uma modelagem de partes do corpo humano.

O algoritmo disponível no NITE para obter um descritor do corpo humano consegue classificar 24 pontos de interesse (joints). Porém ele necessita de uma pose inicial de calibração, chamada de Pose T. Todo novo usuário que entrar na cena deve realizar esta pose para calibrar o algoritmo e iniciar as estimações.

O problema desta calibração no contexto da nossa aplicação acontece quando o usuário tenta executar a pose de calibração estando sentado. O sistema

nem sempre é capaz de reconhecer a pose obrigando o usuário a se levantar e afastar do equipamento para que a calibração ocorra com sucesso.

Além da posição de calibração, o sistema também comete erros de estimação quando o usuário não está totalmente visível na área de interação. Três exemplos de falhas de estimação dos pontos de interesse, que formam um esqueleto do usuário, são mostrados na Figura 21.

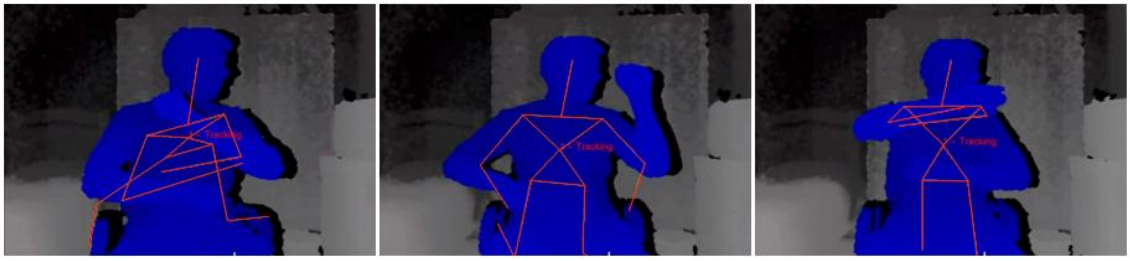


Figura 21 - Problemas de estimação utilizando o PrimeSense Nite.

3.8.2 Solução Proposta no escopo da Libras

Para simplificar o processo de interação com o sistema e evitar possíveis problemas na descrição do gesto sendo executado pelo usuário, optamos por simplificar o processo de modelagem focando nas características do domínio da Libras e não mais tentando estimar a posição das partes superiores do corpo humano e suas articulações.

O contexto da Libras será explicado com mais detalhes na Seção 4.4, mas para definir as decisões de projeto nesta fase de modelagem alguns conceitos serão descritos nesta seção.

Os gestos dinâmicos da Libras são executados utilizando a mão predominante (direita para destros e esquerda para canhotos) ou ambas as mãos sempre a frente do corpo. Em alguns casos expressões faciais ou movimentos da cabeça também fazem parte do gesto.

Neste projeto focaremos somente no movimento das mãos como ponto de partida para iniciar trabalhos de reconhecimento da Libras.

Dentro deste escopo o algoritmo utilizado neste projeto para modelagem e tracking das mãos será descrito nas próximas seções. Esta explicação será dividida em três fases: 1) Adaptação do algoritmo Virtual Wall para se adaptar dinamicamente a posição do usuário; 2) Descrição do método de extração das mãos

do usuário; 3) Descrição do método de extração de características das mãos segmentadas.

3.8.3 Virtual Wall Dinâmico

Na fase de segmentação utilizamos o algoritmo de segmentação Virtual Wall definido na Seção 2.8.3 para segmentar as mãos como sendo quaisquer objetos que ultrapassem a barreira virtual definida a frente do usuário.

Para definir a posição desta barreira virtual precisamos definir uma distância conhecida do usuário a câmera que pode ser calculada a cada novo frame possibilitando assim que a barreira virtual esteja sempre a frente do usuário mesmo que ele se movimente.

Duas possibilidades serão exploradas aqui: Uma seria utilizar a distância da câmera até a face do usuário para definir a posição da barreira virtual, que não pode ser adotada devido a problemas de instabilidade no rastreamento da face; Outra seria calcular a distância do centro de massa do usuário até a câmera.

Nosso sistema adota a segunda opção por problemas encontrados no rastreamento da face. Ambos serão descritos nas subseções a seguir.

3.8.3.1 Reconhecimento e rastreamento da face

Para utilizar a face do usuário como referência para a barreira virtual, propusemos o seguinte algoritmo:

1. Detectar a face utilizando a imagem colorida da câmera RGB do Kinect, utilizar a região que contém a face como entrada para o algoritmo de rastreamento;
2. Obter no mapa de profundidade a localização da região que contém a face detectada e calcular a média de profundidade dos pixels da face, definindo a distância da face até o Kinect;
3. A cada nova imagem capturada recalcular a distância com a nova posição da face.

Para detectar a face utilizamos o algoritmo de detecção de objetos proposto por Viola e Jones [VIOLA et al., 2001] que utiliza Haar-Like Features para encontrar regiões que podem conter uma face. Este algoritmo pode ser executado a cada novo

frame, porém dependendo dos parâmetros utilizados e da resolução da imagem seu custo computacional pode inviabilizar o requisito de tempo real deste processo.

Para contornar este problema utilizamos um algoritmo de rastreamento por características em tempo real chamado Camshift (Continuous Adaptative Mean Shift) [BRADSKI, 1998]. Este método utiliza Mean Shift e Histogram Backpropagation no espaço de cor HSV para encontrar a região que contém o padrão de entrada.

O problema deste algoritmo de rastreamento (tracking) é que ele se baseia nas características de cor da face e no contexto da Libras oclusões entre as mãos e a face são recorrentes. Quando este tipo de oclusão ocorre o algoritmo pode deixar de rastrear a região da face e passar a rastrear a mão, como exemplificado na Figura 22. Na figura a) mostramos o resultado da estimação após detecção da face e início do Camshift. Em b), c) e d) mostramos exemplos de problemas de estimação que ocorreram após o movimento da mão sobre a face. Ao término da oclusão o Camshift considera a mão como parte da região em b) e c) e opta erroneamente por eliminar a face e rastrear somente as mãos em d).

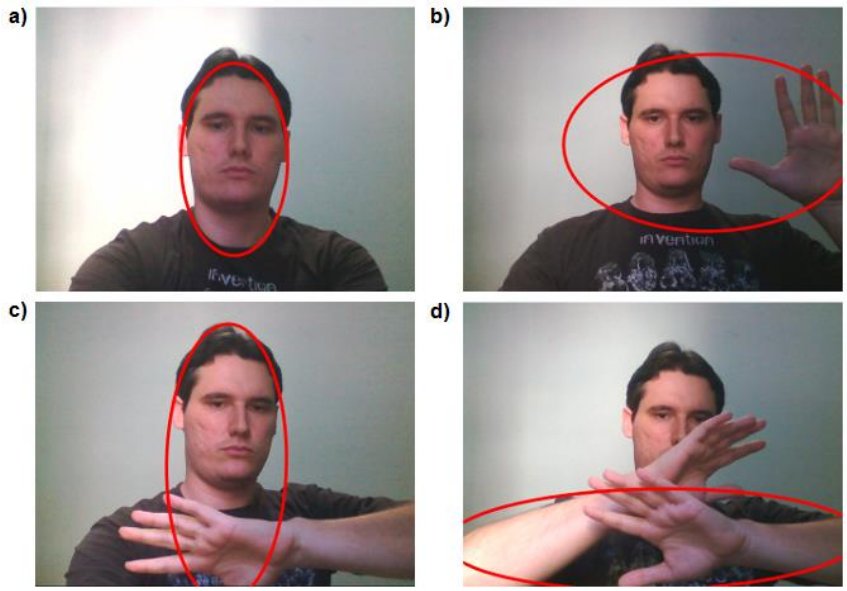


Figura 22 - Problemas no rastreamento da face utilizando Camshift. a) estimação correta; b), c) e d) falhas de estimação após movimentar a(s) mão(s) sobre a face.

Outro problema está na necessidade de uma iluminação controlada no ambiente onde o sistema for utilizado, pois este algoritmo captura a imagem da câmera RGB do Kinect. Esta câmera é de baixa resolução e possui um digitalizador CMOS (Complementary Metal–Oxide–Semiconductor) de baixo custo que gera

ruídos de digitalização, além de todos os seus parâmetros (tempo de exposição, balanceamento de cores, ganho, etc) serem constantemente ajustados, sem a possibilidade de controle por software, para melhorar a qualidade da imagem, ocasionando assim vários problemas já discutidos no Capítulo 2.

Devido à falta de confiabilidade deste algoritmo em caso de oclusões, optamos por não utilizá-lo no sistema final.

3.8.3.2 Centro de Massa do usuário

Através do PrimeSense NITE podemos obter a máscara de pixels (p) correspondente a cada usuário a frente do dispositivo e assim podemos calcular o Centro de Massa (CM) do usuário da seguinte forma:

$$CM_z = \sum_p depth(p)$$

$$CM_x = \sum_p RealWorldCoordinates_x(p)$$

$$CM_y = \sum_p RealWorldCoordinates_y(p)$$

Onde $depth$ retorna a profundidade do pixel p e $RealWorldCoordinates$ (função presente no framework da PrimeSense) retorna a conversão da posição discreta do pixel na imagem para a posição deste ponto no mundo real, definindo assim $<CM_x, CM_y, CM_z>$ como um ponto no espaço 3D correspondente ao centro de massa do usuário.

A fórmula de definição do algoritmo Virtual Wall deve ser alterada para passar a ser calculada dinamicamente, utilizando o centro de massa do usuário, da seguinte forma:

$$MP(d) = \begin{cases} fundo, & d \geq CM_z - \beta \\ objetos, & d < CM_z - \beta \end{cases}$$

Onde CM_z é a coordenada do centro de massa correspondente à profundidade e β é um deslocamento fixo para eliminar parte da face ou do torso que podem vir a transpassar a barreira virtual.

Utilizar o centro de massa do usuário torna a estimação da distância da barreira virtual invariante a problemas de oclusão, mas o centro de massa é suscetível a variação devido a constante movimentação dos braços durante a execução dos gestos da Libras.

Para solucionar este problema recalculamos o centro de massa eliminando desta vez todos os pixels do usuário que ultrapassaram a barreira virtual posicionada com o centro de massa previamente calculado. A Figura 23 mostra os pixels do usuário em amarelo e os pixels a serem eliminados do cálculo em branco. Sendo assim obtemos um centro de massa sem as partes movéis da cena que no contexto dessa aplicação são os braços do usuário.

Figura 23 - Máscara de pixels do usuário (amarelo) sem as partes móveis (branco) que ultrapassaram a barreira virtual.

A Figura 24 mostra uma comparação da coordenada z do centro de massa com e sem partes móveis. Estes dados foram coletados com o usuário aproximando e afastando as duas mãos do Kinect repetidamente. Pode-se notar que o CM_z (vermelho) com as partes móveis evidencia os movimentos do usuário apresentando subsequentes picos de distância. Já o novo algoritmo (azul) que remove as partes móveis foi bem mais tolerante a estes movimentos.

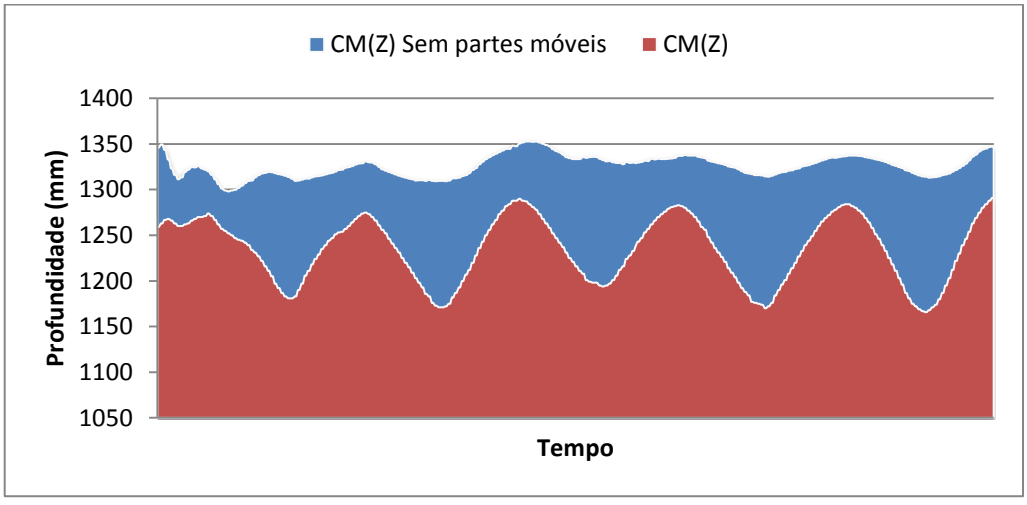


Figura 24 - Gráfico da variação da coordenada Z do Centro de Massa (CM) do usuário com e sem partes móveis.

3.8.4 Segmentação das mãos

Após a eliminação do fundo e das partes do corpo que não interessam para a aplicação através do algoritmo do Virtual Wall, os blobs resultantes que estão a frente da barreira virtual devem ser extraídos e rotulados para análise.

Utilizamos o algoritmo de rotulação de blobs ou componentes conexos proposto pro Chang [CHANG et al., 2004] que utiliza os contornos internos e externos para identificação destes blobs em tempo linear.

Com os blobs rotulados, podemos encontrar três tipos de objetos neste conjunto de blobs:

1. **Ruído:** blobs que podem surgir devido a ruídos gerados pela estimação de profundidade do Kinect;
2. **Pernas do Usuário:** Se o usuário estiver interagindo sentado, e as pernas estiverem no campo de visão do Kinect, elas muito provavelmente irão transpassar a barreira virtual;
3. **Mãos e braços:** as mãos e braços geram os blobs que realmente queremos dar atenção para o reconhecimento dos gestos.

Os blobs de ruído são eliminados por um threshold de área. Todos os blobs inferiores a uma área mínima em pixels são desconsiderados. Neste projeto a área mínima foi definida empiricamente como 400 pixels.

Todos os blobs identificados são ordenados por área em ordem decrescente, e os dois maiores blobs, se presentes, são rotulados como as mãos do usuário.

Porém como citado anteriormente as pernas do usuário podem fazer parte desta lista e muito provavelmente não serão eliminadas pelo threshold de área. Para evitar este problema, definimos uma heurística (Sitting User Heuristics) em que os blobs que tocam a borda inferior da imagem são eliminados. Esta decisão é tomada porque o usuário sentado estará próximo ao dispositivo, para que o sistema tenha resolução suficiente para identificar os gestos, e suas pernas obviamente serão blobs contínuos que sempre irão tocar na borda inferior.

Portanto o resultado final deste algoritmo pode ser: nenhuma mão presente (nenhum blob); uma mão presente, chamada de mão dominante (um blob); ambas as mãos presentes (dois blobs).

Dentro do escopo definido da Libras, quando só uma mão estiver presente, ela sempre será rotulada como a mão dominante, sendo esquerda para canhotos e direita para destros.

Quando duas mãos aparecem simultaneamente o sistema escolhe a mais a esquerda como a mão esquerda e a mais a direita como a mão direita. A partir daí o sistema rastreia os movimentos das mãos associando, por exemplo, sempre o rótulo da mão esquerda para o blob mais próximo do blob anteriormente rotulado como mão esquerda.

Caso uma das mãos desapareça, automaticamente a mão restante é rotulada como a mão dominante.

Um exemplo da rotulação de mãos pode ser visto na Figura 25 onde em vermelho temos a mão esquerda e em azul a mão direita. Pode-se também notar que as pernas do usuário transpassaram a barreira virtual mas foram ignoradas pelo algoritmo pois tocam a borda inferior. O círculo verde determina os centros de massa das mãos segmentadas.

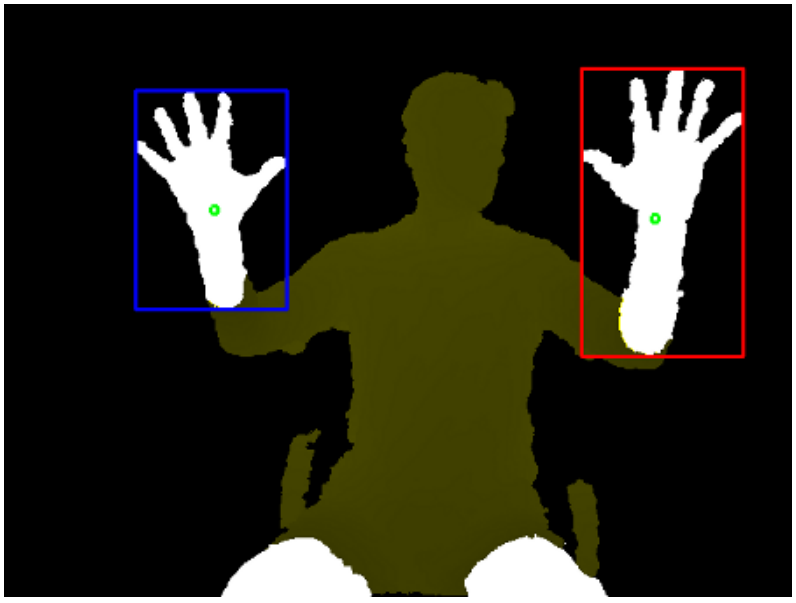


Figura 25 - Resultado da segmentação das mãos do usuário. Em azul a mão direita e em vermelho a mão esquerda.

3.8.5 Extração de Características

Após obter as regiões de interesse das mãos, é necessário descrever as ações do usuário. Para isso utilizaremos um vetor de características que é composto

por propriedades de ambas as mãos juntamente com informações de movimento das mesmas.

Para escolher um conjunto de características nos baseamos no trabalho realizado pelos pesquisadores Thiago Trigo e Sergio [TRIGO et al., 2010] que avaliaram o desempenho de alguns descritores para o reconhecimento de gestos estáticos.

A grande maioria dos trabalhos de reconhecimento de gestos dinâmicos utilizaram dados de luvas com giroscópios e acelerômetros, ou quando não utilizaram destes aparatos extras formataram os gestos com características que foram avaliadas por Trigo [TRIGO et al., 2010].

Dentre todas as características descritas por Trigo, eliminamos as características específicas ao escopo da aplicação de reconhecimento de gestos estáticos e implementamos as restantes que se referem a descritores.

Adicionalmente elencamos duas características adicionais: a posição relacional da mão no espaço, essencial para gestos dinâmicos; e Fourier Descriptors ou Descritores de Fourier.

Portanto, o descritor final de cada uma das mãos em nosso sistema será composto por:

- **Descritores Geométricos:** Aspect Ratio; Circularity; Spreadness; Roudness; Solidity; Finger tips.
- **Descritor de Trajetória:** Posição 3D (X,Y,Z) relativa ao Centro de Massa do usuário.
- **Descritores de Fourier:** Descritor invariável a translação, rotação e escala.

Estes descritores serão explicados nas subseções a seguir.

3.8.5.1 Descritores Geométricos

Os descritores geométricos serão descritos um a um juntamente com a definição de suas fórmulas a seguir.

- **Aspect Ratio:** medida da elongação do blob.

$$AR = EixoMaior/EixoMenor$$

- **Circularity:** medida de similaridade com uma circunferência.

$$C = 4 * \pi * Área/Perímetro^2$$

- **Spreadness:** mede o espalhamento do objeto utilizando alguns momentos invariáveis de HU [HU, 1962].

$$S = \frac{\mu_{20} + \mu_{02}}{\mu_{00} * \mu_{00}}$$

- **Roudness:** é sensível a elongação do blob sendo igual a 1 para um círculo e menos para qualquer outro formato.

$$R = \frac{4 * Área}{\pi * EixoMaior^2}$$

- **Solidity:** calcula a relação entre a área do blob e sua área convexa. Sendo igual a 1 para blobs sem concavidades e menos de 1 para blobs com concavidades. Um exemplo pode ser visto na Figura 26.

$$Sol = Área/ÁreaConvexa$$

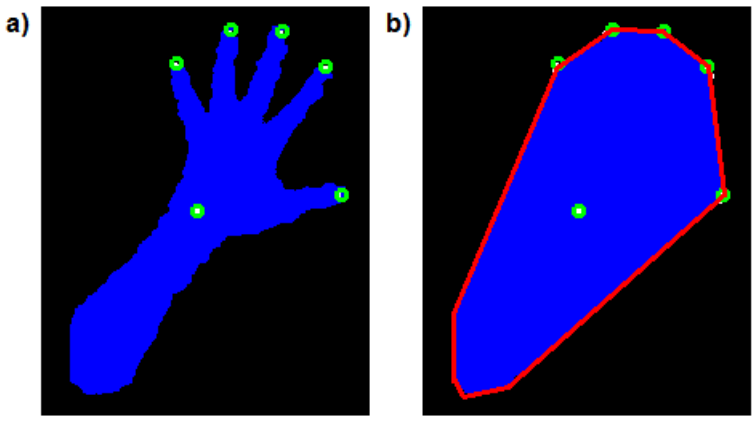


Figura 26 - Pixels em azul correspondem a: a) área do blob; b) área convexa do blob.

- **Finger Tips:** aplicando o algoritmo K-Curvature no contorno dos blobs podemos detectar a ponta dos dedos das mãos. Este algoritmo percorre o contorno do blob e para cada ponto p_i forma dois vetores com os pontos p_{i-k} e p_{i+k} como exemplificado na Figura 27. Se o ângulo θ entre os vetores estiver abaixo de um limiar previamente definido ele é marcado como Finger Tip. Em nossos testes escolhemos empiricamente k = 20 e θ < 50.

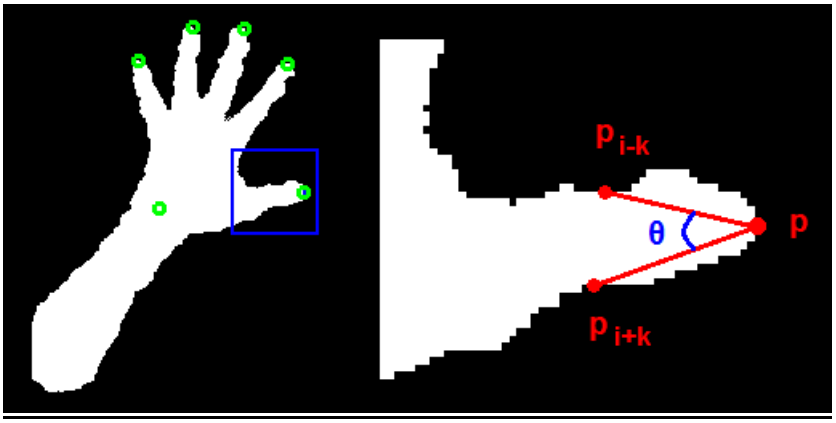


Figura 27 - Detecção de Finger Tips. À esquerda exemplo de detecção com sucesso, à direita ampliação com detalhamento do K-Curvature.

