



Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pósgraduação em Engenharia da Computação

Pablo Vinicius Alves de Barros

Um Sistema Para Reconhecimento e Previsão de Gestos

Dissertação de Mestrado

Recife, Julho 2013



Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pósgraduação em Engenharia da Computação

Pablo Vinicius Alves de Barros

Um Sistema Para Reconhecimento e Previsão de Gestos

Dissertação de Mestrado

Dissertação apresentada no Programa
de Pósgraduação em Engenharia da
Computação como parte requerida
para a obtenção do título de
Mestre em Engenharia da Computação.

Orientador: Prof. Dr. Sérgio Murilo Maciel Fernandes

Recife, Julho de 2013

Dedicatória

Gostaria de dedicar esse trabalho em primeiro lugar aos meus pais, pois sempre estiveram presentes em todos os momentos da minha vida, sempre incentivando e alimentando a minha curiosidade e me fazendo ser o quem eu sou hoje. Devo tudo a eles. Agradeço também aos meus familiares por acompanharem a evolução deste trabalho e da minha vida acadêmica, mesmo à distância.

Gostaria de agradecer ao meu orientador Prof. Dr. Sérgio Murilo, bem como ao professor Msc. Bruno Fernandes, por estarem sempre acompanhando de perto, dando palpites, sugestões e as direções para que esse trabalho fosse realizado. Agradeço as orientações recebidas, tanto nesse trabalho como fora dele, pois culminaram nos vários acontecimentos que vieram a ocorrer durante a realização desse trabalho, e não foram poucos, bem como nos que irão acontecer a partir de agora.

Um agradecimento ao grupo de pesquisa RPPDI, em especial ao Prof. Byron Leite, por auxiliar em algumas orientações para esse trabalho. Agradeço efusivamente à Nestor Júnior e Juvenal Bisneto, que trabalharam lado a lado comigo para a realização desse trabalho. Todas as madrugadas, manhãs, tardes e noites escrevendo códigos, gravando gestos e torcendo para que tudo funcionasse acabaram rendendo este trabalho e todas as publicações obtidas com ele. E quem sabe o embrião para a SKYNET movida à quadricópteros voadores controlados pelo Kinect. Agradeço também aos membros do RPPDI, Felipe, Gearlles, Milla, Everalda entre outros, que sempre apoiaram no desenvolvimento desse trabalho, ajudando sempre que possível, principalmente nos dias do *cupcake*.

Agradeço à Universidade de Pernambuco, Escola Politécnica de Pernambuco, por toda a infra-estrutura que foi utilizada para o desenvolvimento deste trabalho, por todo o suporte e apoio nas apresentações dos artigos publicados, bem como em outros eventos acadêmicos relacionados, ou não, a este trabalho.

Agradeço ao corpo docente do curso de Pós Graduação em Engenharia da Computação - PPGE, da Universidade de Pernambuco, por todo o apoio acadêmico nas áreas afins com esse trabalho, através das aulas ministradas e conselhos dados. Agradeço também à ajuda da sempre prestativa Georgina, que me salvou em várias ocasiões durante todo o curso. Um salve aos alunos de mestrado da turma 2011.2, bem como das subsequentes, por tudo que passamos juntos nesse curso. E não foi pouca coisa. Sim, Probabilidade e Processos Estocásticos, estou falando de você.

Um agradecimento ao pessoal do *Knowledge Technology Group*, da Universidade de Hamburgo, por ter aceito a continuação deste trabalho em um PHD, a ser realizado na Alemanha, com início em outubro próximo. *Vielen Dank für die Gelegenheit.*

Um agradecimento aos amigo(a)s, irmãos(ãs) de uma vida toda, Márcio 'Winne' Alfredo, Márcio Lordelo, Francisco Júnior, Diego 'Thurran' George, Leandro Cavalcanti, Ilzy 'Yah' Sousa e Marcelo 'Mars' Maurício por ajudarem corrigindo e dando palpites em algumas sessões deste trabalho. Estejam prontos, que daqui a três anos vou precisar de vocês novamente. E aprendam alemão, pode vir a ser útil.