3.8.5.2 Descritor de trajetória

Para descrever a trajetória dos gestos executados pelo usuário optamos pela utilização da posição espacial relativa. Para isso, cada mão terá seu centro de massa calculado da mesma forma que o centro de massa do usuário, obtendo a posição 3D da mão na cena sendo observada.

A posição relativa de cada mão ($PRM_{x,y,z}$) será calculada da seguinte forma:

$$PRM_{x,y,z} = CM_{x,y,z} - CMM_{x,y,z}$$

Onde $CM_{x,y,z}$ é o centro de massa do usuário e $CMM_{x,y,z}$ é o centro de massa da mão.

3.8.5.3 Descritores de Fourier

A transformada de Fourier foi definida pelo matemático francês Joseph Fourier em meados de 1800. Basicamente ela é uma transformação de uma função matemática de tempo em uma função de frequência, ou mais especificamente uma função no conjunto dos números reais ($\mathbb{R}$) em uma função no conjunto dos números complexos ($\mathbb{C}$). A transformação inversa também é possível, permitindo a reconstrução da função real.

Esta transformação pode ser contínua ou discreta. Em aplicações de processamento de imagens, os sinais processados são discretos, portanto vamos abordar somente a Transformada Discreta de Fourier que pode ser calculada com a fórmula:

$$u_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi k \frac{n}{N}} \qquad k = 0, \dots, N-1$$

Onde u_k são os números complexos resultantes da transformada de Fourier e x_n os valores discretos de entrada, que podem ser recuperados através da transformada inversa definida por:

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} u_k e^{i2\pi k\frac{n}{N}} \qquad k = 0, \dots, N-1$$

Para calcular a Transformada Discreta de Fourier (DFT – Discrete Fourier Transform) com menor custo computacional utilizaremos o algoritmo para Transformada Rápida de Fourier (FFT – Fast Fourier Transform) de Cooley-Turkey [COOLEY et al., 1965] que utiliza uma técnica de dividir para conquistar, onde ele expressa a DFT como uma composição de menores DFTs de forma recursiva, reduzindo assim o tempo de execução para O(*N* log *N*).

Retornando ao contexto da nossa aplicação, utilizaremos a Transformada de Fourier para gerar um descritor tolerante a variações de translação, rotação e escala, utilizando informações obtidas da borda do objeto.

Este método para descrição de objetos foi explorado por Zhang [ZHANG et al., 2003] que fez um estudo comparativo de técnicas para gerar descritores de Fourier.

Segundo Zhang um descritor de Fourier é obtido em dois passos:

1. **Obter a assinatura do objeto:** Não é eficiente utilizar todos os pontos do contorno do objeto para definir um descritor, portanto deve-se extrair um conjunto de pontos do contorno que seja suficiente para descrevê-lo, chamado aqui de assinatura do objeto.
2. **Aplicar transformada de Fourier:** Esta aplicação deve garantir que o descritor obtido seja tolerante a rotação, translação e variação de escala.

Para definir a assinatura do objeto Zhang propõe a escolhe de pontos utilizando uma subdivisão do contorno por arcos de igual comprimento. Posteriormente estes pontos devem ser representados de alguma forma e Zhang comparou vários métodos como: coordenadas complexas; distância para o centroide; curvatura; e função angular acumulativa. Segundo os resultados apresentados, o melhor método é representar os pontos como a distância para o centroide do objeto.

Portanto, para gerar a assinatura dos objetos calculamos o centroide da seguinte forma:

$$C_{x,y} = \sum_{i=0}^{N} \overline{x_i}$$

Onde N é o número de pontos do contorno e $\overline{x_i}$ representa cada ponto no contorno com coordenadas (x,y).

Posteriormente subdividimos o contorno em K espaços iguais, e para cada ponto resultante calculamos a diferença de coordenadas do mesmo para o centroide do contorno, obtendo assim a assinatura do objeto.

Esta assinatura de K posições será utilizada como entrada para a Transformada Discreta de Fourier obtendo um vetor de K descritores de Fourier:

$$FD = [FD_0, FD_1, \dots, FD_k]$$

Para tornar este descritor de Fourier invariável a translação basta eliminar o primeiro componente FD_0.

Para ser invariável a escala o descritor de Fourier deve ser normalizando usando o componente eliminado FD_0 da seguinte forma:

$$FD = \left[\frac{FD_1}{FD_0}, \frac{FD_2}{FD_0}, \dots, \frac{FD_k}{FD_0}\right]$$

Para tornar os descritores invariáveis a rotação e ao ponto de início da assinatura do contorno, devemos eliminar a informação de fase do descritor de Fourier da seguinte forma:

$$FD = \left[\left|\frac{FD_1}{FD_0}\right|, \left|\frac{FD_2}{FD_0}\right|, \dots, \left|\frac{FD_k}{FD_0}\right|\right]$$

Apesar de tornar o descritor invariável a rotação a eliminação da fase pode dificultar a diferenciação de objetos, pois isto gera uma perda de informação. A Figura 28 mostra um exemplo de dois objetos que tem descritores similares devido a eliminação de fase.

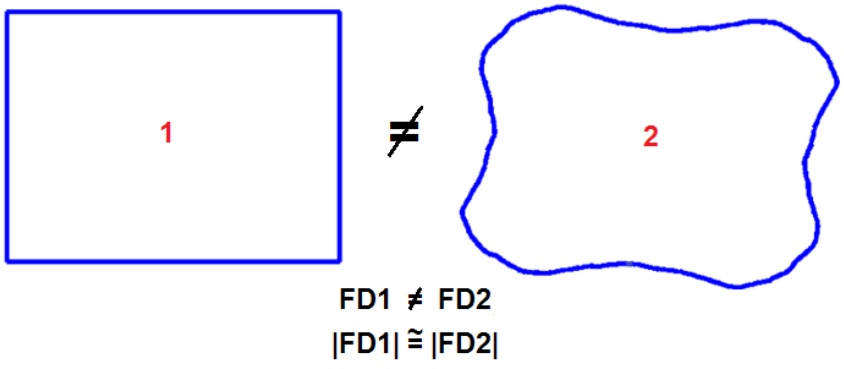


Figura 28 - Exemplo de problema de diferenciação do Descritor de Fourier invariável a rotação |FD|. Estas duas formas são erroneamente classificadas como similares.

Após a obtenção dos descritores de Fourier temos K - 1 números complexos, pois eliminamos o primeiro elemento, que possuem a parte real e imaginária. Podemos representá-los mantendo ambas as partes lado a lado ou utilizando o módulo que pode ser calculado para cada elemento da seguinte forma:

$$|FD_i| = \sqrt{R_i^2 + I_i^2}$$

Onde R_i é a parte real do descritor i e I_i é a parte imaginária.

Para avaliar a performance do descritor de Fourier em nosso contexto de aplicação, escolhemos quatro gestos estáticos da base de dados obtida em [ANJO et al., 2009], sendo estes pertencentes ao alfabeto da Libras e mostrados na Figura 29.

Para classificar os gestos utilizamos um algoritmo de clusterização simples chamado K-Means, descrito na Seção 4.6. E para avaliar o desempenho do classificador, utilizando os descritores de Fourier, definimos 12 experimentos de classificação, descritos na Tabela III.

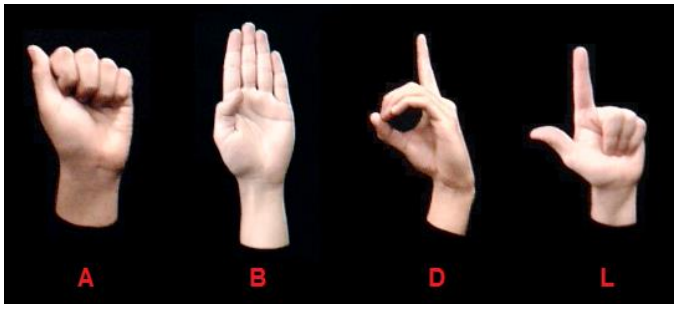


Figura 29 - Gestos estáticos utilizados para avaliar o desempenho dos Descritores de Fourier.

Em todos os 12 experimentos utilizamos 50 amostras para formar um conjunto de treinamento para cada um dos quatro gestos. Este conjunto de treinamento foi utilizado para execução do K-Means e posterior avaliação da separabilidade (eficiência na distinção dos padrões representados por cada cluster) dos clusters resultantes.

Com estes 12 experimentos pôde-se avaliar o desempenho do classificador na presença de amostras rotacionadas; utilizando FD invariante a rotação; representando FD com o módulo dos números complexos; e utilizando 32 ou 64 descritores.

Experimento	Número de Descritores	Amostras Rotacionadas	FD Invariante a Rotação	|FD|
1	32	N	N	N
2	64	N	N	N
3	32	S	N	N
4	64	S	N	N
5	32	N	S	N
6	64	N	S	N
7	32	S	S	N
8	64	S	S	N
9	32	N	S	S
10	64	N	S	S
11	32	S	S	S
12	64	S	S	S

Tabela III - Experimentos para avaliação da classificação de gestos com Descritores de Fourier.

Os resultados obtidos são apresentados no Gráfico da Figura 30, que apresenta as taxas de reconhecimento por gesto para cada experimento, e no Gráfico da Figura 31 que apresenta as médias de reconhecimento para cada experimento.

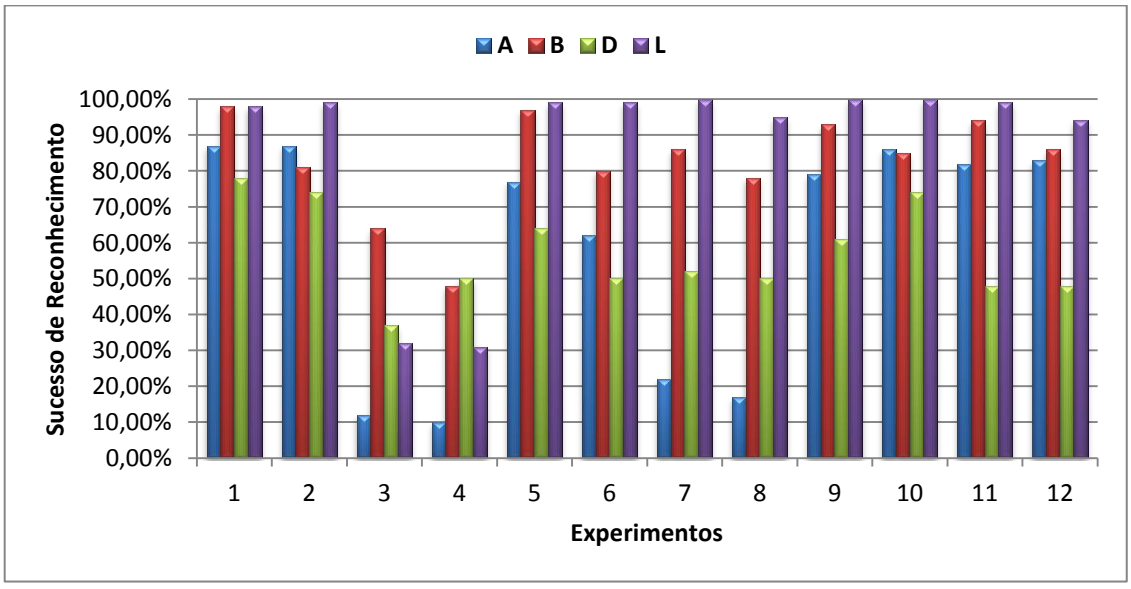


Figura 30 - Gráfico de reconhecimento de cada gesto em cada experimento.

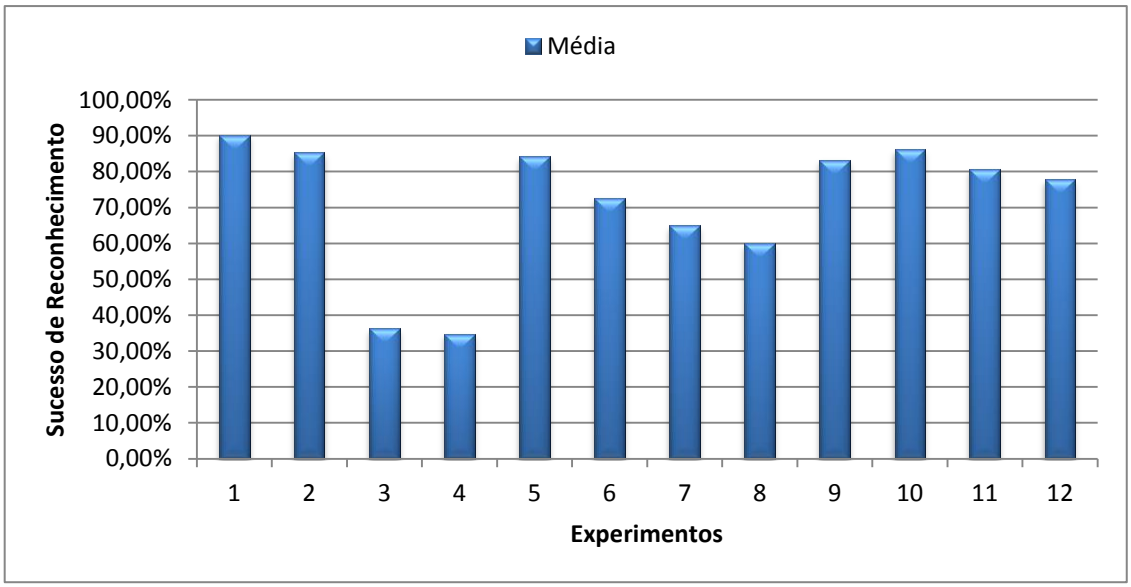


Figura 31 - Gráfico de média de reconhecimento por experimento.

Podemos observar que os melhores resultados foram obtidos utilizado o módulo dos 32 descritores de Fourier. Os resultados também mostram que ao eliminar a informação de fase dos descritores o sistema consegue classificar amostras rotacionadas, mas perde um pouco de precisão na classificação que pode ser notada comparando os experimentos 9 e 11.

Em geral os resultados são satisfatórios e neste projeto utilizaremos 31 descritores de Fourier para cada mão do usuário, aplicando módulo e eliminando a fase.

Capítulo 4

RECONHECIMENTO DE GESTOS

Este capítulo apresenta uma revisão bibliográfica da área de reconhecimento de gestos, com ênfase no reconhecimento de linguagens de sinais. Posteriormente as decisões de projeto são apresentadas.

4.1 Considerações Iniciais

Sistemas de Reconhecimento de Gestos já são amplamente utilizados em aplicações diversas com contextos de utilização específicos. Em aplicações de Interação Humano-Computador (IHC), gestos são comumente utilizados como forma de comunicação ou transferência de informação entre o humano (emissor) e a máquina (receptor e interpretador). Podemos subdividir de forma simplificada os gestos a serem reconhecidos em duas categorias:

- **Gestos Estáticos:** São as posturas estáticas, ou seja, gestos simples sem movimentação ao longo do tempo;
- **Gestos Dinâmicos:** Podem ser definidos como gestos compostos por um conjunto de posturas estáticas ao longo do tempo, que são executadas com determinada trajetória e velocidade, ou também como um gesto que possui uma fase de início ou aceleração chamada *pre-stroke*, uma fase de "desenvolvimento" chamada *stroke* e finalmente uma fase final ou de desaceleração chamada *post-stroke*. Em algumas definições, tanto gestos estáticos quanto gestos dinâmicos podem ser agrupados e compor um único gesto que expresse determinado significado.

Gestos podem ser executados utilizando qualquer parte do corpo ou conjunto delas. Eles podem ser expressos através de movimentos dos braços e mãos com suas posturas, também podem ser executados através de expressões faciais e movimentação da cabeça e dos olhos, ou até serem feitos utilizando todo o movimento do corpo.

As linguagens de sinais englobam a categoria de gestos dinâmicos utilizando as mãos e braços, mas além das posturas com suas trajetórias e velocidades de execução, os gestos nas linguagens de sinais envolvem a posição relativa entre os membros do corpo e algumas vezes até expressões faciais.

Neste capítulo serão apresentados inicialmente alguns projetos em variados cenários de aplicação para reconhecimento de gestos, depois projetos específicos para reconhecimento de língua de sinais serão exemplificados e finalmente uma seção abordará os desafios específicos para o reconhecimento da Língua Brasileira de Sinais (Libras) e nossas decisões de projeto.

4.2 Projetos para Reconhecimento de Gestos Estáticos e Dinâmicos

Após a fase de segmentação e identificação das partes do corpo que serão úteis para o contexto de reconhecimento de gestos a ser aplicado, variados métodos foram desenvolvidos para modelar conceitualmente e reconhecer os gestos. Nesta seção serão apresentados projetos para reconhecimento de gestos estáticos e dinâmicos.

4.2.1 Reconhecimento de Gestos Estáticos

Projetos para reconhecimento de gestos estáticos não precisam levar em consideração a componente temporal, e precisam apenas classificar as posturas em classes bem definidas. As posturas segmentadas são representadas por um conjunto de características que serão utilizadas como entrada para um classificador estatístico. Este classificador pode ser uma rede neural, um algoritmo de

clusterização, um algoritmo de template matching, entre outros. Nesta seção alguns sistemas serão exemplificados brevemente.

Bretzner [BRETZNER et al., 2001] propôs um sistema de reconhecimento de posturas da mão humana, utilizando uma técnica de detecção de partes significativas da imagem sem informações a priori, chamada *Scale-Space Blobs*. Esta é uma solução piramidal que define ao longo das escalas quais blobs são ou não significativos. Através desta técnica, a mão foi modelada como palma e dedos, e a configuração espacial e presença destas partes definiam um modelo de postura. O sistema era capaz de reconhecer quatro gestos para controlar as funcionalidades de uma televisão, e sua desvantagem é o alto tempo de processamento.

Wysoski [WYSOSKI et al., 2002] utilizou uma rede *Multilayer Perceptron* para reconhecer 26 posturas do alfabeto da Língua de Sinais Americana (American Sign Language) e obteve uma taxa de reconhecimento de 96,7%. Para tratamento das imagens utilizou a técnica “boundary chord‘s size histograms” para melhorar a escolha das features.

Phu [PHU et al., 2006] utilizou uma rede *Multilayer Perceptron* com o algoritmo de *Backpropagation* para o reconhecimento de 10 posturas estáticas da Linguagem Americana de Sinais (ASL) e obteve uma taxa de reconhecimento de 92,97% utilizando um limiar de rejeição de 0,7% e um banco de fotos obtidos através de um webcam aplicando um algoritmo de segmentação por cor de pele.

Chen [CHEN et al., 2008] propôs um sistema, em dois níveis, de reconhecimento de gestos em tempo real que utiliza *Haar-like Features* e o algoritmo de aprendizagem *AdaBoost* para detecção das posturas estáticas no nível mais baixo. E estuda a possibilidade de utilizar estas posturas estáticas reconhecidas para representação de gestos dinâmicos através de gramáticas chamadas *Stochastic Context-free Grammar* (SCG). Seus testes englobaram o treinamento de classificadores para quatro posturas estáticas, que foram utilizadas como terminais para construir uma gramática que gerasse um exemplo de uma linguagem livre de contexto. Esta linguagem exemplo continha três gestos dinâmicos que eram compostos por pares de gestos estáticos. Seu sistema foi capaz de reconhecer os gestos em tempo real.

Fang [FANG et al., 2007] propôs a utilização de um classificador para detecção de mãos utilizando *Haar-like Features* e o algoritmo de aprendizagem *AdaBoost*. Uma vez detectada a região que contém a mão, ele utiliza características

de cor e *Scale-space Blobs* em uma estrutura piramidal de 25 níveis, para detectar blobs de interesse e subdividir a mão em palma e dedos. Com a mão devidamente segmentada, um algoritmo de semelhança estatística é aplicado para classificar o gesto estático segundo os modelos de gestos previamente definidos. Seus resultados em cenas com fundo homogêneo obtiveram uma taxa de reconhecimento média de 98% para seis gestos, já em fundos complexos esta taxa caiu para 88%.

Em meu projeto de iniciação científica com apoio da FAPESP [ANJO, 2009], foram implementadas duas redes neurais, a *Multi-Layer Perceptron* (MLP) e a *Neocognitron* para compará-las na tarefa de reconhecimento de gestos estáticos. A rede MLP já estava sendo utilizada em vários projetos de pesquisa para reconhecimento de gestos, porém a Neocognitron não havia sido testada para esta tarefa.

Para comparar as redes, utilizamos os gestos estáticos do alfabeto da Língua Brasileira de Sinais (Libras). Para criamos uma base de gestos, 50 voluntários foram convidados para executar os 23 gestos diante de uma câmera de baixo custo. O Ambiente era interno, com luzes artificiais e um pano preto ao fundo. As pessoas vestiram uma blusa de manga longa preta, e somente a mão permanecia no campo de visão da câmera, facilitando assim a segmentação e análise dos gestos efetuados.

Os resultados apontaram ineficiência da Neocognitron na classificação dos gestos do alfabeto, com uma taxa de reconhecimento média de 64%. Já a MLP se mostrou robusta na classificação dos gestos obtendo uma taxa de reconhecimento geral de 84% sem nenhum tipo de otimização a não ser a centralização dos padrões.

Posteriormente uma nova arquitetura em dois níveis foi proposta para a MLP. Em um primeiro nível são reconhecidos gestos estáticos que são morfologicamente discrepantes dos demais, e os gestos similares são agrupados em classes únicas. O segundo nível é executado em caso de ativação de uma das classes de gestos agrupados do nível anterior. Este nível consiste em uma rede neural para cada um dos agrupamentos, utilizando mais características no treinamento para facilitar a distinção entre os gestos. Com esta arquitetura o sistema obteve uma taxa de reconhecimento de aproximadamente 95%.

O sistema proposto em [ANJO, 2009] foi aprimorado para desenvolvimento de um módulo de reconhecimento de gestos para uma interface web multimodal. Este

sistema foi desenvolvido em conjunto com o Pós-Doutorando Sebastian Feuerstack, que desenvolveu um framework para interfaces web multimodais chamado Mint.

O módulo desenvolvido utiliza luvas coloridas para cada mão, e o sistema detecta estas luvas em tempo real e utiliza a região detectada para extrair características e alimentar a rede neural previamente treinada. Esta rede neural foi treinada para reconhecer cinco gestos, exemplificados na Figura 32, para controlar a navegação em uma interface web.

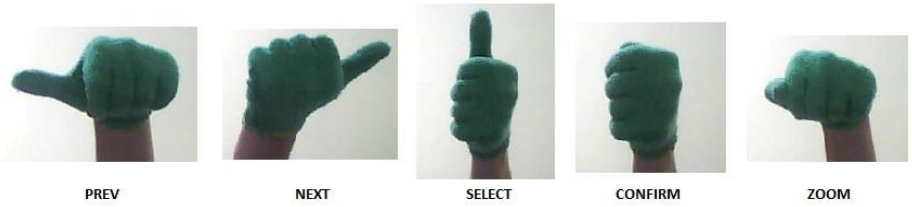


Figura 32 - Exemplos de gestos efetuados com a luva verde.

Os gestos reconhecidos são enviados via TCP/IP para a interface web sendo controlada. A interface do módulo teste de reconhecimento de gestos é exemplificada na Figura 33, mostrando a segmentação das luvas coloridas.

Este sistema protótipo deu origem posteriormente ao GestureUI que será descrito no Capítulo 5 -

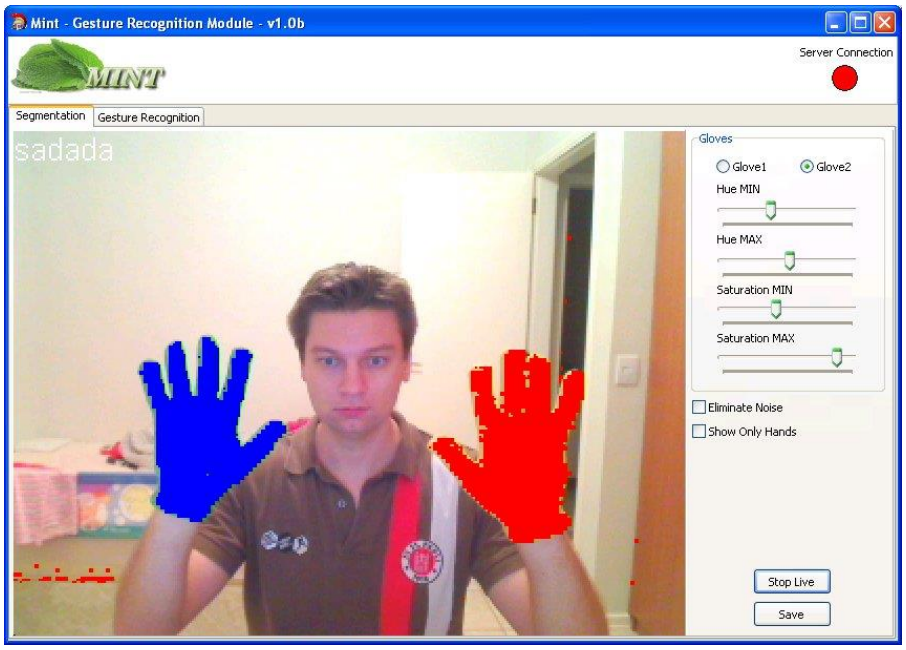


Figura 33 - Exemplo do módulo de segmentação e reconhecimento de gestos utilizando luvas coloridas

4.2.2 Reconhecimento de Gestos Dinâmicos

Para reconhecer gestos dinâmicos a componente temporal deve ser levada em consideração. Gestos dinâmicos podem envolver várias posturas, trajetórias e velocidade ao longo do tempo. Considerando que podem ocorrer erros e imperfeições na segmentação, estes sistemas de reconhecimento devem utilizar modelos estatísticos que absorvam estes erros e levem em consideração o desenvolvimento do gesto ao longo do tempo. Dentre os métodos de modelagem estatística para resolução deste problema, os mais comuns são Hidden Markov Models (HMM), Finite-State Machines (FSM), Time-Delay Neural Networks (TDNN), entre outros.

Hong [HONG et al., 2000] propôs um método para reconhecimento de gestos utilizando Finite-State Machines (FSM). Primeiramente as mãos e cabeça são segmentadas utilizando detectores de cor de pele. Com as partes detectadas, seu algoritmo aprende a separação espacial das características de forma e posicionamento das mesmas e utiliza posteriormente uma FSM para modelar a característica temporal dos gestos. Seu sistema foi testado como controle de ações para o jogo “Simon says”, no qual um personagem pede ao jogador para repetir seus gestos.

Modler [MODLER et al., 2008] propôs um sistema de reconhecimento de gestos cíclicos (Gestos em que o estado ou postura inicial é igual ao estado final) para criação de música utilizando Time-Delay Neural Networks (TDNN). Os gestos são executados com uma só mão em fundo controlado, as características de formato e posição são extraídas e convertidas para o espaço dos números complexos através da transformada de Fourier. A TDNN é treinada para reconhecer os gestos através de sequências de vetores de características, para absorver assim a componente temporal. Em seus testes no sistema de criação musical, ele obteve taxas de acerto de 99%.

Vafadar [VAFADAR et al., 2008] propôs um método para reconhecimento de gestos utilizando uma representação chamada Spatio-Temporal Volumes. Nesta representação, os gestos são segmentados e seu formato e posição definem um volume ao longo do tempo, este volume é utilizado para obter um vetor de características. Para o reconhecimento dos gestos foram testados três diferentes classificadores, K-nearest neighbor, Learning vector quantization e Backpropagation

Neural Networks. Em seus testes para classificação de seis gestos, o classificador de melhor performance foi o K-nearest neighbor com uma taxa de reconhecimento de 99,98% para dados sem ruído e 92,08% para dados ruidosos.

Elmezain [ELMEZAIN et al., 2009] propôs um sistema com câmeras em arranjo estéreo de reconhecimento de gestos utilizando Hidden Markov Models. Os gestos são segmentados com informações de cor e profundidade, e suas trajetórias são posteriormente modeladas utilizando o algoritmo Mean-shift e o filtro de Kalman. O sistema proposto é capaz de reconhecer o alfabeto (A-Z) e números (0-9) desenhados no espaço de visão da câmera utilizando o centro de massa do gesto estático detectado. Os testes apresentaram taxas de reconhecimento de 98,33%.

Utilizando o trabalho realizado em [ANJO, 2009], um novo sistema foi desenvolvido em [PIZZOLATO et al., 2010] para o reconhecimento de gestos dinâmicos. Estes gestos foram definidos como a soletração de palavras utilizando os gestos estáticos ou dinâmicos do alfabeto da Língua Brasileira de Sinais (Libras). As palavras que deveriam ser reconhecidas paralelamente, elegendo a mais provável após a execução do gesto, eram Cisne, Jaguatirica, Leopardo, Pato, Perereca, Peru, Gato, Macaco, Sapo, Tatu, Urso, Arara, Cobra, Foca e Rato.

O sistema funciona em dois níveis, o primeiro sendo o reconhecimento de gestos estáticos proposto em [ANJO, 2009], adicionando apenas os gestos estáticos resultantes da decomposição dos gestos dinâmicos do alfabeto de libras. No segundo nível uma HMM para cada palavra foi treinada, utilizando como entrada a sequência de gestos estáticos reconhecidos no nível anterior. Para o reconhecimento da palavra soletrada, a HMM de maior probabilidade é escolhida após o processamento. Os testes apresentaram taxas de reconhecimento de 91,1%.

4.3 Projetos de reconhecimento de linguagens de sinais

O interesse acadêmico em sistemas de reconhecimento de linguagens de sinais é crescente nos últimos anos devido à evolução de hardware e software para segmentação e mapeamento de trajetórias 3D dos gestos. As linguagens de sinais são diferentes em vários países e cada uma delas apresentam problemas específicos diversos que devem ser tratados de forma diferente. Nesta seção alguns

trabalhos efetuados com variadas linguagens de sinais serão brevemente explicados.

Bauer [BAUER et al., 2000] propôs um sistema para reconhecimento de gestos da Língua de Sinais alemã (German Sign Language – GSL). Ele utilizou Hidden Markov Models (HMM) para modelar os gestos ao longo do tempo. Seu sistema é capaz de reconhecer 97 gestos diferentes com uma taxa de reconhecimento de 91,7%.

Wang [WANG et al., 2002] focou sua pesquisa em como expandir um sistema de reconhecimento de gestos sem perder a precisão nas classificações. Utilizando a Língua de Sinais Chinesa (Chinese Sign Language – CSL), ele subdividiu os gestos em menores unidades chamadas fonemas. Cada fonema foi treinado utilizando uma HMM, e a correlação entre estes fonemas para representar a língua foi estruturada em árvores gramaticais. Em sua pesquisa foram definidos 2400 fonemas para gerar 5119 gestos. Em seus testes que envolviam reconhecimento de gestos, por composição de fonemas, e reconhecimento de sentenças, por composição de gestos, a taxa de reconhecimento média foi de 90%.

Brashear [BRASHEAR et al., 2006] desenvolveu um sistema de reconhecimento de gestos para ser utilizado em um jogo educativo chamado “CopyCat”, em que as crianças surdas aprendem a língua de sinais americana (American Sign Language – ASL). Seu sistema utiliza luvas coloridas com acelerômetros para facilitar a segmentação, e as características de movimentação e postura das mãos das crianças são utilizadas para treinar uma HMM para cada gesto. Seu conjunto de dados de treinamento consiste em 541 frases compostas por 1959 gestos executados pelas crianças. O sistema obteve taxas de reconhecimento de 91.75%.

Pahlevanzadeh [PAHLEVANZADEH et al., 2007] propôs um método para reconhecimento da Língua de Sinais Tailandesa, mapeando a trajetória do gesto utilizando descritores de Fourier e modelando a postura da mão utilizando um descritor chamado *Generic Cosine Descriptor* (GCD). Para prova de conceito, seu sistema foi testado utilizando 15 gestos dinâmicos e obteve uma taxa de reconhecimento de 100%.

4.4 Língua Brasileira de Sinais e seus desafios para segmentação e reconhecimento

Como já mencionado nas seções anteriores, já desenvolvemos variados trabalhos envolvendo a Língua Brasileira de Sinais (Libras), como em [PIZZOLATO et al., 2010] e [ANJO, 2009]. Porém estes trabalhos englobaram somente gestos referentes ao alfabeto, e provaram conceitos de modelagem de gestos dinâmicos através da soletração de palavras por gestos.

Para evoluirmos em direção a um sistema mais robusto, capaz de segmentar e rastrear as partes do corpo ao longo do tempo e interpretar essa informação de acordo com a língua definida pela LIBRAS, é preciso um estudo mais detalhado sobre as particularidades desta língua, e os possíveis problemas que deverão ser tratados.

Esta seção tem por objetivo contextualizar a Língua Brasileira de Sinais e explicar um estudo inicial da língua tentando identificar constantes favoráveis a classificação dos gestos e possíveis problemas que deverão ser tratados por um sistema de reconhecimento de Libras.

4.4.1 Língua Brasileira de Sinais (Libras)

A Língua Brasileira de Sinais foi oficializada por meio da Lei nº 10.436, de 24 de abril de 2002, que tornou a Libras a segunda língua oficial do país. Além disso, a regulamentação do Decreto nº 5.626, no final de 2005, determinou a inclusão da Libras como disciplina curricular obrigatória nos cursos de formação de professores, de nível médio e superior em Instituições públicas e privadas, e também estabeleceu um prazo de 10 anos para que a disciplina de Libras seja oferecida em todos os cursos de graduação do País.

Este Decreto ainda determina que as empresas públicas devem garantir às pessoas surdas atendimento diferenciado, por meio do uso da tradução e interpretação da Libras, que deverá ser utilizada por, pelo menos, 5% do seu quadro de profissionais.

Segundo [ROSSI et al., 2009] a Língua Brasileira de Sinais não é simplesmente mímica, alfabeto manual ou gestos isolados utilizados pelos surdos

para facilitar a comunicação, mas sim, uma língua com uma estrutura gramatical própria, que dentro de um contexto, pode traduzir qualquer situação.

4.4.2 Parâmetros para formação de um sinal em Libras

Nesta subseção alguns conceitos básicos sobre parâmetros para formação de sinais em LIBRAS serão apresentados segundo o material criado pelo professor Rossi na Universidade Federal de Uberlândia (UFU) [ROSSI et al., 2009].

Rossi [ROSSI et al., 2009] afirma:

> Todas as línguas têm seus próprios parâmetros para a formação de palavras ou itens lexicais. A partir de regras cada língua combina elementos que formam palavras e as palavras formam frases em um contexto. Na Libras, assim como em todas as línguas de sinais, um item lexical é chamado de sinal.

Ainda segundo Rossi, este sinal pode ser formado a partir da alteração ou combinação de cinco parâmetros, descritos a seguir:

- **Configuração de mãos:** É a postura das mãos para efetuar o sinal. Pode ser uma letra do alfabeto manual ou outras posturas feitas pela mão dominante (mão direita para os destros, mão esquerda para os canhotos), ou pelas duas mãos do sinalizador. Há sinais que podem ser executados com as duas mãos utilizando a mesma postura e existem também gestos que podem ser executados com posturas distintas em cada uma das mãos. Exemplo, na Figura 34: *Trabalhar* – as duas mãos fazendo a postura da letra L do alfabeto Libras; *Xícara* – cada mão com uma postura diferente da outra.

Figura 34 - Exemplo de Configuração de Mãos para *Trabalhar* e *Xícara.*

- **Ponto de articulação ou localização da mão:** É o local onde a mão dominante realiza o sinal, podendo tocar alguma parte do corpo ou estar perto da mesma. O gesto pode ser simplesmente executado a frente do corpo do emissor, no chamado "*espaço de enunciação"* (do tórax até a cabeça). Exemplo, na Figura 35: O sinal *Brincar (a)* é feito no espaço de enunciação e o sinal *Pensar (b)* é feito tocando a testa.

Figura 35 - Exemplo de Localização da mão em *Brincar* e *Pensar.*

- **Orientação:** É a direção para a qual a palma da mão aponta na produção do sinal. Segundo Rossi [ROSSI et al., 2009], estudiosos da Libras enumeram seis tipos de orientação da palma da mão: para cima, para baixo, para o corpo, para frente, para a direita ou para a esquerda. Exemplo, na Figura 36: a) *Ajudar* - para frente; b) *Levantar* - para cima.

Figura 36 - Exemplo de Orientação em *Ajudar* e *Levantar.*

- **Movimento:** Os sinais podem ter movimento, aqui chamados de gestos dinâmicos, ou serem apenas posturas estáticas, aqui chamadas de gestos

estáticos. Os movimentos podem ser efetuados em várias direções, podendo alterar a configuração das mãos, pulsos e antebraço. Segundo Rossi, os movimentos podem ser classificados por tipo, direcionalidade, maneira e frequência. Quanto à direcionalidade os movimentos podem ser unidirecionais, bidirecionais ou multidirecionais. A maneira é a categoria que descreve a qualidade, a tensão e a velocidade do movimento. A frequência refere-se ao número de repetições de um movimento.

- **Expressão facial e/ou corporal ou expressões não manuais:** Muitos sinais tem em sua execução expressões faciais e/ou corporais, como o sinal de *Alegre (a)*, na Figura 37. Estas expressões são caracterizadas por movimentos da cabeça, dos olhos, da boca, da sobrancelha, ou do tronco. Segundo Rossi, as expressões não manuais marcam os tipos de frases: interrogativa, exclamativa, negativa, afirmativa, etc. Há sinais feitos somente com a bochecha, como *Ladrão*, sinais feitos com a mão e a expressão facial, como o sinal *Bala*, e há ainda sinais em que sons e expressões faciais complementam os traços manuais, como o sinal de *Moto (b)* na Figura 37.

Figura 37 - Exemplos de Expressões em *Alegre* e *Moto*.

A combinação destes parâmetros formam os sinais da Libras. Segundo Rossi, *"falar com as mãos é, portanto, combinar estes elementos que formam as palavras e estas formam as frases em um contexto"*. Lembrando que os sinais não precisam ter todos os parâmetros supracitados, todas as combinações dos mesmos são possíveis para a construção de palavras da Libras.

4.4.3 Estrutura Linguística da Libras

Nesta subseção será apresentada uma breve explicação sobre a estrutura linguística da Libras. Segundo Rossi [ROSSI et al., 2009]:

> As frases em Libras são estruturadas de forma clara e objetiva. Nas línguas visuais-espaciais os verbos no presente, passado ou futuro não sofrem modificação na sua forma, o tempo verbal é indicado por advérbios de tempo colocados na estrutura frasal ou deve ficar claro no contexto do enunciado.
>
> Para entender a estrutura linguística da Libras é preciso ter em mente que um sinal não equivale a uma palavra da língua oral, e sim a uma ideia. Essa ideia pode estar relacionada a uma frase, uma expressão ou até mesmo a uma frase da Língua Portuguesa. Por isso, na Libras, os sinais jamais poderão ser usados na mesma estrutura da Língua Portuguesa pois, consequentemente, a mensagem seria deturpada ou perderia o sentido.
>
> Em Libras não há sinais equivalentes a artigos e preposições, uma vez que estes são usados para designar outras classes gramaticais. Ex: Os artigos especificam ou generalizam os substantivos classificando-os em definidos ou indefinidos. Na construção frasal da Libras a ideia deles está implícita no contexto da fala.

Uma das características da Libras é diferença na estrutura para a construção de frases em relação a Língua Portuguesa que tem a seguinte estrutura:

SUJEITO (S) - VERBO (V) - OBJETO (O)

Maria - gosta - de gatos.

Segundo Rossi, a Língua Portuguesa é uma língua de base sujeito-predicado enquanto que a Libras é uma língua predominantemente do tipo Tópico-Comentário. A ordem Sujeito-Verbo-Objeto também é utilizada em Libras.

A topicalização é exemplificada na Tabela IV.

Tópico	Comentário
PESQUISAR	ELA NÃO GOSTAR (=pesquisar, ela não gosta)
RUA ACIDENTE	NÃO-ENXERGAR (=o acidente na rua eu não vi)
CAFÉ AÇÚCAR	NÃO (=açúcar no café (ela) não pôs)

Tabela IV - Exemplos da estrutura Tópico-Comentário da Libras.

Pelos exemplos acima podemos perceber que a construção da Língua de Sinais é feita apresentando inicialmente o sinal (ou conjunto de sinais) que caracterizem o tópico da frase, e posteriormente os gestos que caracterizam a ação são executados.

Ainda segundo Rossi: *"Vale ressaltar que, ao adquirir um vocabulário considerável em Libras, não devemos atribuir sinais correspondentes aos vocábulos presentes na estrutura frasal da língua portuguesa, pois isto seria português sinalizado".*

4.4.4 Desafios para a segmentação e reconhecimento da Libras

Como visto anteriormente, a Libras é uma língua com sua própria gramática e estrutura lexical que está fortemente ligada ao contexto em que está sendo utilizada para se extrair um significado. Os gestos podem ser efetuados com ambas as mãos na zona de enunciação (a frente do corpo) com movimento ou não, e tocando partes do corpo ou não. Sinais também podem ser efetuados com ajuda de expressões faciais ou movimentações do corpo.

Diante desde cenário podemos exemplificar vários problemas relacionados a segmentação das partes do corpo:

- **Alto grau de liberdade de movimentação:** As mãos e braços possuem alto grau de mobilidade, e isto potencializa o espaço de busca por padrões de movimento.
- **Sobreposição de partes:** Devido ao alto grau de liberdade, os braços podem se sobrepor entre si, e também se sobrepor ao tórax ou cabeça. Isto dificulta a segmentação, pois seriam dois blobs em movimento que se tornariam um único objeto.
- **Oclusão de partes:** Similar ao item anterior ocorre quando a parte não está mais no campo de visão da câmera, aumentando a complexidade do sistema devido a necessidade da estimação (através de modelos previamente estabelecidos) da provável posição da parte oculta.
- **Expressões faciais:** A detecção de faces já é uma atividade relativamente trivial, diferentemente de expressões faciais. Elas são importantes na língua de sinais, e seu reconhecimento requer um sistema de alta complexidade mesmo com a subdivisão da face em partes (olhos,

sobrancelha, nariz, boca) a sua variabilidade de pessoa para pessoa dificulta a precisão na classificação.

- **Trajetórias em 3D:** A imagem capturada de uma câmera é uma projeção 2D do ambiente 3D sendo observado. Esta redução da dimensionalidade dificulta, às vezes até impossibilita, o cálculo com precisão da variação de profundidade dos objetos em movimento que foram segmentados. Os gestos da Libras envolvem movimentação em trajetórias 3D, e esta informação as vezes é crucial para a classificação do gesto executado.

Na tarefa de reconhecimento alguns problemas também podem ser identificados, como:

- **Vasto vocabulário de posturas:** A Libras define para cada gesto uma postura base para a mão dominante, ou ambas as mãos. Estas posturas são as unidades básicas da composição dos sinais e podem ser utilizadas para auxiliar na classificação dos gestos. A Libras possui 73 posturas diferentes, que são exemplificadas na Figura 38.

Figura 38 - Exemplos de posturas estáticas da Libras

- **Variação de perspectiva das posturas:** A variação de perspectiva se dá devido ao fato de a câmera permanecer em ponto fixo e só capturar um ângulo de visão da cena 3D que é projetada em uma imagem 2D. As posturas podem variar em tamanho, e ângulo de visão devido ao alto grau

de liberdade das mãos, dificultado assim o trabalho de um algoritmo de detecção de padrões.

- **Variação da trajetória do sinal:** A trajetória pode ser considerada como característica candidata a classificação dos gestos, porém, mesmo com a padronização, cada pessoa pode executar o gesto de uma maneira ou cometer pequenos erros durante a execução, mas que não comprometem o entendimento do gesto por alguém fluente em Libras. O sistema de reconhecimento deve tentar absorver esta variação.
- **Variação de velocidade de execução:** Quando a componente temporal é levada em consideração, a velocidade também é um parâmetro de identificação dos gestos. Ela pode indicar um contexto ou enfatizar um sinal, mas varia em sua execução de pessoa para pessoa.

4.5 Decisões de Projeto e Resultados

A última etapa do sistema para reconhecimento de gestos da Libras é a classificação do gesto. Para isto optamos pela utilização do Hidden Markov Model (HMM) que é um modelo de classificação estatístico que leva em consideração o fator tempo, que é essencial para classificação de gestos dinâmicos. Baseamos nossa escolha nos resultados promissores obtidos pelos pesquisadores que utilizaram a HMM para o mesmo propósito, como descrito nas Seções 4.2 e 4.3.

Nas próximas seções apresentaremos os detalhes de implementação das ferramentas de classificação que fazem parte do sistema final do projeto, iniciando pela ferramenta de clusterização K-Means e concluindo com o classificador HMM.

Os resultados de classificação de gestos serão apresentados no Capítulo 6 -

4.6 K-Means

K-Means é um algoritmo que tem por objetivo agrupar N amostras em K clusters onde cada amostra pertence ao cluster com centroide mais próximo. Este

problema é considerado computacionalmente difícil e classificado como NP-Difícil, porém existem algoritmos que são aproximações de solução para o problema e que conseguem convergir para uma solução aceitável.

Neste projeto utilizamos o K-Means como ferramenta para clusterização dos vetores de características dos gestos para que os clusters sejam utilizados como símbolos de entrada para o Hidden Markov Model (HMM), que será explicado na próxima seção.

Os algoritmos de Lloyd, Binary Split e normalização por Z-Scores foram implementados para utilização neste projeto e serão descritos nas subseções a seguir.

4.6.1 Algoritmo de Lloyd

O algoritmo de Lloyd é o mais implementado pela comunidade científica e é basicamente um algoritmo iterativo de otimização.