Epígrafe

Daqui para frente, estaremos deixando o terreno firme dos fatos para viajar juntos pelos turvos alagados da memória e nos embrenhar pelo matagal das suposições mais absurdas. (A. Dumbledore)

Recife, Junho 2013

Resumo

A área de reconhecimento de gestos vem sendo estudada há algumas décadas, sendo que nesta última vários avanços foram atingidos, devido à evolução das tecnologias e da capacidade de processamento dos computadores. Alguns desses avanços já permitem que a comunicação entre homens e máquina se dê através de gestos simples, utilizando imagens de um dedo ou de uma mão capturados por uma câmera de vídeo comum, ou até mesmo a captura de gestos realizados pelo corpo como um todo, por meio de dispositivos específicos para isso.

De forma a facilitar o uso de gestos como interação entre o homem e a máquina é utilizado visão computacional conforme vem sendo explorada com afincos pelos pesquisadores. O uso da visão computacional faz com que gestos possam ser capturados em ambientes reais, sem a necessidade de utilizar dispositivos específicos para a captura ou rastreamento dos mesmos. Através da captura por câmeras, um método não invasivo, pode-se utilizar esses sistemas em ambientes naturais, sem a necessidade de equipamentos de custo elevado ou de treinamento prévio pelo usuário.

Um dos problemas quando utiliza-se a visão computacional no reconhecimento de gestos, é a escolha não apenas de quais características serão extraídas, como também o tamanho desse vetor de características capturado. Esse problema acaba gerando alguns empecilhos para as técnicas utilizadas na classificação dos gestos por diminuir a eficácia e eficiência no reconhecimento.

Este trabalho de dissertação propõe um sistema para o reconhecimento e previsão de gestos dinâmicos realizados pela movimentação da mão de um usuário. Esse sistema utiliza uma técnica de extração de características do gesto que gera um vetor de características reduzido, contendo somente a mínima quantidade de pontos capaz de representar a mão. Esse processo reduz o tempo necessário para a classificação do gesto, além de aumentar a taxa de acerto, tornando o sistema mais eficiente e eficaz.

A técnica proposta é avaliada dentro do sistema descrito nesta dissertação e comparada com duas outras técnicas, *Local Contour Sequence* (LCS) e *Speed Up Robust Features* (SURF). Três técnicas de classificação são implementadas para os testes: *Dynamic Time Warping* (DTW), Modelo Oculto de Markov (HMM) e Rede Neural Recorrente de Elman (Elman RNN). São realizados três experimentos, um para a classificação de gestos estáticos, um para classificação de gestos dinâmicos e um outro para a classificação de gestos dinâmicos incompletos, simulando a previsão de gestos. Os resultados são apresentados e discutidos, detalhando as vantagens e desvantagens de cada técnica utilizada.

Palavras-Chaves: Reconhecimento de Gestos, Previsão de Gestos, Visão Computacional, Extração de Características.

Abstract

Several studies in gesture recognition have been developed in the last decades, reaching enormous advances through the technology evolution and the increase in the computer processing. Some of these advances, allow the communication between men and machine can be done by simple gesture captured by a video camera, executed by a hand or a finger. Even a full body gesture recognition can be done, using some specific devices.

The utilization of computer vision have been adopted by researchers with intent of facilitate gesture interaction. Computer vision allow a gesture to be recognized by a single video camera in natural environments, without the need of a specific device for capture or tracking. This non-invasive method permits the recognition without a high investment in hardware or a previous training by the user.

One of the problems using computer vision in gesture recognition, is the choose of which characteristics capture, or even, the size of these characteristics vector. This problem ends up generating some other problems for the classification techniques, such as the lost of efficiency and effectiveness in the recognition task.

This dissertation work proposes a dynamic gesture recognition and prediction system. This system uses a novel technique, proposed in this work also, for feature extraction applied in the hand of a user, that chooses the most effective points in the hand contour, and improves the recognition rate.

The proposed technique is evaluated with the proposed system, and their results are compared with other feature extraction techniques, Local Contour Sequence - LCS and Speed Up Robust Features - SUEF. Four classification techniques are implemented in the system: Dynamic Time Warping - DTW, Hidden Markov Model - HMM and Elman Recurrent Neural Network (Elman RNN). Three experiments are realized, one for classification of static gestures, the second for classification of dynamic gestures and the last one for classification of incomplete gestures, simulating a prediction.

The results are showed and discussed, detailing the good points and bad points of each technique.

KeyWords: Gesture Recognition, Gesture Prediction, Computer Vision, Features Extraction.

Sumário

Lista de Figuras	VIII
Lista de Tabelas	XI
Lista de Algoritmos	XII
Lista de Acrônimos	XIII
1 Introdução	14
1.1 Motivação e Objetivos	15
1.1.1 Objetivos Específicos	17
1.2 Estrutura deste Documento	17
2 Fundamentos em Reconhecimento de Gestos	18
2.1 Introdução	18
2.1.1 Organização do capítulo	20
2.1.2 Trabalhos Relacionados	20
2.2 Processamento Digital de Imagens - PDI	22
2.2.1 Extração de contorno	22
2.2.2 Algoritmo da extração do contorno	24
2.3 Extração de características	25
2.3.1 <i>Local Contour Sequence - LCS</i>	26
2.3.2 Algoritmo LCS	28
2.3.3 <i>Speed Up Robust Features Extraction - SURF</i>	31
2.3.4 Algoritmo SURF	35
2.4 Reconhecimento de Padrões	36
2.4.1 Modelos Ocultos de Markov - HMM	36
2.4.2 Algoritmo HMM	42
2.4.3 <i>Dynamic Time Warping - DTW</i>	42
2.4.4 Algoritmo DTW	44
2.4.5 Rede Neural Recorrente de Elman - Elman RNN	45
2.4.6 Algoritmo Elman RNN	50
3 Reconhecimento de Gestos Dinâmicos Através da Abordagem Convexa	52
3.1 Introdução	52

3.1.1	Organização do capítulo	53
3.2	Abordagem Convexa	54
3.2.1	Minimização do modelo geométrico	55
3.2.2	Extração dos pontos mais significativos	56
3.2.3	Composição do vetor de características	58
3.2.4	Algoritmo da Abordagem Convexa	60
4	Experimentos e Resultados	61
4.1	Introdução	61
4.1.1	Organização do capítulo	61
4.2	Base de dados de gestos dinâmicos RPPDI	62
4.3	Experimentos	63
4.3.1	Experimento 1: Reconhecimento de gestos estáticos	63
4.3.2	Experimento 2: Reconhecimento de gestos dinâmicos	66
4.3.3	Experimento 3: Previsão de gestos dinâmicos	74
5	Conclusão e Trabalhos Futuros	81
5.1	Conclusão	81
5.2	Contribuições	82
5.3	Trabalhos Futuros	83

Lista de Figuras

2.1	Resultado da aplicação do <i>Otsu Threshold</i> em uma imagem de postura de mão. (a) exhibe a imagem original e (b) a saída do <i>Otsu Threshold</i> . (c) exhibe o resultado final do processo de segmentação.	22
2.2	Processo de binarização de uma imagem em tons de cinza utilizando K como <i>threshold</i>	23
2.3	Ilustração do cálculo da distância do LCS em uma imagem contendo o contorno de uma mão.	26
2.4	Ilustração dos vizinhos do pixel p_0	27
2.5	Vetor de características que descreve a Figura 2.1(c) obtido através da aplicação do LCS.	28
2.6	Imagem exemplo utilizada como entrada no algoritmo LCS.	29
2.7	Imagem exemplo utilizada como entrada no algoritmo LCS, após o processo de ordenação horária.	30
2.8	(a) Exemplo do cálculo da distância a partir do <i>pixel</i> P0. (b) Exemplo do cálculo da distância a partir do <i>pixel</i> P1.	30
2.9	O cálculo para soma de intensidades de uma dada região retangular, delimitada pelos <i>pixels</i> A, B, C e D, é dado por $(A + C) - (B + D)$	31
2.10	(a) Derivada Gaussiana de segunda ordem aplicadas em y ($L_{yy}(x, \sigma)$), (b) Derivada Gaussiana de segunda ordem aplicadas em xy ($L_{cy}(x, \sigma)$) e (c) Derivada Gaussiana de segunda ordem aplicadas em x ($L_{xx}(x, \sigma)$)	32
2.11	Resultado da aplicação do filtro Laplaciano Gaussiana nas imagens da Figura 2.10.	33
2.12	Relação entre <i>Octave</i> e as escalas de aplicação dos filtros.	33
2.13	Transformada de <i>Haar</i> aplicada em ambas as direções, (a) x e (b) y	34
2.14	Extração dos pontos de interesse e orientação aplicadas à Figura 2.1(a).	35
2.15	Representação gráfica de um HMM contendo três estados (x_1, x_2 e x_3) e os observáveis que podem ser emitidos por eles (y_1, y_2 e y_3).	37
2.16	Duas sequências com tamanhos diferentes sendo alinhadas utilizando o DTW.	43
2.17	Matriz de custos C para duas sequências.	44
2.18	Representação gráfica de um peceptron, onde X_1, X_2, X_p representam as entradas, W_1, W_2, W_p representam os pesos, \sum representa a função soma e $f(a)$ representa o limiar de ativação. Fonte: [Hay99]	46