A primeira etapa do algoritmo é a inicialização, que consiste em escolher os K centroides iniciais para os K clusters. Segundo Peña [PEÑA et al., 1999] esta é considerada a fase mais crítica do algoritmo pois pode determinar tanto a velocidade de convergência do K-Means quanto a qualidade da solução encontrada. Existem vários métodos de inicialização, mas os mais comumente utilizados são chamados Forgy e Random Partition.

- **Forgy:** Este algoritmo escolhe K amostras do conjunto de treinamento como centroides iniciais.
- **Random Partition:** Este algoritmo divide as amostras randomicamente em K agrupamentos e calcula os centroides iniciais.

Ambos os métodos são randômicos e fazem com que cada execução do K-Means possa gerar um resultado diferente, exigindo assim que o K-Means seja executado múltiplas vezes até que se encontre a melhor solução. Segundo Peña [PEÑA et al., 1999] que avaliou vários métodos de inicialização, o que obteve melhores resultados foi o Random Partition.

Após a fase de inicialização o algoritmo de Lloyd inicia seu processo iterativo para encontrar os clusters otimizados da seguinte forma:

1. Todas as amostras são associadas ao cluster mais próximo;

2. Os novos centroides são calculados como sendo a média das amostras associadas a ele;
3. Passos 1 e 2 são repetidos até a convergência do algoritmo, que acontece quando todas as amostras são sempre associadas ao mesmo cluster do passo anterior.

Para o cálculo da distância entre vetores, utilizado pelo K-Means para associar amostras aos clusters, utilizamos a Distância Euclidiana definida por:

$$DE = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

Onde $< p_1, p_2, \ldots, p_n >$ e $< q_1, q_2, \ldots, q_n >$ definem os dois vetores que queremos mensurar a distância euclidiana.

4.6.2 Algoritmo Binary Split

Como dito anteriormente o algoritmo do K-Means depende de sua inicialização e como os métodos previamente apresentados são não determinísticos, ou seja, a cada execução geram um resultado diferente, temos de executar o K-Means várias vezes e corremos o risco de sempre convergir para mínimos locais.

Chiangkai [CHIANGKAI, 1997] propôs em sua tese de mestrado sobre reconhecimento de fala, um algoritmo para clusterização chamado Binary Split. Basicamente ele é uma otimização que utiliza o Lloyd K-Means, mas torna o algoritmo determinístico e tenta garantir que a subdivisão do espaço seja feita de forma uniforme, facilitando o processo de convergência e reduzindo as chances de convergir em um mínimo local.

O algoritmo Binary Split consiste em:

1. **Cluster Inicial:** Todas as amostras são consideradas como um único grande cluster, calculando seu centróide como a média de todas as amostras de treinamento.
2. **Subdividir Centroides:** Cada centroide dos clusters já definidos deve ser dividido em dois novos centroides da seguinte forma:

$$\boldsymbol{C_{i,1} = C_i - \gamma}$$

$$\boldsymbol{C_{i,2} = C_i + \gamma}$$

Onde $\boldsymbol{C_{i,1}}$ e $\boldsymbol{C_{i,2}}$ são os dois novos clusters e $\boldsymbol{\gamma}$ é o parâmetro de divisão geralmente definido no intervalo $0{,}01 \leq \boldsymbol{\gamma} \leq 0{,}05$.

3. **Lloyd K-Means:** Aplicar Lloyd K-Means para obter duas vezes mais clusters utilizando como clusters iniciais os clusters subdivididos.
4. **Iteração:** Repetir passos 2 e 3 até que o número desejado de clusters seja atingido.

Como este algoritmo subdivide cada cluster em duas partes iterativamente, ele restringe o número de clusters a potências de 2 (2, 4, 8, 16, 32, etc).

4.6.3 Normalização de Dados: Z-Scores

Os vetores de características que serão utilizados para reconhecimento dos gestos da Libras são compostos por valores com escalas de variação diferentes. Como o K-Means utiliza a distância euclidiana para tomar decisões de associação de amostras a clusters, esta variação de escala ocasionará problemas na clusterização.

Para solucionar este problema as amostras precisam ser normalizadas antes da execução do Binary Split ou K-Means, tornando a comparação dos vetores de características factível.

Um método de normalização amplamente utilizado é chamado de Z-Scores, ou Standard Scores ou até Distância de Mahalanobis [DUDA et al., 2001] que consiste em obter um valor normalizado que representa a distância de uma amostra para a média total em número de desvios padrões.

Portanto, para todas as amostras pertencentes ao conjunto de treinamento cada uma das características deve ter sua média e desvio padrão calculados para posteriormente obter-se o Z-Score da seguinte forma:

$$\mu_c = \sum_{a=0}^{N} x_{c,a}$$

$$\sigma_c = \sqrt{\frac{1}{N}\sum_{a=0}^{N}(x_{c,a} - \mu_c)^2}$$

$$Z(x_{c,a}) = \frac{x_{c,a} - \mu_c}{\sigma_c} \quad para \quad s = 0 \ldots N$$

Onde μ_c e σ_c são média e desvio padrão, respectivamente, da característica c; $x_{c,a}$ é o valor da característica c da amostra a; N é o número de amostras no set de treinamento.

4.7 Hidden Markov Models (HMM)

Segundo Rabiner [RABINER, 1989], os Hidden Markov Models (HMM) ou Modelos Ocultos de Markov são modelos estatísticos, para sinais discretos ou contínuos, que levam em consideração o fator tempo para descrever os padrões da fonte geradora do sinal sendo modelado.

O modelo da HMM, denotado como $\lambda = (A, B, \pi)$ no caso discreto, é definido pelos seguintes elementos:

- **Número de estados do modelo (N):** Cada modelo é composto por um número finito de estados que estão interligados através de transições.
- **Número de símbolos que podem ser observados (M):** O modelo tem um conjunto finito de símbolos, chamado de alfabeto, que consiste nas unidades básicas do gerador ou interpretador de sequências.
- **Distribuição de Probabilidade de Transição de Estados (A):** Cada modelo possui uma matriz de probabilidades de transição de um estado S_i para um estado j ($a_{i,j}$), definido da seguinte forma:

$$a_{i,j} = P\left(q_{t+1} = S_j \mid q_t = S_i\right), \qquad 1 \leq i, j \leq N$$

$$a_{i,j} \geq 0 \quad e \quad \sum_{j=1}^{N} a_{i,j} = 1$$

  Onde q_t é o estado atual no tempo t denotado por S_i.

- **Distribuição de Probabilidade de Observação de um Símbolo (B):** Valores que definem a probabilidade de emissão ($b_j(k)$) ou observação de um símbolo (v_k) da seguinte forma:

$$b_j(k) = P(v_k \; em \; t \mid q_t = S_i), \qquad 1 \leq j \leq N, \; 1 \leq k \leq M$$

  Onde $P(v_k \; em \; t \mid q_t = S_i)$ é a probabilidade de emissão de v_k no tempo t no estado S_i.

- **Probabilidade do Estado Inicial (π):** Valores que determinam a probabilidade (π_i) de um estado S_i ser um estado inicial (q_1), definidos da seguinte forma:

$$\pi_i = P(q_1 = S_i), \qquad 1 \leq i \leq N$$

Segundo Rabiner [RABINER, 1989] existem três problemas básicos que precisam ser solucionados pela HMM para permitir a utilização da mesma em aplicações reais:

1. **Problema de Avaliação:** Dada uma sequência de observações $O = O_1, O_2, \dots, O_T$ e um modelo HMM $\lambda = (A, B, \pi)$, como calculamos a probabilidade $P(O|\lambda)$ de obtermos O dado o modelo λ?
2. **Problema de Estimação ou Decodificação:** Dada uma sequência de observações e o modelo HMM, como estimamos a melhor sequência de estados que possa ter decodificado esta sequência?
3. **Problema de Treinamento:** Como ajustar os parâmetros do modelo λ para maximizar a probabilidade $P(O|\lambda)$ dado um conjunto de treinamento?

Cada um destes problemas será descritos e suas soluções apresentadas nas próximas subseções. Após a descrição destes problemas uma apresentação das possíveis arquiteturas da HMM será feita para justificar a escolha da arquitetura Esquerda-Direita.

4.7.1 Problema de Avaliação

Neste problema, dado um modelo HMM e uma sequencia de observação deve-se calcular a probabilidade deste modelo ter gerado a sequência, ou ainda, quantificar quanto o modelo se assemelha a sequencia de observações.

A solução para este problema ajuda quando temos de avaliar em um conjunto de modelos qual tem maior probabilidade de ter gerado uma dada sequencia, sendo essencial para nossa aplicação de reconhecimento de gestos.

Segundo Rabiner [RABINER, 1989], solucionamos este problema utilizando o algoritmo Forward-Backward.

Inicialmente o algoritmo define a variável Forward $\alpha_t(i)$ que pode ser definida como $\alpha_t(i) = P(O_1 O_2 \dots O_T, q_t = S_i | \lambda)$, e será calculada em três etapas da seguinte forma:

1. **Inicialização:**

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N$$

2. **Indução:**

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \leq t \leq T-1, \qquad 1 \leq j \leq N$$

3. **Término:**

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Como obtemos $P(O|\lambda)$ ao fim do processo, somente o algoritmo de Forward já é suficiente para solucionar este primeiro problema. Porém apresentaremos também o Backward que será necessário na solução dos problemas de Estimação e Treinamento.

De uma maneira similar ao Forward, o Backward define a variável $\beta_t(i)$, definina como $\beta_t(i) = P(O_{t+1}O_{t+2} \ldots O_T | q_t = S_i, \ \lambda)$ e calculada em duas etapas:

1. **Inicialização:**

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N$$

2. **Indução:**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

4.7.2 Problema de Estimação ou Decodificação

Para determinar a melhor sequencia de estados, $Q = \{q_1 q_2 \ldots q_T\}$, dada uma sequencia de observações $O = \{O_1 O_2 \ldots O_T\}$, utilizaremos o algoritmo de Viterbi que consiste na definição:

$$\delta_t(i) = \max_{q_1 q_2 \ldots q_{t-1}} P(q_1 q_2 \ldots q_t = i, O_1 O_2 \ldots O_t | \lambda)$$

Onde $\delta_t(i)$ é a maior probabilidade em um caminho de estados no tempo t que leva em consideração as primeiras t observações e termina no estado S_i. Por indução temos:

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] b_j(O_{t+1})$$

Para obter a sequência de estados precisamos calcular este parâmetro maximizado para todo t e j. Isto é feito pelo algoritmo de Viterbi dividido em quatro passos como a seguir:

1. **Inicialização:**

$$\delta_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N$$

$$\psi_1(i) = 0$$

2. **Recursão:**

$$\delta_t(j) = \max_{1 \leq i \leq N} \left[\delta_{t-1}(i) a_{ij}\right] b_j(O_t), \qquad 2 \leq t \leq T, \qquad 1 \leq j \leq N$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \leq i \leq N} \left[\delta_{t-1}(i) a_{ij}\right], \qquad 2 \leq t \leq T, \qquad 1 \leq j \leq N$$

3. **Finalização:**

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

$$q_T^* = \operatorname*{argmax}_{1 \leq i \leq N} [\delta_T(i)]$$

4. **Retropropagação do caminho de estados:**

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

4.7.3 Problema de Treinamento

O problema de treinamento é considerado o mais difícil e consiste na estimação dos parâmetros que definem o modelo HMM. Segundo Rabiner [RABINER, 1989] podemos definir $\lambda = (A, B, \pi)$ visando maximizar localmente a probabilidade $P(O|\lambda)$ de forma iterativa com o método Baum-Welch.

No método de treinamento de Baum-Welch devemos inicialmente definir $\xi_t(i,j)$ que é a probabilidade de estar no estado S_i no tempo t e no estado S_j no tempo $t+1$. Utilizando o algoritmo Foward-Backward explicado anteriormente podemos definir $\xi_t(i,j)$ da seguinte maneira:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)}$$

$$P(O|\lambda) = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

Posteriormente devemos definir $\gamma_t(i)$ que é a probabilidade de estar no estado S_i no tempo t dada uma sequência de observação e um modelo. Podemos definir $\gamma_t(i)$ da seguinte forma:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

Utilizando estas fórmulas definidas acima podemos finalmente definir o método de Baum-Welch para estimação dos parâmetros do modelo HMM da seguinte maneira:

- **Estimação da probabilidade do Estado Inicial:**

$$\overline{\pi}_i = \gamma_1(i), \qquad \sum_{i=1}^{N} \overline{\pi}_i = 1$$

- **Estimação das probabilidades de transição:**

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \qquad \sum_{i=1}^{N} \overline{a}_{ij} = 1, \qquad 1 \le i \le N$$

- **Estimação das probabilidades de emissão de símbolos do alfabeto:**

$$\overline{b}_j(k) = \frac{\sum_{\substack{t=1 \\ O_t = v_k}}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}, \qquad \sum_{i=1}^{M} \overline{b}_j(k) = 1, \qquad 1 \le j \le N$$

Estes procedimentos de ajuste dos parâmetros que definem o modelo são executados a cada iteração do algoritmo de treinamento, que consiste em apresentar todos os padrões do conjunto de treinamento, até quer a regra de convergência final seja satisfeita.

Neste projeto consideramos que o modelo convergiu quando a probabilidade $P(O|\lambda)$, que está sendo maximizada, não sofrer alteração significativa da iteração anterior para a iteração atual. Sendo assim, nossa regra de convergência consiste em:

$$P_t(O|\lambda) - P_{t-1}(O|\lambda) \le \tau$$

Onde $P_t(O|\lambda)$ corresponde à probabilidade no tempo t (iteração atual), $P_{t-1}(O|\lambda)$ corresponde à probabilidade no tempo $t-1$ (iteração anterior) e τ é o limiar de convergência que neste projeto definimos com o valor $\tau = 0{,}01$.

4.7.4 Arquiteturas da HMM

Após a definição do modelo estatístico proposto na HMM, precisamos definir sua arquitetura que será fundamental para o comportamento do modelo no processo de treinamento e posteriormente na utilização para o reconhecimento de gestos dinâmicos da Libras.

A arquitetura da HMM é definida pelo arranjo de interconexões entre os estados que é definida pela matriz de probabilidade de transições (A). Existem várias formas de se definir estas interconexões, as mais comumente utilizadas estão descritas na Tabela V.

Em nosso projeto utilizaremos o modelo Left-Right, pois gestos dinâmicos possuem uma sequência definida de eventos que não permitem transições de retorno nos modelos HMM.

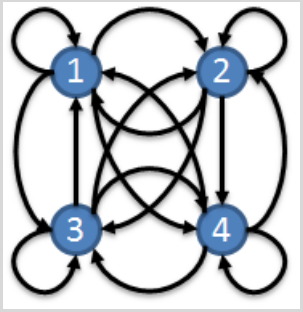
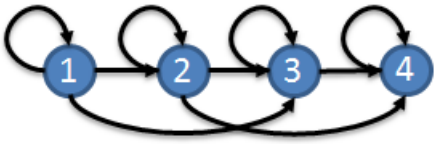
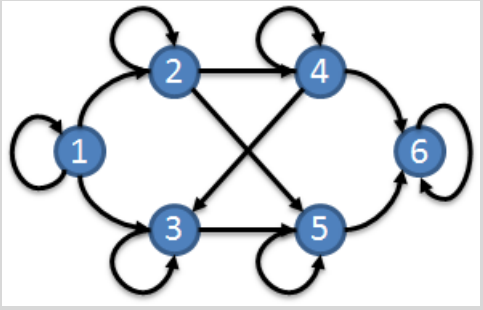
Tipo	Representação	Matriz de Transição
Ergodic		$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$
Left-Right		$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$
Parallel Left-Right		$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & a_{24} & a_{25} & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} & 0 \\ 0 & 0 & 0 & a_{44} & 0 & a_{46} \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & 0 & a_{66} \end{bmatrix}$

Tabela V - Exemplos de possíveis arquiteturas para a HMM.

Capítulo 5

SOFTWARE GESTURE USER INTERFACE (GESTUREUI)

Este capítulo apresenta o software GestureUI que é uma ferramenta para reconhecimento de gestos estáticos e dinâmicos que pode ser utilizada para controlar interfaces multimodais.

5.1 Considerações Iniciais

Com o objetivo de facilitar as pesquisas na área de reconhecimento de gestos o software Gesture User Interface (GestureUI) foi concebido. Este aplicativo encapsula várias técnicas já desenvolvidas durante nossas pesquisas facilitando a reutilização das mesmas por outros pesquisadores no Departamento de Computação da Universidade Federal de São Carlos.

A ideia do software é que o pesquisador possa escolher o método de segmentação, configurar os algoritmos de tracking e modelagem e também escolher qual tipo de gesto pretende reconhecer: Estático ou Dinâmico. O sistema também permite enviar informações sobre os gestos e as ações do usuário através de um protocolo TCP/IP, permitindo assim integração com quaisquer tipos de interfaces multimodais que estejam em um dispositivo com acesso a rede.

Nas próximas seções serão apresentadas a metodologia de desenvolvimento e as funcionalidades deste sistema, assim como algumas aplicações em que ele foi utilizado.

5.2 Metodologia de desenvolvimento

Para o desenvolvimento do software, focamos na utilização de ferramentas que possibilitassem a compilação do código em computadores com sistema operacional Windows ou Linux.

Devido ao requisito de desempenho em tempo real, a linguagem escolhida foi o C++, em conjunto com as bibliotecas de extensão Boost Libraries[11] para manipulação de arquivos em modo multi-plataforma e também para comunicação em rede através do protocolo TCP/IP.

Para processamento de imagens utilizamos a biblioteca OpenCV[12], uma robusta biblioteca de funções de visão computacional e processamento de imagens desenvolvida inicialmente pela Intel e hoje mantida pela comunidade científica da área. Esta biblioteca oferece uma estrutura básica de manipulação de imagens juntamente com funções otimizadas para plataformas Intel. A OpenCV também possibilita a captura de frames da grande maioria de webcams do mercado.

Para integração com o Kinect, ou quaisquer dispositivos desenvolvidos utilizando a tecnologia de estimação de profundidade da PrimeSense, utilizamos o middleware OpenNI[13].

O OpenNI (Open Natural Interaction) é um middleware especialmente desenvolvido para interação natural que permite que o programador possa manipular funcionalidades de um sistema multimodal que envolvam gestos e comandos de voz, sem se preocupar com o hardware a ser utilizado. Sua estrutura permite a adaptação do sistema de acordo com os dispositivos de captura de interação (Kinect, microfones, Asus Xtion, etc) disponíveis no momento da execução do software.

Para desenvolvimento de uma interface com o usuário, optamos pela utilização do WxWidgets[14] que é uma biblioteca de desenvolvimento de interfaces multiplataforma.

---

[11] BOOST C++ Libraries. Disponível em: < http://www.boost.org/ >. Acessado em: 25/08/2012.

[12] OpenCV. Disponível em: < http://sourceforge.net/projects/opencvlibrary/ >. Acessado em: 25/08/2012.

[13] OpenNI. Disponível em: < http://www.openni.org/ >. Acessado em: 25/08/2012.

[14] WxWidgets. Cross Platform Graphical User Interface Library, Disponível em: < http://www.wxwidgets.org/ >. Acessado em: 25/08/2012.

Como utilizamos C++ e WxWidgets optamos pela utilização do Code::Blocks[15] como IDE (Integrated Development Environment) C++ e do wxFormBuilder[16] como IDE para construção de interfaces WxWidgets. Ambas as soluções também são multiplataforma.

Finalmente para armazenar as configurações de interfaces de interação através de gestos utilizamos uma estrutura de arquivos em XML utilizando a biblioteca PugiXML[17].

5.3 Funcionalidades

Nesta seção as principais funcionalidades do sistema foram subdivididas em quatro categorias: 1) Segmentação; 2) Reconhecimento de Gestos; 3) Coleta de Amostras; e 4) Protocolo de Comunicação. As mesmas serão descritas nas subseções a seguir.

5.3.1 Segmentação

O usuário inicialmente pode escolher entre duas formas de captura de gestos, sendo elas (exemplificadas na Figura 39):

1. **Luvas Coloridas:** utilizando luvas coloridas e uma câmera de baixo custo o usuário pode interagir com o sistema.
2. **Câmera de Profundidade:** qualquer dispositivo baseado na tecnologia da PrimeSense de estimação de profundidade que seja compatível com o middleware OpenNI pode ser utilizado para interagir com o sistema de forma natural.

Com a solução utilizando luvas coloridas, permitimos que o usuário possa interagir através de gestos estáticos configurando uma cor de luva para cada mão. Esta funcionalidade permite a utilização do sistema com câmeras de baixo custo.

[15] Code::Blocks. The open source, cross platform, free C++ IDE. Disponível em: < http://www.codeblocks.org/ >. Acessado em: 25/08/2012.

[16] WxFormBuilder. A RAD tool for wxWidgets GUI design, Disponível em: < http://sourceforge.net/projects/wxformbuilder/ >. Acessado em: 25/08/2012.

[17] PugiXML. Light-weight, simple and fast XML parser for C++ with XPath support. Disponível em: < http://code.google.com/p/pugixml/ >. Acessado em: 25/08/2012.

Porém, o sistema precisa ser instalado em ambientes internos bem iluminados e precisa sempre de ajustes em casos de alteração da iluminação ambiente.

Utilizando câmeras de profundidade o sistema se torna mais robusto e permite a interação através de gestos estáticos ou dinâmicos sem a necessidade de nenhuma configuração inicial, permitindo que o usuário possa usufruir das interfaces gestuais assim que o mesmo entre no campo de visão do dispositivo.

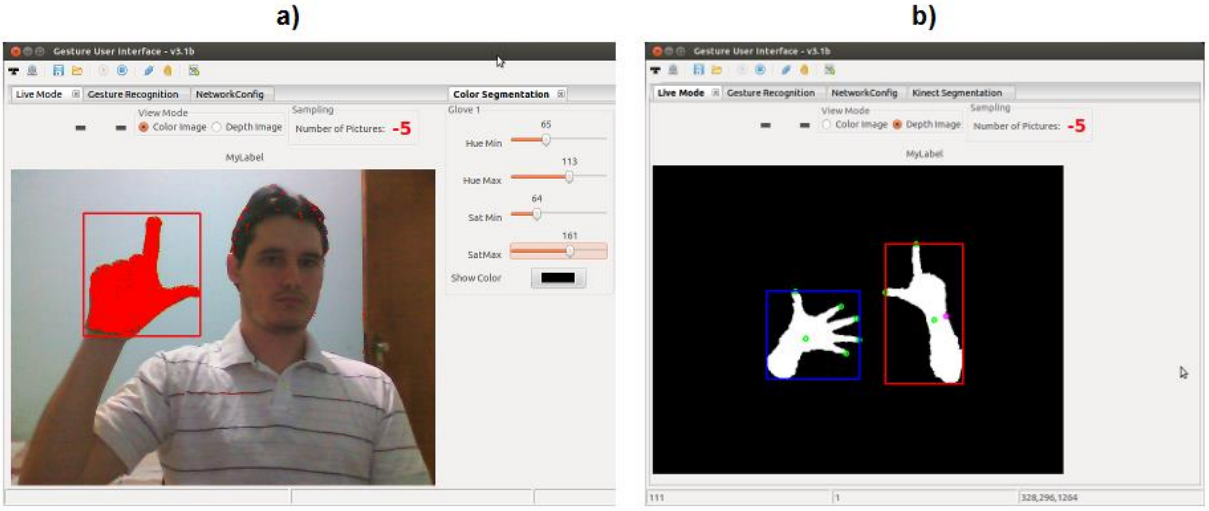


Figura 39 - Exemplos de tipos de Segmentação do Gesture UI. a) Luvas coloridas; b) Câmera de profundidade.