2.19	Representação gráfica de uma MLP contendo as camadas de entrada, uma camada escondida e a camada de saída.	47
2.20	Representação gráfica de uma Elman RNN contendo as camadas de entrada, escondida, de saída e de contexto.	49
3.1	Arquitetura que descreve o sistema de reconhecimento de gestos proposto.	53
3.2	Fluxograma que descreve as etapas da Abordagem Convexa, método de extração de características para reconhecimento de gestos.	54
3.3	Execução do algoritmo Douglas-Peucker. (a) exhibe o primeiro passo, delimitando o corredor da distância $2T$. O segundo passo (b) seleciona o ponto mais distante do segmento de linha formados por âncora e flutuante e cria dois subgrupos, onde o algoritmo será executado novamente.	55
3.4	Resultado da aplicação do algoritmo de minimização de polígonos Douglas-Peucker na imagem original (a). (b) exhibe a saída do algoritmo, com o modelo minimizado, contendo menos curvas e mais vértices.	56
3.5	A aplicação do algoritmo de Sklansky em um polígono minimizado da imagem de uma mão, gera uma representação que enaltece os vértices externos(a). Já a aplicação do realce nos vértices internos, gera um modelo mais detalhado (b).	58
3.6	(a)Resultado da composição do vetor de características para a postura de mão exibida na Figura 3.5(b). Já (b) exhibe o mesmo vetor de características normalizado com 10 elementos.	59
4.1	Ilustração dos sete gestos dinâmicos contidos na base de gestos dinâmicos RPPDI.	62
4.2	Ilustração das oito posturas de gestos estáticos extraídos da base de gestos dinâmicos RPPDI.	64
4.3	Gráfico exibindo os resultados de classificação para todas as técnicas testadas.	72
4.4	Gráficos exibindo os tempos de treinamento e classificação para todas as combinações de técnicas.	73
4.5	Gráfico exibindo a evolução da predição de gestos corretos para todas as combinações testadas.	79
4.6	Gráficos de comparação do tempo de treinamento e classificação para todas as combinações de técnicas.	80

Lista de Tabelas

4.1	Quantidade de sequências gravadas para cada gesto da base de gestos dinâmicos RPPDI	62
4.2	Quantidade de sequências presentes para cada classe de posturas no experimento Reconhecimento de gestos estáticos.	64
4.3	Configurações utilizadas para cada par de algoritmos	64
4.4	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o LCS no módulo de extração e o DTW no módulo de classificação	65
4.5	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o SURF no módulo de extração e o DTW no módulo de classificação	65
4.6	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CSURF no módulo de extração e o DTW no módulo de classificação	65
4.7	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CLCS no módulo de extração e o DTW no módulo de classificação	66
4.8	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o LCS no módulo de extração e a MLP no módulo de classificação	66
4.9	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o SURF no módulo de extração e a MLP no módulo de classificação	66
4.10	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CSURF no módulo de extração e a MLP no módulo de classificação	67
4.11	Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CLCS no módulo de extração e a MLP no módulo de classificação	67
4.12	Configuração de cada par de técnica de extração e classificação	68
4.13	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o HMM no módulo de classificação	68

4.14	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e o HMM no módulo de classificação	68
4.15	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e o HMM no módulo de classificação	69
4.16	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e o HMM no módulo de classificação	69
4.17	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o DTW no módulo de classificação	69
4.18	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e o DTW no módulo de classificação	70
4.19	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e o DTW no módulo de classificação	70
4.20	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e o DTW no módulo de classificação	70
4.21	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o Elman RNN no módulo de classificação	71
4.22	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e a Elman RNN no módulo de classificação	71
4.23	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e a Elman RNN no módulo de classificação	71
4.24	Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e a Elman RNN no módulo de classificação	72
4.25	Resultado da previsão adicionando-se os quatro primeiro <i>frames</i> no vetor de características para o CLCS.	74
4.26	Resultado da previsão adicionando-se os <i>frames</i> cinco à oito no vetor de características para o CLCS.	75
4.27	Resultado da previsão adicionando-se os <i>frames</i> de nove à doze no vetor de características para o CLCS.	75
4.28	Resultado da previsão adicionando-se os <i>frames</i> treze e catorze no vetor de características para o CLCS.	75
4.29	Resultado da previsão adicionando-se os quatro primeiro <i>frames</i> no vetor de características para o CSURF.	76
4.30	Resultado da previsão adicionando-se os <i>frames</i> cinco à oito no vetor de características para o CSURF.	76