5.3.2 Reconhecimento de Gestos

Após definir e configurar o método de segmentação, o usuário deve definir qual tipo de gestos pretende utilizar para interagir com as interfaces remotas.

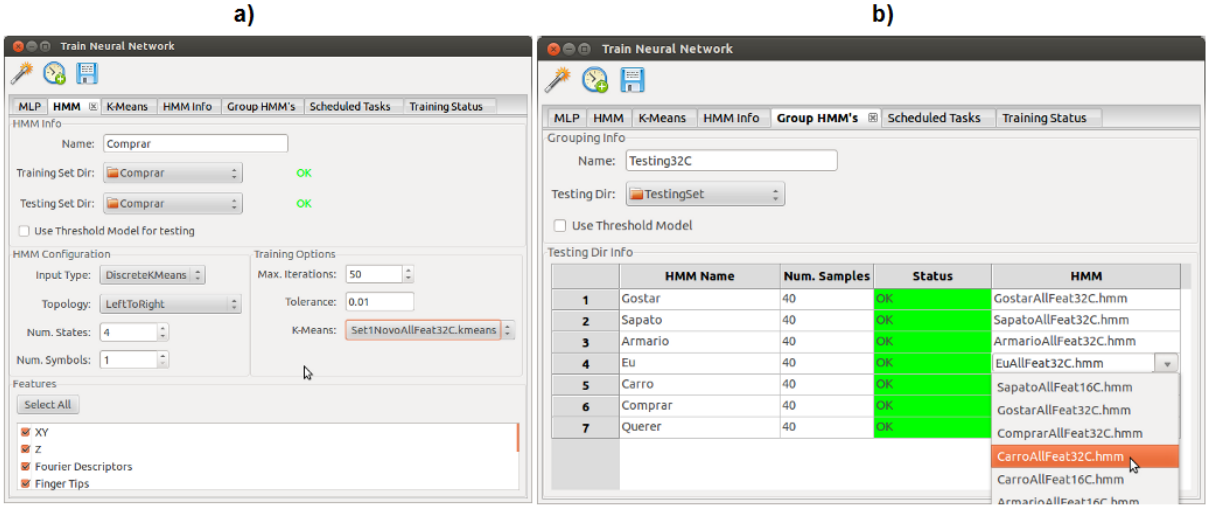


Figura 40 - Exemplos de telas de treinamento de gestos. a) Treinamento da HMM; b) Avaliação de agrupamentos de HMMs previamente treinadas.

O usuário pode escolher entre dois tipos de gestos:

- **Gestos Estáticos:** posturas estáticas da mão que podem ser reconhecidas utilizando a MLP (Multi-Layer Perceptron). Este classificador pode ser treinado através do sistema selecionando um conjunto de treinamento e teste e definindo todos os parâmetros do modelo.
- **Gestos Dinâmicos:** gestos que envolvam movimento também podem ser treinados pelo sistema através da HMM (Hidden Markov Model) implementada, permitindo também que o usuário treine e avalie a performance do classificador através do sistema (como exemplificado da Figura 40) selecionando também um conjunto de treinamento e teste. O usuário pode optar pelo reconhecimento de um gesto isolado ou de um grupo de gestos dinâmicos.

5.3.3 Coleta de Amostras

Para o treinamento dos classificadores de gestos estáticos e dinâmicos desenvolvemos uma ferramenta de auxílio à coleta e organização de amostras.

O usuário deve definir um nome para o gesto estático ou dinâmico, uma pasta onde pretende salvar as informações e a quantidade de amostras que deseja coletar.

O processo de coleta de amostras ocorre em tempo real de formas diferentes para cada tipo de gesto que serão descritas a seguir.

5.3.3.1 Gestos Estáticos

Para coletar amostras de gestos estáticos o usuário pode definir um intervalo de tempo entre a captura das amostras. O sistema então irá fazer sucessivas contagens regressivas até que a quantidade total de amostras seja obtida, permitindo que o usuário possa posicionar as mãos e aguardar pela próxima captura.

Cada postura estática é salva como um arquivo de imagem já preparado para posterior utilização no treinamento da rede neural.

5.3.3.2 Gestos Dinâmicos

Para o processo de coleta de amostras de gestos dinâmicos o usuário pode escolher as características que deseja utilizar para construir o descritor dos gestos como descrito na seção 3.8.5.

O sistema armazena para cada gesto dinâmico, todos os vetores de características extraídos ao longo da sua execução em um único arquivo, onde cada linha representa um vetor de características completo.

Para cada conjunto de amostras coletadas, os arquivos de definição dos gestos dinâmicos são salvos em uma pasta com o nome do gesto e um arquivo XML com as informações sobre as amostras coletadas como: Dimensão total do vetor de características; quais características estão presentes nas amostras; quantos valores representam cada uma destas características; e o nome do gesto.

O sistema precisa saber quando um novo gesto dinâmico começou, para iniciar a captura em tempo real dos vetores de características em memória, e posteriormente quando este gesto terminar finalmente salvar estes dados em disco.

Para identificar que um gesto dinâmico iniciou duas condições precisam ser satisfeitas:

1. Ao menos uma das mãos deve estar visível;
2. Ao menos uma das mãos visíveis deve estar em movimento.

Para identificar que um gesto dinâmico terminou uma das duas regras abaixo deve ser satisfeita:

1. **Ausência de mãos:** Quando ambas as mãos não estiverem mais no campo visual o gesto dinâmico é encerrado.
2. **Ausência de Movimento:** Quando as mãos visíveis param de se mover por um intervalo de tempo, o gesto é encerrado.

5.3.4 Protocolo de Comunicação

Para permitir que os gestos estáticos e dinâmicos reconhecidos possam ser utilizados para controlar interfaces multimodais, definimos um protocolo de comunicação TCP/IP.

Este protocolo é simples e envia informações para um destinatário (I.P. e Porta) a cada quadro processado da imagem com as seguintes informações:

- Quais mãos estão presentes;
- Gestos dinâmicos ou estáticos que foram reconhecidos neste quadro;
- Posição 3D do usuário;
- Posição 3D de cada uma das mãos presentes.

Desta forma o programador pode utilizar estas informações para manipular interfaces implementadas em computadores ou dispositivos móveis.

5.4 Aplicações

O GestureUI foi testado e validado em aplicações reais de interação com interfaces por meio de gestos. Algumas destas aplicações exemplo serão apresentadas nesta seção.

Sebastian Feuerstack criou um framework de interação multimodal chamado Mint[18] que tem por objetivo o auxílio no design e implementação destas interfaces permitindo que vários tipos de entrada possa ser utilizadas como por exemplo: Voz, Controle do Wii, Gestos, Mouse, Teclado, Telas Touch, etc.

Neste projeto o GestureUI foi utilizado nos trabalhos de avaliação de performance e usabilidade utilizando gestos como entrada de interfaces web.

Um exemplo é mostrado na Figura 41 onde o usuário interage com uma interface web, exemplificada em a), através dos gestos mostrados em b). Com os gestos o usuário deve navegar entre as opções até atingir o item em destaque e selecioná-lo.

Esta aplicação exemplo resultou nas três seguintes publicações: Informatik [FEUERSTACK et al., 2011]; IHC 2011 [FEUERSTACK et al., 2011]; I-Com 2011 (Alemanha) [FEUERSTACK et al., 2011].

[18] MINT. The Multimodal Interface Framework. < http://www.multi-access.de/ >. Acessado em: 26/08/2012.

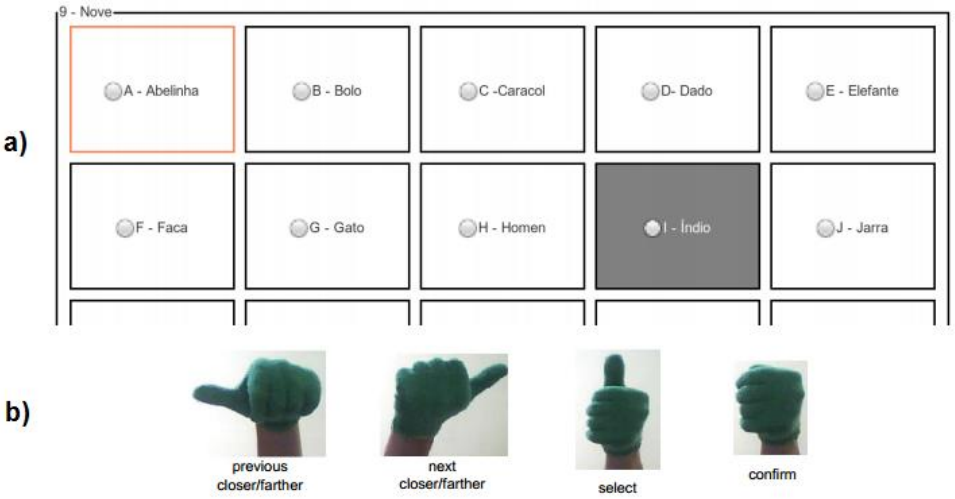


Figura 41 - Exemplo de forma de interação utilizando o Mint e o GestureUI. a) Interface web sendo controlada; b) gestos para navegação nos itens da interface.

Oliveira e Feuerstack desenvolveram uma aplicação chamada “The Augmented Drag-and-Drop”, demonstrada na Figura 42, onde o usuário deve escolher entre objetos apresentados em uma interface web a), e pode arrastar este objeto para o ambiente de Realidade Aumentada b) e posicionar este objeto no espaço 3D.

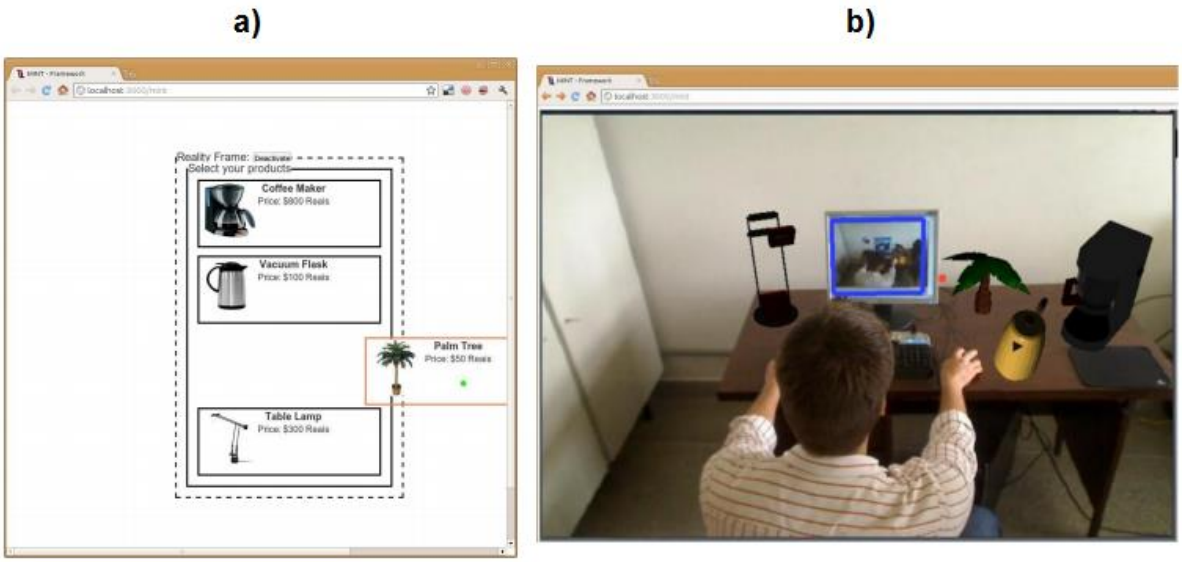


Figura 42 - Aplicação de Realidade Aumentada (RA) utilizando Mint e GestureUI. a) Interface web com objetos disponíveis; b) cena de RA com os objetos selecionados.

Inicialmente este software utilizava apenas entrada do mouse ou do controle do Nintendo Wii. Agora ele foi adaptado para utilizar o GestureUI, permitindo que o usuário possa facilmente arrastar e posicionar os objetos virtuais no espaço de

Realidade Aumentada através de gestos e informações de profundidade obtidas através do Kinect.

Os resultados com este novo arranjo são promissores e serão publicados em breve.

Capítulo 6

RESULTADOS GERAIS

Este capítulo apresenta os resultados gerais obtidos pelo sistema de reconhecimento de gestos da Libras implementado, juntamente com a metodologia utilizada.

6.1 Considerações Inicias

Neste capítulo apresentaremos resultados obtidos no reconhecimento de gestos estáticos e dinâmicos da Libras, avaliando também o desempenho do sistema para o requisito de tempo real proposto neste projeto.

O reconhecimento de gestos estáticos, apresentado em [PIZZOLATO et al., 2010] utilizando 27 posturas, será reavaliado em um conjunto menor de gestos para garantir maiores taxas de reconhecimento, possibilitando assim a avaliação do mesmo em tempo real sem ocasionar falhas de reconhecimento durante a interação do usuário.

Posteriormente o reconhecimento de gestos dinâmicos da Libras será avaliado em um conjunto de sete gestos e outro de 14 gestos, para verificar a relevância e eficiência das características escolhidas para compor o descritor dos gestos.

Finalmente, avaliaremos o desempenho do sistema como um todo mostrando seus tempos de execução para as tarefas de Segmentação, Modelagem (Análise) e Reconhecimento.

Os resultados aqui apresentados tem como restrição de tempo real de execução a capacidade do sistema em processar cada quadro (frame) capturado no

tempo mínimo de 40 milissegundos e tomar uma decisão com uma frequência mínima de 25Hz.

6.2 Reconhecimento de gestos estáticos em tempo real

Em nossos trabalhos anteriores de reconhecimento de gestos estáticos da Libras em [ANJO, 2009] e [PIZZOLATO et al., 2010] reportamos nossos resultados para 27 gestos estáticos, executados em ambiente com fundo controlado, através de médias gerais de reconhecimento e não avaliamos este sistema no requisito de tempo real.

Apresentaremos aqui uma avaliação do reconhecimento de gestos estáticos em tempo real utilizando o novo sistema de reconhecimento com o Kinect. Primeiramente será explicado o funcionamento do sistema mostrando resultados de reconhecimento dos classificadores treinados, e posteriormente na Seção 6.4 apresentaremos a avaliação de tempo de execução deste sistema.

6.2.1 Segmentação e Conjunto de Amostras

Para avaliação deste sistema em tempo real decidimos reduzir o conjunto de gestos estáticos para superar o problema de baixas taxas individuais de reconhecimento em grandes grupos de gestos, como ocorreu na arquitetura apresentada em nosso trabalho anterior [PIZZOLATO et al., 2010].

Optamos então por dois conjuntos de gestos estáticos (A, E, I, O, U) e (B, C, F, L, V), exemplificados na Figura 43, sendo o primeiro composto por todas as vogais do alfabeto Libras e o segundo por cinco consoantes.

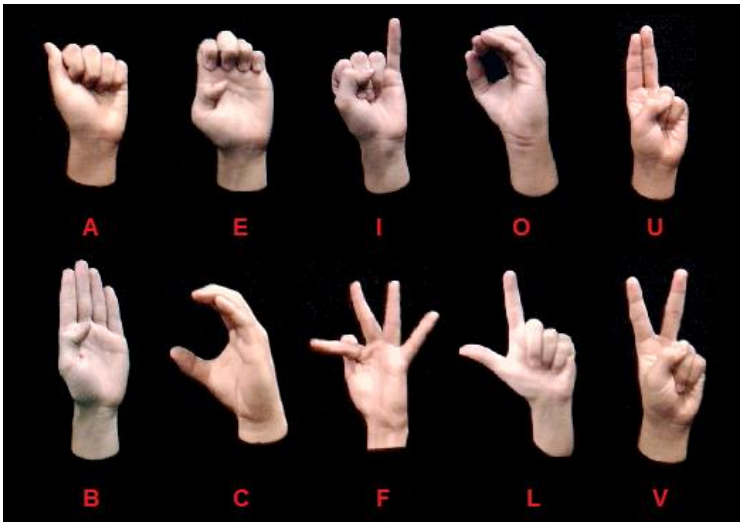


Figura 43 - Gestos estáticos selecionados para testes de reconhecimento em tempo real.

A representação dos gestos para classificação foi implementada de acordo com o trabalho apresentado em [ANJO, 2009], onde tínhamos por objetivo a comparação da classificação de padrões visuais utilizando as redes neurais Multi-Layer Perceptron (MLP) e Neocognitron, e utilizamos imagens binárias com tamanho 25x25 como descritor do gesto sendo executado pelo usuário.

Para transformar a imagem da região que contém a mão do usuário, obtida pelo GestureUI, seguimos o processo demonstrado na Figura 44: a) obtemos a região de interesse que contém a mão do usuário; b) escolhemos a maior dimensão da imagem (no caso 284) para redimensionar para 25 pixels, possibilitando assim um redimensionamento sem quebra de proporção da imagem; c) Centralizamos o padrão para obter uma imagem quadrada de 25x25.

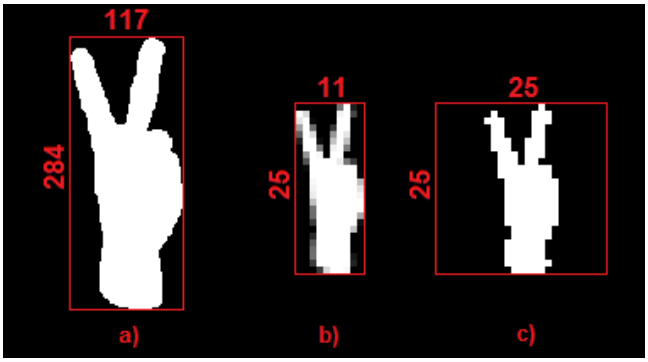


Figura 44 - Passo a passo da preparação da imagem para processamento pela rede neural. a) Imagem original; b) Redimensionamento; c) Threshold e centralização.

Ao submeter as imagens obtidas ao processo de treinamento e teste do classificador de gestos estáticos, percebemos um problema de perda de informações quando o braço do usuário passa a fazer parte da região de interesse.

Utilizando o sistema de segmentação com o Kinect o braço do usuário pode estar visível, principalmente quando o mesmo estiver efetuando gestos do alfabeto da Libras onde o posicionamento da mão e braço deve ser sempre aproximadamente perpendicular ao eixo horizontal.

Explorando essa restrição de domínio, definimos um algoritmo simples chamado *Aspect Ratio Hand Cropping Algorithm (ARHCA)* [ANJO et al., 2012] que tem por objetivo eliminar parte do braço visível baseando-se na elongação ($Aspect\ Ratio$) da região de interesse (ROI) da seguinte forma:

$$Aspect\ Ratio = \frac{ROI_{height}}{ROI_{width}}$$

$$ROI_{height} = \begin{cases} CoM_y + \alpha, & Aspect\ Ratio > \beta \\ ROI_{height}, & caso\ contrário \end{cases}$$

Onde ROI_{width} e ROI_{height} são as dimensões da região de interesse; CoM_y é o centro de massa da mão; β é o limiar para definir se o ajuste de elongação deve ser efetuado ou não; α é um deslocamento adicional em relação ao eixo y da região de interesse. Através de testes empíricos definimos $\beta = 1{,}34$ e $\alpha = 1{,}05 * ROI_{height}$.

O problema de perda de informação solucionado pelo algoritmo *ARHCA* pode ser observado na Figura 45. Em a) e c) mostramos dois exemplos onde o braço passou a fazer parte da região de interesse e, devido ao redimensionamento para 25x25 a mão passou a ser representada por uma menor quantidade de pixels. Já em b) e d) exemplificamos os resultados obtidos após a utilização do algoritmo *ARHCA*.

Resultados comparativos do desempenho do classificador utilizando ou não o algoritmo *ARHCA* serão apresentados na próxima subseção.

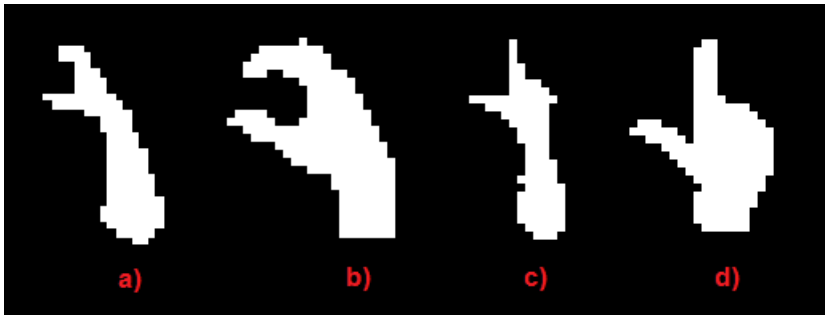


Figura 45 - Exemplo dos problemas encontrados com a presença do braço na região de interesse a) e c) e a solução utilizando o algoritmo ARHCA.

6.2.2 Classificador Multi-Layer Perceptron

Para classificar os gestos estáticos escolhidos optamos pela utilização do classificador Multi-Layer Perceptron (MLP) implementado em Anjo [ANJO, 2009]. A MLP é uma rede neural baseada na Perceptron que consiste em camadas formadas por neurônios em um grafo orientado, onde o sinal é propagado da entrada até as camadas de saída e cada camada é totalmente conectada com a camada anterior. As camadas entre a camada de entrada e a camada de saída são chamadas de camadas escondidas.

Cada neurônio que compõe a MLP nas camadas escondidas possui uma função de ativação, que consiste em mapear as entradas do mesmo em uma saída numérica correspondente ao grau de "excitação". Várias funções são utilizadas, mas neste trabalho optamos pela tangente hiperbólica que segundo Karlik [KARLIK et al., 2010] possibilita melhores resultados. A mesma é definida por:

$$s_n = bias_n + \sum_{j=0}^{N} w_{nj} * y_{nj}$$

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

$$f_n(\cdot) = \tanh(s_n)$$

Onde s_n é a soma ponderada (w_{nj}) das entradas (y_{nj}) com um viés ($bias_n$); $\tanh(x)$ é a função tangente hiperbólica que retorna um valor no intervalo [-1,1]; e finalmente $f_n(\cdot)$ é a função de ativação do neurônio.