4.31	Resultado da previsão adicionando-se os <i>frames</i> nove à doze no vetor de características para o CSURF.	76
4.32	Resultado da previsão adicionando-se os <i>frames</i> treze e catorze no vetor de características para o CSURF.	77
4.33	Resultado da previsão adicionando-se os quatro primeiro <i>frames</i> no vetor de características para o LCS.	77
4.34	Resultado da previsão adicionando-se os <i>frames</i> cinco à oito no vetor de características para o LCS.	78
4.35	Resultado da previsão adicionando-se os <i>frames</i> nove à doze no vetor de características para o LCS.	78
4.36	Resultado da previsão adicionando-se os <i>frames</i> treze e catorze no vetor de características para o LCS.	78

Lista de Algoritmos

Algoritmo 1: Algoritmo para segmentação de uma mão.....	25
Algoritmo 2: Algoritmo para detecção de borda.....	26
Algoritmo 3: Algoritmo LCS	29
Algoritmo 4: Algoritmo SURF	36
Algoritmo 5: Algoritmo para Treino de um modelo HMM.....	42
Algoritmo 6: Algoritmo para a classificação utilizando HMM.....	43
Algoritmo 7: Algoritmo DTW	45
Algoritmo 8: Algoritmo para treino uma Elman RNN..	51
Algoritmo 9: Algoritmo para classificação em uma Elman RNN....	51
Algoritmo 10: Algoritmo para fecho convexo de um polígono.....	57
Algoritmo 11: Algoritmo Abordagem Convexa.....	57

Lista de Acrônimos

<i>LIBRAS</i> : Língua Brasileira de Sinais	22
<i>PDI</i> : Processamento Digital de Imagens	27
<i>LCS</i> : <i>Local Contour Sequence</i>	27
<i>SURF</i> : <i>Speed Up Robust Features</i>	27
<i>HMM</i> : Modelo Oculto de Markov	37
<i>DTW</i> : <i>Dynamic Time Warping</i>	37
<i>MLP</i> : <i>Multilayer Perceptron</i>	37
<i>ElmanRNN</i> : Rede Neural Recorrente de Elman	37

Capítulo 1

Introdução

Sistemas de reconhecimento de gestos proporcionam uma forma natural e amigável de interação com sistemas computacionais, que se tornam mais naturais para seres humanos. A área de reconhecimento de gestos busca identificar padrões de movimentos em seres humanos, aprendê-los e generalizá-los de forma que seja possível reconhecê-los ao serem executados por qualquer pessoa. Existem várias aplicações nesse campo de estudo como aplicações em jogos eletrônicos [LH10] [RA11], interação entre homens e robôs [BDH13] [Lee06], interação com aparelhos televisores [JJS⁺12] e reconhecimento de linguagem de sinais [CH11] [ZCZ⁺08] [SA07].

Hoje em dia, utilizando somente uma câmera de vídeo em um telefone celular, *tablet* ou *notebook* é possível capturar gestos realizados por seres humanos. Uma vez que a maioria das pessoas têm a sua disposição pelo menos um desses aparelhos, a utilização de uma interação baseada em gestos se torna ainda mais natural e necessária. A evolução dos computadores também auxilia nesse processo, já que mesmo aparelhos celulares dispõem de poder computacional suficiente para executar essa tarefa, podendo ser aplicadas técnicas baseadas em visão computacional nos próprios aparelhos.

Sistemas para reconhecimento de gestos dinâmicos podem ser separados em três categorias [IK12]: Sistemas que utilizam luvas ou sensores externos ligados ao corpo do usuário para captar o gesto executado [HMWA11] [XGDT10] [JLK11] [KTK09] [Han10]; Sistemas que reconhecem um gesto através do desenho gerado por um hardware, como um mouse, ou parte do corpo humano [SCO11] [JJS⁺12] [CR11] [COB⁺04]; Sistemas que capturam gestos através de câmeras para o reconhecimento de gestos complexos [ZZ08] [SCH⁺05] [BNT09] [LBM01] [KK10].

A primeira abordagem tende à captação mais precisa dos gestos executados porém se torna invasiva ao usuário já que ele utiliza um hardware muitas vezes incômodo para a captura. A aplicação de luvas de captura se dá em um ambiente controlado, onde as luvas são conectadas a um computador, tornando complicada sua utilização em um ambiente externo como por exemplo uma aplicação no mundo real.

A segunda abordagem é feita com um mouse ou mesmo através de uma câmera, fazendo o rastreamento do usuário como um todo ou de algum membro em específico. Essa abordagem utiliza um conceito simplificado de gesto o que torna mais fácil o

reconhecimento, porém os gestos que podem ser expressados por essa abordagem são menos significativos e menos precisos do que os representados pelas outras abordagens. Normalmente utiliza-se um dispositivo para capturar o desenho do gesto realizado pelo usuário, e a partir desse desenho, o gesto é reconhecido.

A última categoria utiliza câmeras para capturar o gesto e identificá-lo através de vários fatores: movimentação, posicionamento, velocidade, cor, observações biométricas entre outros. Essa abordagem é a que contempla um conceito de gesto mais amplo e é menos incômodo ao usuário, já que sua captura se dá através de uma câmera de vídeo, seja ela de um *smartphone*, *notebook* ou mesmo uma câmera de segurança. Essa abordagem utiliza técnicas de visão computacional para identificar o gesto que está sendo realizado e classificá-lo através das imagens obtidas na câmera de vídeo.

Uma característica desejada pelos sistemas de reconhecimento de gestos é a capacidade de reconhecimento em tempo real aplicadas em situações reais. Alguns trabalhos apresentam soluções para esses problemas, mas como debatido por Mori et al. [MUK⁺06], um sistema para reconhecimento de gestos de atuação em tempo real é capaz de prever o gesto que está sendo executado antes do seu fim, de forma que o sistema obtenha respostas em tempo hábil, facilitando o processamento dos dados. Essa capacidade é denominada previsão de gestos, onde um gesto parcial é reconhecido com sucesso.

A predição melhora o reconhecimento, tentando identificar um padrão antes que ele seja executado por completo. Técnicas de predição vêm sendo utilizadas com sucesso para melhorar o reconhecimento de voz [ST07] [HJAJG09] [VH99] [SGS11] [HN07]. Poucos trabalhos utilizam o conceito de previsão de gestos para o reconhecimento de gestos dinâmicos, como descrito por Kohlsdorf et al. [KSA11].

1.1 Motivação e Objetivos

A utilização de sistemas para reconhecimento de gestos como forma de interação natural já vem sendo explorada, mas alguns aspectos ainda precisam ser otimizados. A diminuição do tempo de processamento para o reconhecimento pode ser realizada, tanto com a captura de um vetor de características menor, como com a utilização de previsão de gestos, de forma que um gesto incompleto, ainda sendo realizado, possa ser reconhecido. Outro problema é a falta de uma base de gestos dinâmicos que possa ser utilizada para o teste de algoritmos de extração de características. A maioria das bases de gestos dinâmicas existentes estão mais focadas na segmentação da mão do que no reconhecimento do gesto, necessitando uma abordagem mais focada no processo de segmentação. Essas bases tendem a tirar o foco da utilização dos algoritmos de extração de características baseados no contorno [HK12].

Várias técnicas são aplicadas na extração de características para o reconhecimento de gestos, porém muitas delas são afetadas por um problema denominado maldição da dimensionalidade, como exibido no trabalho publicado por [BA-ESS11]. A maldição da dimensionalidade diz que uma aproximação numérica de

uma função irá ter um custo computacional maior de acordo com o crescimento de variáveis utilizadas por ela [KE11]. Existem algumas propostas para solução da maldição da dimensionalidade, como a redução de dimensões no vetor de características [ZMP10] [PKF00], otimização dos algoritmos de classificação [QM06] [QT09] e utilização de estratégias para a seleção de características otimizadas para o problema [TS07].