Para treinamento utilizamos um algoritmo supervisionado, que consiste em apresentar padrões rotulados e mapear a saída desejada do modelo, facilitando assim a convergência do mesmo para uma solução aceitável. O algoritmo Backpropagation foi o método escolhido por ser o principal método utilizado pelos pesquisadores da área segundo Karlic [KARLIC et al., 2010].

O método de treinamento Backpropagation consiste na propagação do sinal da entrada até a última camada, onde podemos calcular o erro em relação à saída desejada e retropropagar este erro através de ajustes nos pesos da camada de saída até a camada de entrada. Este ajuste de pesos (Δw_{nj}) é feito da seguinte forma:

$$\Delta w_{nj} = \eta * o_n * \delta_n$$

$$\delta_n = o_n * (1 - o_n) * (d_n - o_n) \quad (camada\ de\ saída)$$

$$\delta_n = o_n * (1 - o_n) * \sum_{j=1}^{N} w_{nj} * \delta_j \quad (camadas\ escondidas)$$

Onde ɳ é a taxa de aprendizagem; o_n é a saída do neurônio; e d_n é a saída desejada para a entrada apresentada.

Este ajuste de pesos é efetuado até que a média de erro de todos os padrões de treinamento seja minimizada, caracterizando convergência do método. Este algoritmo pode levar um longo tempo para terminar o treinamento, dependendo das taxas de aprendizado escolhidas.

Para evitar estagnação em um mínimo local Rumelhart [RUMELHART et al., 1986] propôs uma alteração na fórmula de ajuste de pesos para que a mesma leve em consideração a alteração de peso do passo anterior adicionando o termum momentum (μ) da seguinte forma:

$$\Delta w_{n(j+1)} = \eta * o_n * \delta_n + \mu * \Delta w_{nj}$$

Onde $\Delta w_{n(j+1)}$ é a atualização de peso atual e Δw_{nj} é a atualização do peso anterior.

As taxas de aprendizagem e o termum momentum (μ) das camadas dos classificadores MLP treinados são apresentadas na Tabela VI.

Layer	ɳ	μ
Input	0.15	0.5
Hidden	0.09	0.6
Output	0.05	0.7

Tabela VI - Taxas de aprendizagem e Termum Momentum de cada uma das camadas do classificador MLP.

Antes de executar o algoritmo de treinamento os pesos precisam ser inicializados. Para reduzir o tempo de convergência e a chance de atingir um mínimo local Nguyen [NGUYEN et al., 1990] propôs um método para inicialização dos pesos chamado Nguyen–Widrow que consiste na inicialização dos pesos com valores randômicos entre -1 e 1 e posterior atualização dos mesmos da seguinte forma:

$$\beta = 0.7 * h^{\frac{1}{i}}$$

$$n = \sqrt{\sum_{i} w_i^2}$$

$$w_{t+1} = \frac{\beta * w_{t+1}}{n}$$

Onde n é a norma euclidiana; β é o ajuste definido pela heurística de Nguyen–Widrow; e w_{t+1} é o peso final após a inicialização.

Com os classificadores treinados, as imagens em 25x25 dos gestos são apresentadas a rede neural MLP com arquitetura representada na Figura 46, onde na camada de entrada temos 625 posições para a imagem, na camada escondida utilizamos 100 neurônios e na camada de saída cinco neurônios correspondentes aos cinco gestos estáticos de cada um dos conjuntos escolhidos (A, E, I, O, U) e (B, C, F, L, V).

Para avaliação da performance dos classificadores criamos dois conjuntos de amostras para treinamento e teste da rede. O conjunto de treinamento é composto, para cada gesto estático, por 150 amostras e o de teste por 250 amostras. Estes conjuntos de treinamento foram construídos utilizando imagens do banco de gestos estáticos obtido no projeto Fapesp [ANJO, 2009] que reúne gestos estáticos de 27 posturas estáticas da Libras por 45 pessoas diferentes. Para otimizar a performance complementamos estes conjuntos com amostras obtidas utilizando o Kinect.

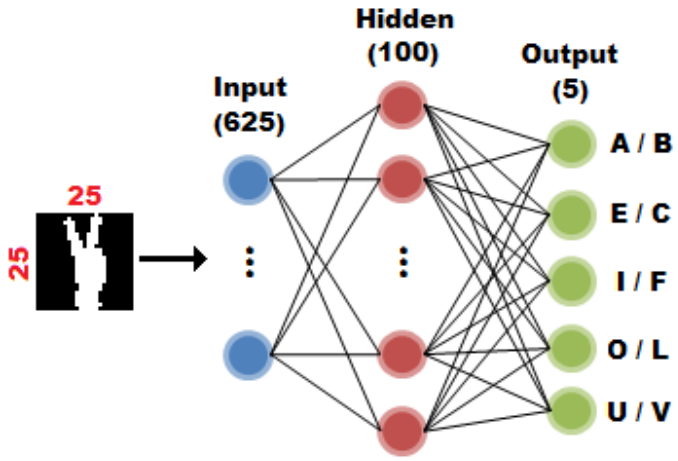


Figura 46 - Arquitetura da MLP para reconhecimento dos dois grupos de gestos estáticos: (A, E, I, O, U) e (B, C, F, L, V).

Os resultados obtidos são apresentados na Tabela VII (A, E, I, O, U) e Tabela VIII (B, C, F, L, V) avaliando as taxas de reconhecimento ao apresentar apenas os conjuntos de teste de todos os gestos estáticos em cada um dos classificadores. Os resultados sem o ARHCA mostram a dificuldade dos classificadores em diferenciar os gestos com a perda de informação, já os resultados com o algoritmo ARHCA evidenciam que a solução proposta foi suficiente para obter 100% de reconhecimento em todas as classes.

Gesto Estático	Sem ARHCA	Utilizando ARHCA
A	65%	100%
E	60%	100%
I	74%	100%
O	70%	100%
U	68%	100%
Mean	**67,4%**	**100%**

Tabela VII - Taxas de reconhecimento para os gestos estáticos A, E, I, O, U com e sem ARHCA.

Gesto Estático	Sem ARHCA	Utilizando ARHCA
B	73%	100%
C	77%	100%
L	80%	100%
F	73%	100%
V	74%	100%
Mean	**75,4%**	**100%**

Tabela VIII - Taxas de reconhecimento para os gestos estáticos B, C, L, F, V com e sem ARHCA.

6.3 Reconhecimento de gestos dinâmicos em tempo real

A tarefa de reconhecimento de gestos dinâmicos envolve maior complexidade tanto na segmentação e modelagem dos dados quanto na interpretação dos mesmos para obtermos como resultado um gesto reconhecido. A linguagem de sinais adiciona novos desafios a este problema como a interpretação do contexto e a estrutura da linguagem.

Segundo Vogler [VOGLER et al., 2008], fazendo um paralelo com o reconhecimento da fala, a segmentação de gestos da linguagem de sinais em “fonemas” ou unidades seria o ideal para modelagem e escalabilidade de sistemas de reconhecimento de linguagem de sinais. Porém, esta tarefa é muito complexa

devido ao alto grau de liberdade das mãos do usuário e de outras variáveis adicionais como: expressões faciais; mudança de significado de gestos quando inseridos em determinados contextos; e variações regionais da linguagem.

Podemos definir formalmente quais são as configurações de postura de mão, movimentos e expressões faciais, mas esta definição pode não ser factível para reconhecimento por um classificador estatístico. Por exemplo, como descrito na seção 4.4 existe um conjunto finito de possíveis configurações de mão (posturas) para a formação de palavras da Libras. Porém, como o usuário deve efetuar movimentos com as mãos e a cena está sendo observada através de uma câmera que gera uma projeção 2D do ambiente, não é viável treinar um classificador para reconhecer estas posturas, pois as mesmas se apresentar-se-ão de formas diferentes devido a mudanças de perspectiva e oclusões.

Ciente destes problemas e da dimensão do desafio de se propor a modelar e reconhecer a Linguagem Brasileira de Sinais (Libras) decidimos nortear nossas pesquisas partindo de um escopo reduzido e com um objetivo de aplicação bem definido que serão apresentados na seção 6.3.1.

Posteriormente apresentaremos a metodologia de testes na seção 6.3.2, e na seção 6.3.3 os resultados obtidos no reconhecimento de algumas palavras da Libras, com uma análise dos componentes dos vetores de características.

6.3.1 Definição de escopo e objetivo

Vamos avaliar neste trabalho a viabilidade do reconhecimento de um conjunto finito de palavras da Libras utilizando vetores de características das mãos do usuário e um classificador Hidden Markov Model (HMM).

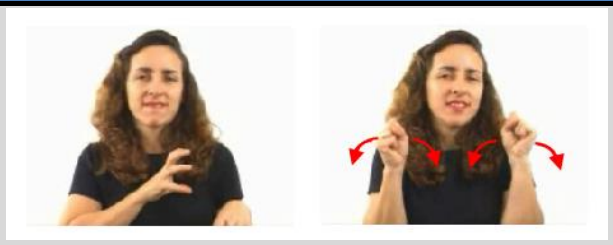
Este sistema tem por objetivo viabilizar o reconhecimento de um conjunto de palavras da Libras em tempo real, que permita posteriormente a confecção de um software de auxílio no ensino da linguagem. Tivemos por motivação a dificuldade encontrada em transferir o conhecimento da Libras através de livros ou apenas vídeos, e da dificuldade de um professor inspecionar a correta execução de gestos por cada um de seus alunos. Desta forma, o software viria para preencher esta lacuna e auxiliar no processo de aprendizagem e disseminação da linguagem.

O escopo dos testes efetuados neste projeto é definido pelos seguintes itens:

1. As palavras escolhidas não deverão envolver expressões faciais.

2. Dentro do conjunto de palavras escolhidas devemos conter casos de oclusão de mão e face; toque de partes do corpo (mão-mão; mão-tórax; etc); gestos só com a mão dominante e gestos com ambas as mãos.

Para descrever os gestos utilizamos as características apresentadas na seção 3.8.5 formando um vetor composto pelas características de cada uma das mãos. Quando uma das mãos estiver ausente, seu vetor será preenchido com zeros até atingir a dimensão total do descritor.

Gesto	Ilustração
Adorar	
Armário	
Carro	
Comprar	
Eu	

Querer	
Sapato	 

Tabela IX - Relação de gestos de palavras da Libras escolhidos para avaliação do sistema de reconhecimento.

Separamos dois grupos de gestos compostos por sete palavras da Libras, para avaliar o impacto no desempenho do reconhecimento dos gestos ao dobrar o número de possíveis palavras a serem classificadas.

O primeiro conjunto de sete gestos com palavras escolhidas da Libras é apresentado na Tabela IX. Podemos observar pela ilustração dos gestos que: *Adorar* e *Eu* possuem o toque da mão no tórax; em *Armário*, *Comprar* e *Sapato* ocorrem toque entre as mãos; e em *Armário* existe a possibilidade de oclusão da face dependendo da amplitude do movimento executado pelo usuário. Portanto, este conjunto de gestos atende a todos os requisitos de escopo definidos anteriormente.

O segundo conjunto de sete gestos é apresentado na Tabela X, contendo também gestos que satisfazem os requisitos de escopo.

Gesto	Ilustração
Banheiro	 
Borboleta	

Casa	
Cesta	
Complicado	
Nós	
Simples	

Tabela X - Relação dos sete gestos adicionais da Libras para avaliação do sistema de reconhecimento.

6.3.2 Metodologia de Testes

Para avaliar a viabilidade de utilização do vetor de características definido na Seção 3.8.5, definimos três casos de teste para os gestos escolhidos onde o vetor de características de cada mão é composto por:

- **Cenário 1:** Todas as características;
- **Cenário 2:** Somente posição 3D e Descritores de Fourier;

- **Cenário 3:** Somente posição 3D e descritores geométricos (Aspect Ratio, Circularity, Spreadness, Roundness, Solidity, Finger Tips).

O vetor de características no primeiro caso tem dimensão igual a 80, no segundo caso reduzimos para 68 dimensões, e no terceiro caso para 18 dimensões.

Para cada um dos 14 gestos apresentados na Tabela IX e Tabela X foram coletadas 40 amostras de cada palavra para o conjunto de treinamento e 40 amostras de cada palavra para o conjunto de testes. Estas amostras são coletadas através do GestureUI, e são arquivos de texto onde cada linha representa o vetor de características de cada frame capturado durante a amostragem do gesto da Libras.

Na coleta de amostras, para garantir variabilidade suficiente dos dados para posterior análise de viabilidade, obtivemos as seguintes variações:

- Posição em relação à câmera;
- Velocidade de execução dos gestos;
- Rotação moderada do usuário para avaliar mudanças de perspectiva.

Para coletar estas amostras o usuário deve ficar sentado no campo de visão do Kinect para executar cada um dos gestos da Libras propostos aqui para reconhecimento.

Após obter os conjuntos de treinamento e teste, todas as amostras de treinamento são utilizadas para clusterização utilizando K-Means. Para cada um dos casos de teste descritos anteriormente obtivemos um K-Means com 16 Clusters e outro com 32 Clusters.

Para cada um dos gestos foi treinado um modelo HMM distinto, onde o número de estados do modelo foi definido de acordo com o desempenho do mesmo para classificar o conjunto de treinamento.

Visando avaliar este desempenho individualmente utilizamos o algoritmo de Viterbi [RABINER, 1989], definido na Seção 4.7.2, para verificar se para cada amostra de treinamento o modelo conseguiu produzir um caminho de estados até o estado final. Durante o treinamento, o número de estados era incrementado até que o melhor desempenho fosse encontrado.

6.3.3 Resultados de Reconhecimento de palavras da Libras

Inicialmente avaliamos o desempenho dos classificadores utilizando apenas sete gestos. No primeiro caso, onde o descritor dos gestos é composto por todas as características, apresentamos todas as amostras dos conjuntos de treinamento e teste a cada um dos classificadores HMM dos sete gestos.

A Figura 47 mostra as taxas de reconhecimento dos classificadores utilizando 16 clusters no K-Means com Binary Split. A média da taxa de reconhecimento para este experimento foi de 98,39%, onde somente o gesto *Eu* ficou abaixo da média geral.

O gráfico apresentado na Figura 48 representa das taxas de reconhecimento do primeiro experimento utilizando 32 clusters no K-Means. A média de reconhecimento foi de 99,82%, evidenciando uma melhora em relação ao resultado do gráfico da Figura 47 devido a melhor representação dos dados do espaço amostral utilizando mais clusters.

Apesar de neste primeiro cenário de testes os resultados obtidos serem satisfatórios, os dois próximos cenários foram definidos para avaliar o desempenho dos classificadores na ausência de descritores geométricos ou descritores de Fourier.

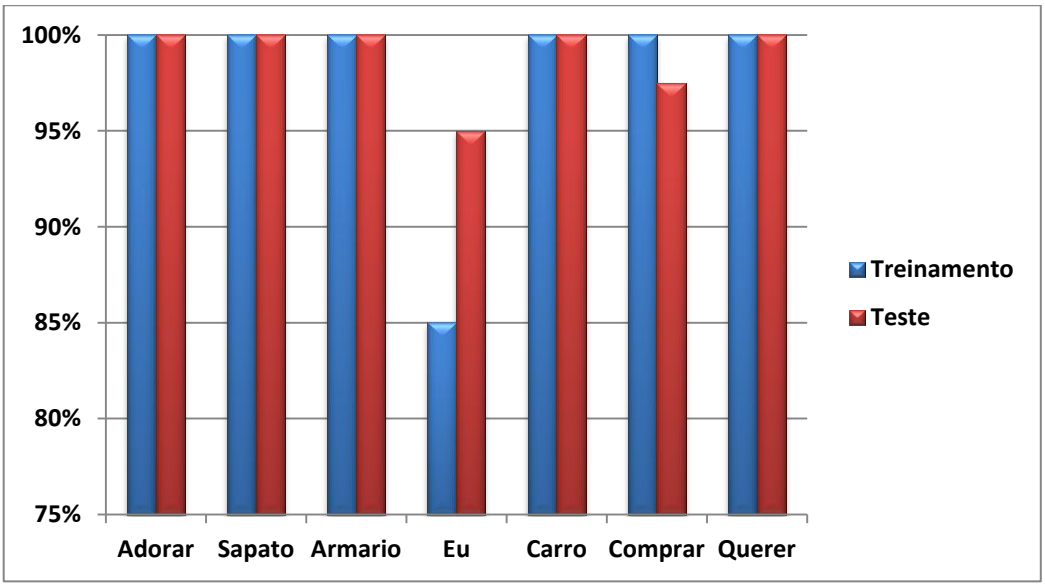


Figura 47 - Gráfico das taxas de reconhecimento de cada um dos gestos dinâmicos para os conjuntos de Treinamento e Teste utilizando 16 clusters no K-Means.

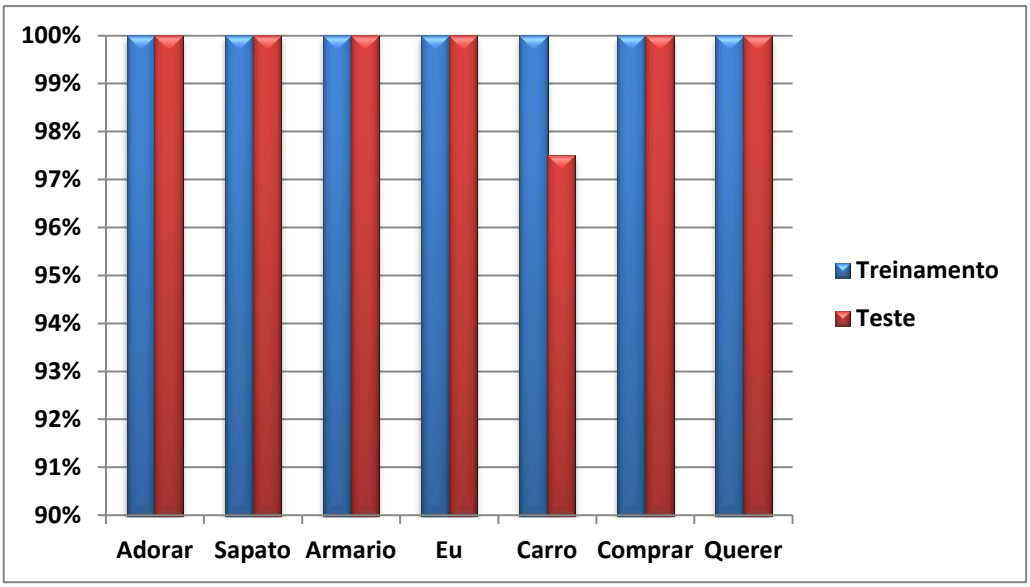


Figura 48 - Gráfico das taxas de reconhecimento de cada um dos gestos dinâmicos para os conjuntos de Treinamento e Teste utilizando 32 clusters no K-Means.

Os resultados no cenário 2, onde não utilizamos descritores geométricos, são apresentados no gráfico da Figura 49. Agrupamos no mesmo gráfico os resultados utilizando 16 clusters e 32 clusters no K-Means. As médias das taxas de reconhecimento utilizando 16 clusters e 32 clusters foram 12,50% e 14,82% respectivamente.

Este cenário de testes evidencia um problema na diferenciação dos gestos utilizando somente Descritores de Fourier em conjunto com a posição 3D. O algoritmo K-Means de clusterização não pôde obter bons clusters que subdividissem o espaço amostral. Este problema fica mais claro na Tabela XI que representa a matriz de confusão da classificação dos gestos dinâmicos da Libras utilizando 16 clusters. Pode-se perceber nesta matriz uma concentração de erros de classificação no gesto Sapato, mostrando que o alfabeto gerado pelo algoritmo K-Means não foi suficiente para que a HMM pudesse modelar a sequência de observações.

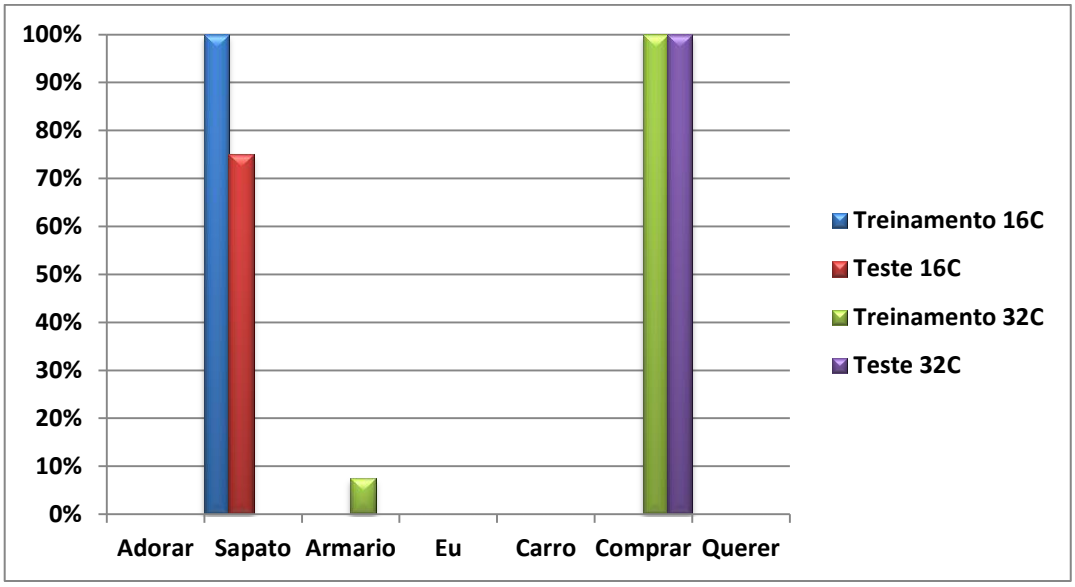


Figura 49 - Gráfico das taxas de reconhecimento dos gestos dinâmicos utilizando somente a posição 3D e Fourier Descriptors no vetor de características.

	Adorar	Sapato	Armário	Eu	Carro	Comprar	Querer
Adorar	0	80	0	0	0	0	0
Sapato	5	60	15	0	0	0	0
Armário	0	80	0	0	0	0	0
Eu	0	80	0	0	0	0	0
Carro	74	6	0	0	0	0	0
Comprar	0	74	6	0	0	0	0
Querer	0	80	0	0	0	0	0

Tabela XI - Matriz de confusão na classificação dos gestos dinâmicos utilizando no vetor de características apenas Descritores de Fourier e a Posição 3D.

O terceiro e último cenário avaliado, onde eliminamos apenas os Descritores de Fourier do vetor de características, tem seus resultados apresentados no gráfico da Figura 50. Os classificadores HMM obtiveram uma média de 99,64% na taxa de reconhecimento, evidenciando que os descritores geométricos foram suficientes, neste cenário, para diferenciar os sete gestos dinâmicos da Libras.

Só foi possível avaliar o sistema utilizando 16 clusters no K-Means, pois ao executar o mesmo com o algoritmo Binary Split utilizando 32 clusters obtivemos oito clusters vazios, ou seja, durante o processo de subdivisão de clusters oito deles não tiveram nenhuma amostra associada. Isto ocorreu devido à baixa dimensionalidade do vetor de características neste cenário (18 dimensões), impossibilitando assim a obtenção de uma clusterização passível de ser avaliada com o classificador HMM.

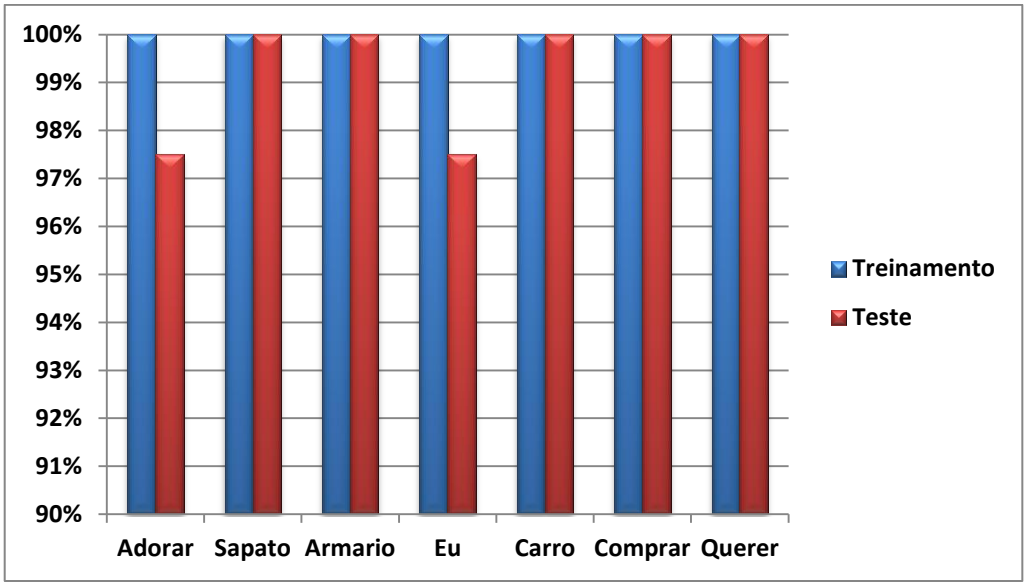


Figura 50 - Gráfico das taxas de reconhecimento dos gestos dinâmicos utilizando descritores geométricos e a posição 3D no vetor de características, e 16 clusters no K-Means.

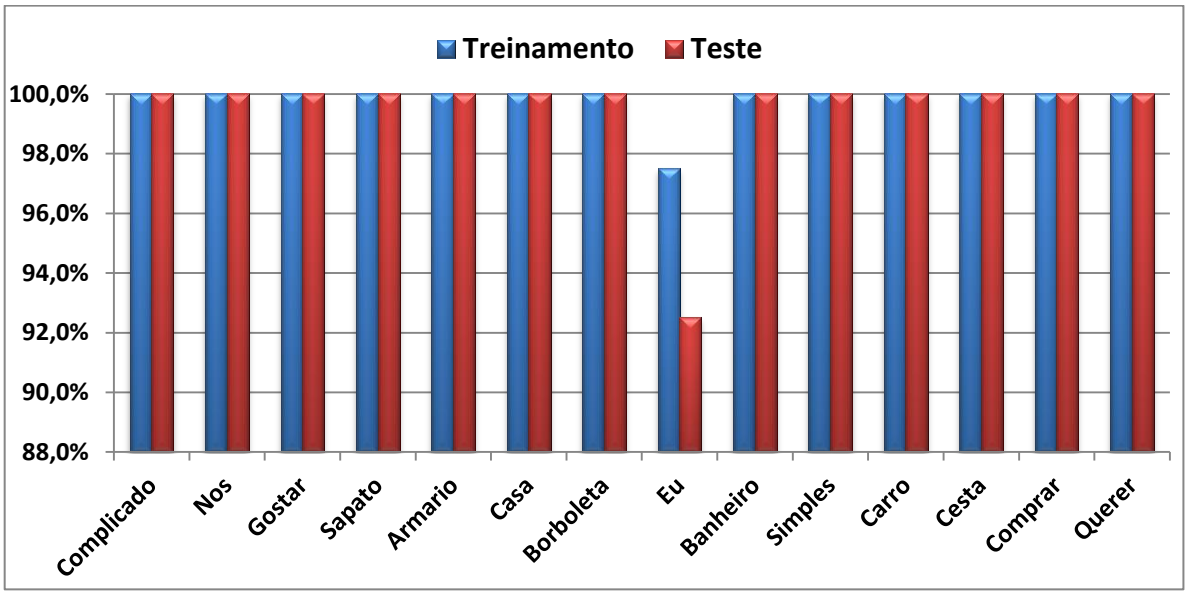


Figura 51 - Gráfico das taxas de reconhecimento dos 14 gestos dinâmicos utilizando descritores geométricos e a posição 3D no vetor de características, e 32 clusters no K-Means.

Após analisarmos o conjunto de sete gestos nos três cenários propostos, concluímos que os melhores resultados foram obtidos utilizando um vetor de características composto por descritores geométricos e a posição 3D. Portanto, ao duplicarmos a quantidade de gestos a serem reconhecidos treinamos e avaliamos apenas um classificador no melhor cenário, sendo os resultados apresentados na

Figura 51. A média de reconhecimento atingida foi de 99,64% utilizando 32 clusters no K-Means e mantendo assim o bom desempenho obtido nos testes anteriores.

6.4 Avaliação da execução em tempo real

Para avaliar os tempos de execução, utilizamos um notebook com processador Intel i7 e subdividimos as tarefas executadas pelo nosso sistema em três etapas principais, sendo elas:

- **Segmentação:** Capturar imagem de profundidade do Kinect; segmentar usuário e aplicar algoritmo de Virtual Wall.
- **Análise:** Detecção de blobs; extrair região de interesse das mãos; calcular vetor de características; e preparar imagens 25x25 das mãos.
- **Reconhecimento:** Tempo para que a rede neural MLP ou os modelos HMM processem a entrada e possam classificar um gesto estático ou dinâmico respectivamente.

Algumas tarefas são executadas em um espaço de tempo da ordem de microssegundos, portanto para contabilizar os tempos de execução das etapas supracitadas utilizamos a biblioteca Posix Time da Boost [BOOST, 2012] para C++ que implementa ferramentas para calcular tempos com precisão de micro e nanosegundos.

Para avaliação dos tempos de Segmentação e Análise em conjunto com o reconhecimento de Gestos Estáticos, foram coletados cinco vídeos com 25 segundos de duração cada. Nestes vídeos, um usuário executou gestos estáticos aleatórios (dentro dos conjuntos de gestos definidos na seção 6.2) se movimentando em frente ao Kinect, e os tempos de cada uma das etapas foram calculados quadro a quadro. Estes vídeos totalizam um total de 3.750 quadros processados.

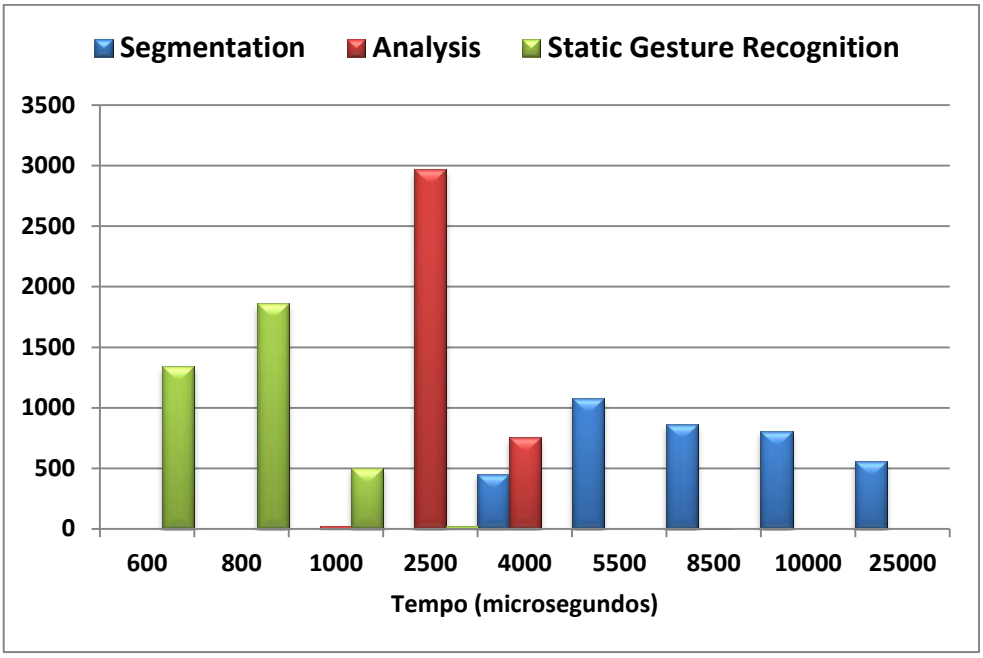


Figura 52 - Histograma de tempos de execução para Segmentação, Análise e Reconhecimento de Gestos Estáticos.

No gráfico da Figura 52 apresentamos um histograma dos tempos de execução, por quadro capturado, das etapas de Segmentação, Análise e Reconhecimento de Gestos Estáticos. Podemos observar uma flutuação dos tempos de execução na etapa de Segmentação, isto ocorre devido às variações na cena observada pelo Kinect quando o usuário se movimenta.

A média e desvio padrão de cada uma das etapas são apresentadas na Tabela XII. Somando estas médias de tempo de execução obtemos um tempo total de processamento por quadro de $10.127{,}05 \mu segundos$, ou seja, o sistema é capaz de reconhecer gestos estáticos a uma frequência de $98{,}1\text{Hz}$. Porém, existe um overhead de aproximadamente seis milissegundos para atualização da interface com um feedback para o usuário (imagem do mesmo fazendo gesto e a postura reconhecida), reduzindo assim a frequência de execução para 62,5Hz.

Etapa	Média	Desvio Padrão
Segmentação	7295,17 µs	3169,06 µs
Análise	2180,62 µs	724,56 µs
Reconhecimento	651,26 µs	131,27 µs

Tabela XII - Média e Desvio Padrão dos tempos de execução das etapas de Segmentação, Análise e Reconhecimento de Gestos Estáticos.

Os gestos dinâmicos não podem ser avaliados quadro a quadro, pois para reconhecê-los é necessário aguardar o processamento de uma sequência de frames. Portanto escolhemos três conjuntos de classificadores HMM treinados para reconhecimento dos sete gestos dinâmicos da Libras apresentados na seção anterior, onde cada um deles utilizou um vetor de características diferente e consequentemente um modelo K-Means diferente, sendo estes: 1) vetor de características sem Descritores de Fourier utilizando 16 clusters; 2) todas as características utilizando 16 clusters; e 3) todas as características utilizando 32 clusters. Por fim, avaliamos o desempenho do sistema de reconhecimento com o conjunto de 14 gestos utilizando o melhor vetor de características, como concluído na seção anterior.

Avaliaremos aqui os tempos de execução no momento final da decisão se uma sequência de observações foi gerada por determinado modelo HMM, pois este configura o pior caso onde o conjunto de observações contém um gesto completo e deve ser apresentado a cada um dos modelos treinados.

O gráfico com os tempos de execução aferidos é apresentado na Figura 53. Como esperado devido às diferenças de dimensionalidade o classificador com 32 clusters no K-Means obteve, em média, os maiores tempos de execução seguido pelo classificador utilizando todas as características com 16 clusters e por último o classificador sem Descritores de Fourier e 16 clusters.

A média e desvio padrão dos tempos de reconhecimento dos conjuntos de classificadores HMM são apresentados na Tabela XIII. Utilizando a média de tempo no pior caso, $11{,}91ms$, podemos avaliar a influência do tempo de processamento do conjunto de classificadores no momento final da decisão pela classificação de um gesto dinâmico. Utilizando as médias de tempo de execução das etapas de Segmentação ($7{,}29ms$) e Análise ($2{,}18ms$) juntamente com o overhead de atualização da interface ($6ms$), obtemos um tempo de processamento de $27{,}38ms$, ou seja, o sistema no pior caso pode reconhecer gestos dinâmicos a uma frequência máxima de $36{,}52Hz$. Já no melhor caso, utilizando um vetor de características sem Fourier Descriptors e 16 Clusters, o sistema consegue reconhecer gestos a uma frequência máxima de $55{,}34Hz$.

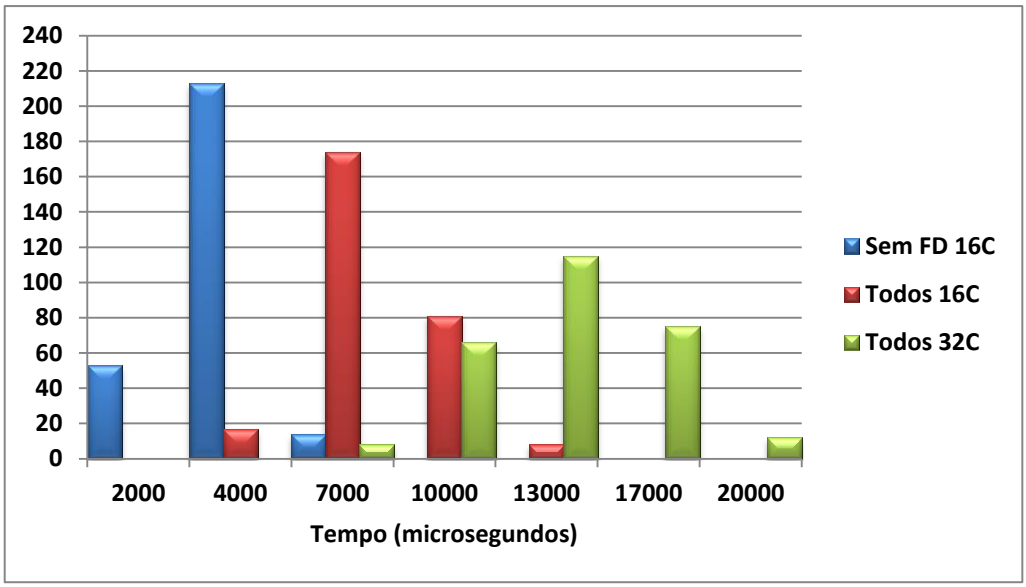


Figura 53 - Histograma de tempos de reconhecimento de gestos dinâmicos na fase final.

Etapa	Média	Desvio Padrão
Sem FD 16C	2,60 ms	0,68 ms
Todos 16C	6,38 ms	1,60 ms
Todos 32C	11,91 ms	3,00 ms

Tabela XIII - Média e Desvio Padrão dos tempos de reconhecimento de gestos dinâmicos da Libras.

Para avaliar o impacto no tempo de execução ao dobrar o número de gestos, também computamos os tempos de reconhecimento para um conjunto de gestos composto por 14 palavras da Libras. O histograma de tempos de execução é apresentado na Figura 54. A média de tempo de reconhecimento neste caso foi de $10{,}77ms$ com desvio padrão de $4{,}24\ ms$. Acrescentando a este tempo os tempos de Segmentação ($7{,}29ms$), Análise ($2{,}18ms$) e overhead de atualização da interface (6ms) atingimos um tempo de ciclo total de $26{,}24ms$, ou seja, o sistema de reconhecimento para 14 gestos tem uma frequência de execução de $38{,}10Hz$.

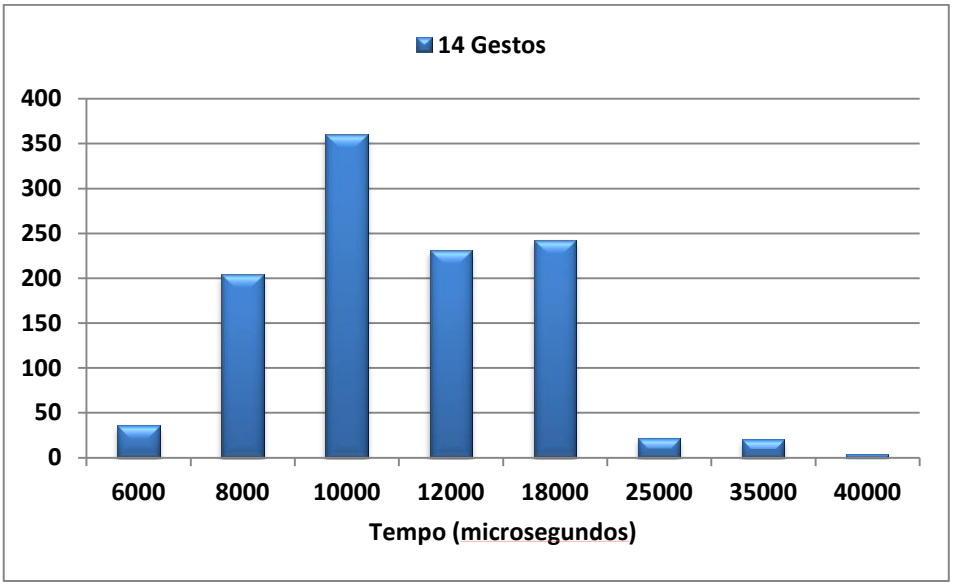


Figura 54 - Histograma de tempo de reconhecimento com 14 Gestos.

Ao dobrar o número de gestos de sete para 14 tivemos também de dobrar o número de clusters no K-Means (16 para 32) e o tempo de execução total aumentou em 45,21%.

Como a quantidade de clusters representa as unidades básicas de construção dos gestos e várias palavras são compostas por unidades similares, em um dado momento, um número finito de clusters será capaz de construir uma grande quantidade de palavras. Portanto, o aumento de tempo de execução ao que acrescentamos novas palavras será cada vez menor a medida que o conjunto de clusters do K-Means for representativo o suficiente.

Devemos também ressaltar que o sistema aqui apresentado não utiliza nenhuma técnica de otimização dos modelos treinados da HMM e também não paraleliza a avaliação dos modelos correspondentes as palavras da Libras treinadas, o que com certeza contribuirá para um aumento considerável do desempenho do sistema em casos de maiores quantidades de gestos.

Apesar dos tempos de execução aqui apresentados, o sistema, com a tecnologia hoje existente, será limitado a uma frequência de processamento de 30Hz devido a taxa máxima de 30 quadros por segundo do Kinect.

Capítulo 7

CONCLUSÕES

Este capítulo apresenta as conclusões deste trabalho, juntamente com as contribuições e trabalhos futuros.

O reconhecimento de gestos vem se popularizando em interfaces multimodais, e ainda carece de boas ferramentas para simplificar tanto as pesquisas na área quanto a implementação de tais interfaces.

O reconhecimento da linguagem de sinais também vem sendo investigado por vários pesquisadores como apresentado nesta dissertação, mas segundo Vogler [VOGLER et al., 2008] ainda está bem distante do reconhecimento de fala. Isto se deve a grande complexidade envolvida na definição de um modelo para a estrutura da linguagem de sinais, e mais que isso este modelo ser compatível com os métodos para reconhecimento de padrões existentes. Esta definição de estrutura facilitaria, assim como facilita no reconhecimento de fala, o crescimento da abrangência do vocabulário e nuances da linguagem reconhecida pelos classificadores.

Além disso, o sistema deve ser capaz de extrair e interpretar todas as informações necessárias, com as tecnologias hoje existentes, em tempo real para ser realmente passível de utilização em sistemas multimodais.

A Linguagem Brasileira de Sinais (Libras) ainda não foi explorada, até o presente momento, com o objetivo de se obter sistemas que pudessem realmente ser utilizados para auxiliar na difusão e ensino da linguagem.

Dentro deste contexto, apresentamos neste trabalho uma avaliação das possíveis soluções para as etapas de *Segmentação*, *Modelagem* e *Reconhecimento* visando um sistema passível de ser utilizado para reconhecimento da Libras em

aplicações diversas atendendo ao requisito da interação natural de execução em tempo real.

Na etapa de *Segmentação*, apresentamos uma avaliação dos métodos já utilizados para o reconhecimento de gestos tanto utilizando câmeras tradicionais quanto câmeras de profundidade. Pôde-se concluir que os algoritmos tradicionalmente utilizados com câmeras do espectro visual não atendem aos requisitos de precisão na morfologia dos objetos segmentados e também execução em tempo real. Portanto propusemos uma solução utilizando o Microsoft Kinect e o algoritmo Virtual Wall, que consiste na criação de uma parede virtual a frente do usuário que segmenta os objetos de interesse que transpassem a mesma. Esta barreira virtual acompanha o usuário enquanto o mesmo se movimenta.

Na etapa de *Modelagem*, avaliamos os métodos utilizados para modelagem das partes do corpo humano para posterior interpretação dos gestos efetuados pelo usuário. Pudemos concluir que a grande maioria dos métodos não é passível de implementação em tempo real. Dentre os métodos avaliados somente os algoritmos de estimação da PrimeSense e Microsoft embutidos em seus respectivos hardwares de captura de imagens de profundidade podem ser executados em tempo real porém com baixa precisão na estimação dos movimentos quando o usuário está próximo ao dispositivo. Portanto, considerando o domínio de aplicação para o reconhecimento da Libras, apresentamos um algoritmo baseado no Virtual Wall que se concentra na segmentação apenas das mãos do usuário e monta um descritor para cada uma delas que pode ser composto por: Fourier Descriptors; Aspect Ratio; Circularity; Spreadness; Roudness; Solidity; Finger tips; e Posição 3D (X,Y,Z). Por utilizar apenas descritores das mãos, nosso sistema só é capaz de ser treinado para gestos que não envolvam expressões faciais ou movimentos com a cabeça.

E finalmente na etapa de *Reconhecimento*, avaliamos os trabalhos já efetuados na área de reconhecimento de gestos, e apresentamos um classificador Hidden Markov Model (HMM) Discreto utilizando K-Means com Binary Split para gestos dinâmicos, e um classificador Multi-Layer Percetron (MLP) para gestos estáticos. Nossos classificadores obtiveram taxas de reconhecimento médias de 100% para cinco gestos estáticos em um conjunto de amostras coletado com 45 pessoas. Já para os gestos dinâmicos a taxa de reconhecimento média dos classificadores foi de 99,64% para sete e 14 gestos em um conjunto de amostras controlado com variações de distância, perspectiva, e velocidade de execução do

gesto para avaliar a viabilidade de utilização do vetor de características apresentado nesta dissertação.

O sistema apresentado aqui é robusto para segmentar e modelar as ações do usuário, e sua viabilidade para aplicação em tempo real depende do tamanho do vocabulário utilizado para treinar os classificadores. Obtivemos frequências de execução de $62{,}5\text{Hz}$ para cinco gestos estáticos, $55{,}34\text{Hz}$ para sete gestos dinâmicos e $38{,}10\text{Hz}$ para 14 gestos dinâmicos.