Esta dissertação propõe um método de diminuição do vetor de características para o reconhecimento de gestos dinâmicos. Esse método se baseia na seleção das características que tornem o processo de classificação mais eficaz e eficiente, a partir da escolha de uma quantidade mínima de pontos que represente uma mão. Esse método é uma proposta de resolução da maldição da dimensionalidade para o reconhecimento de gestos dinâmicos.

Um sistema para o reconhecimento de gestos dinâmicos é proposto nesse trabalho de dissertação. De forma a resolver alguns dos problemas citados, uma técnica de extração de características de gestos dinâmicos realizados com a mão é proposta. Essa técnica utiliza como entrada o contorno de uma mão, extraído de uma imagem. De forma a obter um resultado otimizado, essa técnica é aplicada em outras duas técnicas de extração, o *Local Contour Sequence* (LCS) e o *Speed Up Robust Feature* (SURF). O LCS e o SURF conseguem extrair o contorno da mão através de métodos diferentes, aplicados à imagem uma imagem binária. O LCS utiliza uma série de técnicas de processamento de imagens para extrair o contorno da mão e aplica um cálculo de distância entre os pontos do contorno, formando o vetor de características que descreve a mão. O SURF utiliza de uma série de operações matemáticas para extrair os pontos de interesse em uma imagem, representando o contorno da mão, e então extrai um vetor de característica que descreve esses pontos.

A técnica proposta nesse trabalho, denominada Abordagem Convexa, utiliza a mesma ideia de extração de um vetor de características a partir dos pontos do contorno da mão presentes no LCS. Ao invés de extrair as características utilizando o contorno completo extraído pelo LCS, a Abordagem Convexa procura minimizar o contorno, selecionando os pontos que mais eficientemente conseguem representar a mão. A aplicação da Abordagem Convexa no LCS é denominada CLCS. A Abordagem Convexa também é aplicada nos pontos de interesse extraídos pelo SURF, gerando o CSURF. Essas duas novas técnicas, o CLCS e o CSURF, são utilizadas dentro do sistema de reconhecimento de gestos dinâmicos proposto por esse trabalho. Os resultados obtidos são comparados com resultados obtidos utilizando o LCS e o SURF.

De forma a testar o sistema proposto e a Abordagem Convexa aplicada ao LCS e o SURF, vários testes são executados. Além do teste para o reconhecimento de gestos dinâmicos, citado acima, o reconhecimento de gestos estáticos e a previsão de gestos por extrapolação também são avaliados. O sistema implementa três técnicas de classificação diferentes, de forma que os resultados possam ser comparados com a combinação de todas as técnicas. As três técnicas implementadas são: Modelos Ocultos de Markov (HMM), Rede Neural Recorrente de Elman (Elman RNN) e um classificador baseado na comparação de sequências através do *Dynamic Time*

Warping (DTW).

Esse trabalho também apresenta uma nova base de gestos dinâmicos contendo sete gestos realizados pela mão de um usuário. Essa base possui a característica de facilitar o teste dos algoritmos de extração, visto que ela é desenvolvida com um fundo estático e contém gestos com várias posturas de mãos diferentes.

1.1.1 Objetivos Específicos

- Propor uma técnica de extração de características para gestos baseados na mão de um usuário, que pode ser utilizada em conjunto com o LCS e o SURF. Essa técnica gera um vetor de características minimizado, otimizando o processo de classificação;
- Avaliar o sistema de reconhecimento de gestos com a utilização das quatro técnicas de extração de características citadas acima, LCS, SURF, CLCS e CSURF. A combinação das quatro técnicas de extração de características com cada uma das técnicas de classificação, DTW, HMM e Elman RNN, são testadas;
- Propor uma base de gestos dinâmicos de mão.

1.2 Estrutura deste Documento

Para um melhor entendimento desta dissertação, é indicada uma leitura sequencial. Esta dissertação está estruturada da seguinte forma:

- Capítulo 1: apresenta a motivação e objetivos deste trabalho;
- Capítulo 2: descreve o estado da arte em reconhecimento de gestos e trabalhos relacionados ao tema;
- Capítulo 3: descreve o novo método para extração de características e o sistema para reconhecimento de gestos proposto;
- Capítulo 4: apresenta a base de gestos desenvolvida, exhibe a metodologia aplicada nos testes do sistema proposto e os resultados obtidos;
- Capítulo 5: descreve as contribuições do trabalho proposto, a conclusão e trabalhos futuros.