No caso da utilização de uma grande quantidade de gestos, o sistema precisa ser otimizado através da paralelização de tarefas (avaliação dos modelos treinados em várias linhas de execução paralelas) e da otimização dos modelos treinados pelos classificadores (Ex.: aplicação de um método de prunning/poda nos modelos HMM), para continuar atendendo ao requisito de execução em tempo real proposto nesta dissertação.

Todos os métodos apresentados neste trabalho foram encapsulados no software Gesture User Interface (GestureUI), que fornece uma base de pesquisa para classificadores mais completos da Libras e também uma ferramenta para pesquisas práticas na utilização da linguagem de sinais em aplicações multimodais reais, através do protocolo de comunicação TCP/IP. Este sistema foi validado em aplicações multimodais reais que resultaram, até agora, nas publicações descritas na próxima seção.

Acreditamos que com esta ferramenta e os resultados obtidos aqui, temos várias novas possibilidades de investigação nesta área de pesquisa e possibilitaremos que outros pesquisadores possam explorar a utilização de gestos em várias aplicações multimodais.

7.1 Contribuições

Quatro artigos foram produzidos utilizando os resultados para reconhecimento de gestos apresentados neste trabalho, sendo estes:

1. Anjo, M. S.; Pizzolato, E.; Feuerstack, S.; *A Real-Time System to Recognize Static Hand Gestures of Brazilian Sign Language (Libras)*

alphabet using Kinect. IHC - The 6th Latin American Conference on Human-Computer Interaction, 2012.

2. Feuerstack, S.; Anjo, M. S.; Colnago, J.; Pizzolato, E.; *Modeling of User Interfaces with State-Charts to Accelerate Test and Evaluation of different Gesture-based Multimodal Interactions.* Accepted for Workshop "Modellbasierte Entwicklung von Benutzungsschnittstellen (MoBe2011)", Informatik, 2011.
3. Feuerstack, S.; Anjo, M. S.; Pizzolato, E.; *Model-based Design, Generation and Evaluation of a Gesture-based User Interface Navigation Control.* 5th Latin American Conference on Human-Computer Interaction (IHC 2011), Outubro, 2011.
4. Feuerstack, S.; Anjo, M. S.; Pizzolato, E.; *Modellierung und Ausführung von multimodalen Anwendungen auf Basis von Zustandsdiagrammen.* i-com, Zeitschrift für interaktive und kooperative Medien, Vol. 10 No.3, pp. 40-48, ISSN 1618-162, 2011.

Este trabalho também resultou nas seguintes contribuições:

1. Avaliação dos métodos de captura e segmentação de imagens utilizando câmeras de espectro visual e câmeras de profundidade;
2. Avaliação dos métodos de modelagem de partes do corpo humano utilizando câmeras de espectro visual e câmeras de profundidade;
3. Avaliação de várias características para compor vetores descritores dos gestos dinâmicos;
4. Avaliação do estado da arte no reconhecimento de gestos estáticos e dinâmicos, e implementação de classificadores HMM e MLP totalmente parametrizados;
5. Algoritmo Virtual Wall adaptativo que cria uma barreira virtual a frente do usuário e a ajusta em tempo real sem ser influenciado pelos movimentos das mãos e braços do usuário;
6. Criação da ferramenta GestureUI que encapsula as seguintes funcionalidades:
    a. Métodos de segmentação configuráveis utilizando o Kinect (Virtual Wall) e câmeras de espectro visual com luvas coloridas;

b. Algoritmo para captura e organização de amostras de gestos estáticos e dinâmicos;
c. Interfaces para configuração, treinamento e avaliação de classificadores para gestos estáticos (MLP) e dinâmicos (HMM);
d. Protocolo de comunicação TCP/IP que possibilita o envio de informações sobre as ações do usuário para quaisquer aplicações com acesso a rede.

7.2 Trabalhos Futuros

Como trabalho futuro planejamos incrementar o GestureUI com novas características para descrição dos gestos, como movimentos com a cabeça, com o objetivo de expandir as possibilidades de palavras que podem ser reconhecidas pelos classificadores.

Pretendemos também avaliar a utilização do GestureUI na tarefa de reconhecimento de gestos dinâmicos através de um software de apoio ao ensino da Libras chamado LearnLibras, que está atualmente em sua fase inicial de modelagem e concepção. Este software tem por objetivo auxiliar professores no ensino da Libras para adultos ou crianças, enriquecendo o aprendizado da linguagem através de um inspetor virtual que por meio de estímulos audiovisuais (vídeos e áudios explicativos) possa auxiliar o aluno a executar o gesto de uma determinada palavra e saber se o mesmo foi efetuado de forma correta ou não.

Também é necessária a otimização do sistema de reconhecimento de gestos dinâmicos para que ao aumentarmos significativamente a quantidade de palavras da Libras a serem reconhecidas, o sistema continue passível de execução em tempo real. Para isso devemos: Paralelizar a avaliação dos modelos HMM; Otimizar a busca de centróides no K-Means e também avaliar novos métodos de clusterização para obtermos melhores resultados na representação das unidades básicas da Libras; e por fim otimizar os modelos HMM treinados desenvolvendo e avaliando técnicas de prunning dos estados.

REFERÊNCIAS

ALLEN, B.; CURLESS, B.; POPOVIC, Z. Articulated Body Deformation from Range Scan Data. ACM SIGGRAPH - Proceedings of the 29th annual conference on Computer graphics and interactive techniques, Pages 612-619, 2002.

ANDRILUKA, M.; ROTH, S.; SCHIELE, B. Pictorial Structures Revisited: People Detection and Articulated Pose Estimation. IEEE Computer Vision and Pattern Recognition, 2009.

ANJO, M. S. Reconhecimento de Gestos Estáticos de Mãos Utilizando a Rede Neural Neocognitron e a Multi-Layer Perceptron. Projeto de iniciação científica, Orientador: Prof. Dr. Ednaldo Brigante Pizzolato, Co-Orientador: José Hiroki Saito. Fundação de Amparo a Pesquisa do Estado de São Paulo, FAPESP, 2009.

ANJO, M. S.; PIZZOLATO, E.; FEUERSTACK, S.; A Real-Time System to Recognize Static Hand Gestures of Brazilian Sign Language (Libras) alphabet using Kinect. IHC - The 6th Latin American Conference on Human-Computer Interaction, 2012.

BAINBRIDGE-SMITH, A.; LANE, R. G. Determining Optical Flow Using a Differential Method. Image and Vision Computing, vol. 15, pp. 11-22, 1997.

BAUER, B.; HIENZ, H. Relevant Features for Video-Based Continuous Sign Language Recognition. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.

BEUCHER, S.; MEYER, F. The morphological approach to segmentation: the watershed transformation. In Mathematical Morphology in Image Processing, (Ed. E.R. Dougherty), p. 433-481, 1993.

BRADSKI, G.R. Computer video face tracking for use in a perceptual user interface. Intel Technology Journal, Q2, 1998.

BRASHEAR, H.; HENDERSON, V.; PARK, K.; HAMILTON, H.; LEE, S.; STARNER, T. American sign language recognition in game development for deaf children. 8th International ACM SIGACCESS Conference on Computers and Accessibility, 2006.

BRETZNER, L.; LAPTEV, I.; LINDEBERG, T.; LENMAN, S.; SUNDBLAD, Y.; A Prototype System for Computer Vision Based Human Computer Interaction. Technical Report, Department of Numerical Analysis and Computing Science KTH (Royal Institute of Technology), Stockholm, Sweden, 2001.

CANNY, J. A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-8 , Issue 6, Pages 679 – 698, 1986.

CHAI, D.; NGAN, K.N. Face segmentation using skin-color map in videophone applications. IEEE Trans. Circuits Syst. Video Technol., Vol. 9, Issue 4, Pages 551 – 564, 1999.

CHANG, F.; CHEN, C.; LU, C.A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique. ACM Journal Computer Vision and Image Understanding, Vol. 93, Issue 2, Pages 206 - 220, 2004.

CHEN, Q.; GEORGANAS, N. D.; PETRIU, E. M. Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar. IEEE Transactions On Instrumentation And Measurement, Vol. 57, No. 8, 2008.

CHEN, Y.; CHEN, C. Fast Human Detection Using a Novel Boosted Cascading Structure With Meta Stages. IEEE Transactions on Image Processing, Vol. 17, No. 8, 2008.

COLE, J. B.; GRIMES, D. B.; RAO, R. P. N. Learning Full-Body Motions from Monocular Vision: Dynamic Imitation in a Humanoid Robot. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007.

COOLEY, J. W.; TUKEY, J. W. An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation, Vol. 19, Pages 297-301, 1965.

CUCCHIARA, R.; GRANA, C.; PICCARDI, M.; PRATI, A. Detecting Moving Objects, Ghosts, and Shadows in Video Streams. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No.10, 2003.

DUDA, R. O.; HART, P. E.; STORK, D. G. Pattern Classification. Wiley-Interscience, Second Edition, 2001.

DUQUE, D.; SANTOS, H.; CORTEZ, P. Moving Object Detection Unaffected by Cast Shadows, Highlights and Ghosts. IEEE International Conference on Image Processing (ICIP), 2005.

ELMEZAIN, M.; AL-HAMADI, A.; MICHAELIS, B. Hand Gesture Recognition Based on Combined Features Extraction. World Academy of Science, Engineering and Technology, v. 60, 2009.

CHIANGKAI. Speech Recognition by Clustering Wavelet and PLP Coefficients. Master of Engineering in electrical Engineering and Computer Science Thesis, MIT, 1997.

Fang, Y.; Wang, K.; Cheng, J.; Lu, H.A Real-Time Hand Gesture Recognition Method. IEEE International Conference on Multimedia and Expo, 2007.

FELZENSZWALB, P. F.; HUTTENLOCHER, D. P. Pictorial Structures for Object Recognition. International Journal of Computer Vision, v. 61, p. 55–79, 2005.

FEUERSTACK, S.; OLIVEIRA, A.; ARAUJO, R. Model-based Design of Interactions that can bridge Realities – The Augmented Drag-and-Drop. 13th Symposium on Virtual and Augmented Reality (SVR 2011), ISSN 2177-676, Uberlândia, Minas Gerais, 23th-26th May, 2011.

FEUERSTACK, S.; ANJO, M. S.; PIZZOLATO, E.; Modellierung und Ausführung von multimodalen Anwendungen auf Basis von Zustandsdiagrammen. i-com, Zeitschrift für interaktive und kooperative Medien, Vol. 10 No.3, pp. 40-48, ISSN 1618-162, 2011.

FEUERSTACK, S.; ANJO, M. S.; PIZZOLATO, E.; Model-based Design, Generation and Evaluation of a Gesture-based User Interface Navigation Control. 5th Latin American Conference on Human-Computer Interaction (IHC 2011), Outubro, 2011.

FEUERSTACK, S.; ANJO, M. S.; COLNAGO, J.; PIZZOLATO, E.; Modeling of User Interfaces with State-Charts to Accelerate Test and Evaluation of different Gesture-based Multimodal Interactions. Accepted for Workshop “Modellbasierte Entwicklung von Benutzungsschnittstellen (MoBe2011)”, Informatik, 2011.

GONZALEZ, R. C.; WOODS, R. E. Digital Image Processing, 3ª Edição, Agosto 2007.

GRADY, L. Random Walks for Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1768-1783, Vol. 28, No. 11, 2006.

HAYKIN, S. Neural Networks: A Comprehensive Foundation. Prentice Hall, 2º ed., 1998.

HIRSCHMÜLLER, H. Improvements in real-time correlation based stereo vision. IEEE Workshop on Stereo and Multi-Baseline Vision (IJCV), 2001.

HONG, P.; TURK, M.; HUANG, T. S. Gesture Modeling and Recognition Using Finite State Machines. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.

HU, M. K. Visual Pattern Recognition by Moment Invariants. IRE Transactions on Information Theory, vol. 8, pp.179–187, 1962.

HU, C.; YU, Q.; LI, Y.; SONGDE, MA. Extraction of Parametric Human Model for Posture Recognition Using Genetic Algorithm. IEEE International Conference on Automatic Face and Gesture Recognition, 2000.

IVANOVS, J. Recognition of Human Poses Using Pictorial Structures. Masters Dissertation in Applied Computing Science, Universiteit Utrecht, Holanda, 2007.

JAIN, H. P.; SUBRAMANIAN, A. Real-time Upper-body Human Pose Estimation using a Depth Camera. HP Laboratories, 2010.

JOHNSON, J. L.; PADGETT, M. L. PCNN models and applications. IEEE Transactions on Neural Networks, v. 10-3 p. 480-498, 1999.

KAKUMANU, P.; MAKROGIANNIS, S.; BOURBAKIS, N.A survey of skin-color modeling and detection methods. Elsevier Journal of Pattern Recognition, 2007.

KARLIK, B.; OLGAC, A. V. Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks. International Journal of Artificial Intelligence and Expert Systems, Volume 1, Issue 4, pages 111-122, 2010.

KOHONEN, T. Self-Organizing Maps. Third extended edition, Springer, 2001.

LEE, J.Y.; YOO, S.I. An elliptical boundary model for skin color detection. Proceedings of the International Conference on Imaging Science, Systems and Technology, 2002.

LINDEBERG, T. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. International Journal of Computer Vision, v. 11(3), p. 283-318, 1993.

LLOYD, S. P. Least square quantization in PCM. IEEE Transactions on Information Theory, Vol. 28, Pages 129-137, 1982.

MASON, M.; DURIC, Z. Using Histograms to Detect and Track Objects in Color Video. Applied Imagery Pattern Recognition Workshop (AIPR), 2001.

MESTER, R.; FRANKE, U. Statistical Model Based Image Segmentation Using Region Growing, Countour Relaxation and Classification. Cambridge Symposium on Visual Communications and Image Processing, p. 616-624, November, 1988.

MILLER, A.; BASHARAT, A.; WHITE B.; LIU, J.; SHAH, M. Person and Vehicle Tracking in Surveillance Video. Multimodal Technologies for Perception of Humans, 2008.

MITTAL, A.; ZHAO, L.; DAVIS, L. S. Human Body Pose Estimation Using Silhouette Shape Analysis. IEEE Advanced Video and Signal Based Surveillance, 2003.

MOBAHI, H.; RAO, S.; YANG, A.; SASTRY, S.; MA, Y. Segmentation of Natural Images by Texture and Boundary Compression. Artificial Intelligence and Information Systems (AIIS) Seminar, 2010.

MODLER, P.; MYATT, T. Recognition of Separate Hand Gestures by Time-delay Neural Networks Based on Multi-state Spectral Image Patterns from Cyclic Hand Movements. IEEE International Conference on Systems, Man and Cybernetics, 2008.

MOESLUND, T. B.; GRANUM, E.A Survey of Computer Vision-Based Human Motion Capture. Elsevier, Computer Vision and Image Understanding, v. 81, p. 231-268, 2001.

MOESLUND, T. B.; HILTON, A.; KRUGER, V.A survey of advances in vision-based human motion capture and analysis. Elsevier, Computer Vision and Image Understanding, v. 104, p.90-126, 2006.

NGUYEN, D.; WIDROW, B. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. Proceedings of the International Joint Conference on Neural Networks, 3:21–26, 1990.

OSHER, S.; PARAGIOS, N. Geometric Level Set Methods in Imaging Vision and Graphics. Springer Verlag, 2003.

PAHLEVANZADEH, M.; VAFADOOST, M.; SHAHNAZI, M. Sign language recognition.9th International Symposium on Signal Processing and Its Applications, 2007.

PARKS, D. H.; FELS, S. S. Evaluation of Background Subtraction Algorithms with Post-processing. IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, 2008.

PEÑA, J. M.; LOZANO, J. A.; LARRANAGA, P. An empirical comparison of four initialization methods for the K-Means algorithm. Pattern Recognition Letters, v. 20, pp. 1027-1040, 1999.

PHU, J. J.; THAY, Y. H.; Computer Vision Based Hand Gesture Recognition Using Artificial Neural Network. Conference on Artificial Intelligence in Engineering and Technology, 2006.

PICCARDI, M. Background subtraction techniques: a review. IEEE International Conference on Systems, Man and Cybernetics, volume 4, p. 3099–3104, 2004.

PIZZOLATO, E. B.; ANJO, M. S.; PEDROSO, G. C. Automatic recognition of finger spelling for LIBRAS based on a two-layer architecture. ACM Symposium on Applied Computing, 2010.

PLAGEMANN, C.; GANAPATHI, V.; KOLLER, D.; THRUN, S. Real-time Identification and Localization of Body Parts from Depth Images.IEEE International Conference on Robotics and Automation (ICRA), 2010.

RABINER, L. R. A tutorial on hidden markov models and selected applications in speech recognition. Readings in speech recognition, p. 267 - 296, 1989.

ROSSI, A. R.; ROSSI, J. M. A. Curso de Libras Básico. Centro de Ensino, Pesquisa e Extensão e Atendimento em Educação Especial, Universidade Federal de Uberlândia, 2009.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning Internal Representations by Error Propagation. Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations. MIT Press, 1986.

SCHARSTEIN, D.; SZELISKI, R.A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. International Journal of Computer Vision, v. 47 i. 1-3, Junho 2002.

SEBE, N.; COHEN, T.; HUANG, T.S.; GEVERS, T. Skin detection, a Bayesian network approach. International Conference on Pattern Recognition (ICPR), 2004.

SHAPIRO, L. G.; STOCKMAN, G. C. Computer Vision. Prentice-Hal, New Jersey, pp 279-325, 2010.

STAUFFER, C.; GRIMSON, W. Adaptive background mixture models for real time tracking. IEEE Computer Vision and Pattern Recognition, 1999.

TAKAHASHI, K.; TANIGAWA, T. Remarks on Real-Time Human Posture Estimation from Silhouette Image Using Neural Network. IEEE International Conference on Systems, Man and Cybernetics, 2004.

THU, Q. H.; MEGURO, M.; KANEKO, M. Skin-color extraction in images with complex background and varying illumination, Sixth IEEE Workshop on Applications of Computer Vision, 2002.

TJANDRANEGARA, E. Distance Estimation Algorithm for Stereo Pair Images. Electrical and Computer Engineering, ECE Technical Reports, 2005.

TRIGO, T. R.; PELLEGRINO, S. R. M. An Analysis of Features for Hand-Gesture Classification. 17th International Conference on Systems, Signals and Image Processing (IWSSIP), 2010.

VAFADAR, M.; BEHRAD, A. Human hand gesture recognition using spatio-temporal volumes for human-computer interaction. International Symposium on Telecommunications, 2008.

VEZHNEVETS, V.; SAZONOV, V.; ANDREEVA, A. A Survey on Pixel-Based Skin Color Detection Techniques. PROC. GRAPHICON, 2003.

VIOLA, P.; JONES, M. Robust Real-time Object Detection. Second International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing, and Sampling, 2001.

VOGLER, C.; GOLDENSTEIN, S. Toward computational understanding of sign language. Technology and Disability, 20:2 pp. 109–119, 2008.

WANG, C.; GAO, W.; SHAN, S. An approach based on phonemes to large vocabulary Chinese sign language recognition. Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

WANG, Y.; YUAN, B. A novel approach for human face detection from color images under complex background. Elsevier Journal of Pattern Recognition, Vol. 34, Pages 1983 - 1992, 2001.

WREN, C.; AZARHAYEJANI, A.; DARRELL, T.; PENTLAND, A. P. Pfinder: real-time tracking of the human body. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, Pages 78-85, 1997.

WYSOSKI, S. G.; LAMAR, M. V.; KUROYANAGI, S.; IWATA, A.A Rotation Invariant Approach On Static-gesture Recognition Using Boundary Histograms and Neural Networks. Proceedings of the 4th International Conference on Neural Information Processing , Vol. 4, 2002.

YANG, A.Y.; IYENGAR, S.; SASTRY, S.; BAJCSY, R.; KURYLOSKI, P.; JAFARI, R. Distributed Segmentation and Classification of Human Actions Using a Wearable Motion Sensor Network. IEEE Computer Vision and Pattern Recognition Workshops, 2008.

YANG, Q.; YANG, R.; DAVIS, J.; NISTER, D. Spatial-Depth Super Resolution for Range Images. IEEE Conference on Computer Vision and Pattern Recognition, 2007.

ZHANG, D.; LU, G. A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures. Journal of Visual Communication and Image Representation, No. 14 (1), Pages 41-60, 2003.

ZHAO, L. Dressed Human Modeling, Detection, and Parts Localization. PhD Thesis, The Robotics Institute, Carnegie Mellon University, Julho, 2001.

ZHU, Y. Model-Based Human Pose Estimation with Spatio-Temporal Inferencing. PhD. Dissertation, Ohio State University, 2009.

ZHU, Y.; DARIUSH, B.; FUJIMURA, K. Controlled human pose estimation from depth image streams. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008.

ZHU, Y.; FUJIMURA, K. A Bayesian Framework for Human Body Pose Tracking from Depth Image Sequences. Honda Research Institute USA, Maio, 2010.

GLOSSÁRIO

Background Subtraction - Subtração de fundo ou remoção de fundo, técnica muito utilizada para dicotomizar a imagem em fundo e objetos de interesse.

Blob - Agrupamento de pontos (pixels) com características similares que representam um objeto na imagem.

Cluster - Agrupamento de pontos ou vetores que possuem menor distância ao centroide. É definido nas chamadas técnicas de clusterização como o K-Means.

Edge - Pontos localizados na borda dos objetos em uma imagem.

Exposure Time - Tempo de exposição à luz no digitalizador da câmera (CCD ou CMOS) para cada quadro capturado.

Features - Características dos objetos segmentados que são agrupadas para formar descritores que podem ser posteriormente utilizados para reconhecimento de padrões.

Frame - Quadro ou imagem capturada por uma câmera.

Ghosts - Termo utilizado em técnicas de Subtração de Fundo. A formação de ghosts (fantasmas) acontece quando em uma cena com o fundo já modelado um objeto que fazia parte do mesmo entra em movimento deixando para trás a silhueta do lugar que ocupava na cena anteriormente.

Highlight - Termo utilizado em técnicas de Subtração de fundo que significa mudanças repentinas de iluminação na cena em que o fundo está sendo modelado.

Histogram - Um histograma é uma representação gráfica da distribuição dos dados em classes ou intervalos de valores, que explicita a frequência que um valor foi observado.

Threshold - É um termo muito utilizado na área de Processamento de Imagens, Visão Computacional e Reconhecimento de Padrões que consiste na definição de um limiar que divide os valores em duas classes: as acima e as abaixo do limiar. O que será feito com essas duas classes depende do método sendo utilizado e da aplicação.

Tracking - Termo utilizado em Visão Computacional que significa rastrear a movimentação de objetos (blobs) previamente segmentados.

White Balance - Recurso presente em câmeras para nivelar o balanço dos canais de cores (azul, verde e vermelho) para garantir que o branco digitalizado se mantenha o mais próximo possível do branco real.