Capítulo 2

Fundamentos em Reconhecimento de Gestos

2.1 Introdução

Um gesto pode ser executado por um ou mais seres humanos, através da movimentação de braços, mãos, dedos, cabeça, pernas ou mesmo o corpo inteiro. Pode expressar informações, sentimentos, representar conceitos ou ser o meio de comunicação entre várias pessoas ou até mesmo pessoas e máquinas. Gestos também podem ter conceitos diferentes dependendo do contexto onde se encontram, e a união de vários gestos pode ter significados diferentes dos encontrados quando separados. Gestos podem ser classificados de acordo com sua execução em relação ao tempo [Gav99]:

- Estáticos: uma certa posição específica de mãos, braços, pernas ou cabeça representa o gesto. Este cenário não necessita nenhuma alteração do gesto para que ele seja executado. O usuário precisa permanecer na posição por um tempo determinado de forma que o gesto possa ser interpretado;
- Dinâmicos: o gesto é o resultado de uma sequência de gestos estáticos e os movimentos necessários para realizá-los. Um gesto dinâmico leva em consideração, além da posição específica dos membros, a sua movimentação. Este cenário é mais complexo pois cada pessoa pode realizar a movimentação do gesto em formas e velocidades diferentes.

Outra classificação de gestos pode ser em relação aos seus usuários [AC99]:

- Monusuário: gestos monousuários são executados por um ser humano. Nesta modalidade, um gesto é realizado por um só ser humano utilizando o seu próprio corpo. É a modalidade que está presente na maioria das aplicações que utilizam gestos, como nas línguas de sinais, e interação homem máquina;

- **Multiusuário:** nessa modalidade a conjunção de movimentos realizados por dois ou mais seres humanos, ou mesmo de seres humanos e objetos inanimados representam um gesto. Um gesto multiusuário é mais complexo e é totalmente dependente de contexto, já que sua utilização está mais presente em jogos eletrônicos ou interpretação de ações em contextos informais, como por exemplo um filho perguntando à sua mãe onde guardar um pacote segurando-o na mão e apontando para um armário.

Uma das aplicações mais complexas envolvendo gestos são as Línguas de Sinais. Cada Língua de Sinais é considerada um idioma à parte, com suas próprias regras gramaticais, sintáticas e semânticas. As línguas de sinais espalhadas pelo globo se baseiam em gestos dinâmicos e monousuários. Cada sinal nas Línguas de Sinais podem ser representados por um ou mais gestos e possuem um ou vários significados, dependendo do contexto.

No Brasil a Língua de Sinais utilizada é denominada Língua Brasileira de Sinais – LIBRAS. Em LIBRAS um sinal é composto por cinco características [FAL10]:

- **Postura:** A postura das mãos e braços durante a realização do gesto. A postura pode ser alterada durante a execução do gesto, sendo observados também os estados de transição entre posturas.
- **Movimentação:** A movimentação das mãos e braços é observada. A movimentação pode ser horizontal, vertical ou diagonal.
- **Orientação:** A orientação da movimentação é observada. A orientação pode ser qualquer uma presente na rosa dos ventos somada com as orientações diagonais [LNV⁺12].
- **Posição espacial:** Descreve a posição dos braços e mão em relação ao corpo do usuário. Vários gestos possuem significados diferentes de acordo com a distância entre os braços e o corpo.
- **Expressão facial:** A expressão facial na hora da realização do gesto também é observada, sendo este um aspecto muito importante pois define o aspecto biométrico da realização do gesto, além dos aspectos espaço-temporais citados anteriormente.

Cada sinal de Libras é composto por variações dessas cinco características, podendo dois ou mais sinais conterem as mesmas posições, movimentações ou mesmo expressões faciais, executadas em momentos diferentes. Sinais iguais podem ter significados diferentes dependendo do contexto, ou apresentarem uma variação mínima em uma das cinco configurações.

O sistema proposto nesta dissertação utiliza visão computacional para o reconhecimento de gestos dinâmicos. Esses gestos abrangem três das cinco características de um sinal de LIBRAS: posição, movimentação e orientação.

2.1.1 Organização do capítulo

Uma leitura sequencial de todas as sessões deste capítulo é indicada para um melhor entendimento. Este capítulo está estruturado da seguinte forma:

- Sessão 2.1 - Introdução: apresenta uma análise introdutória sobre os sistemas para reconhecimento de gestos;
- Sessão 2.2 - Trabalhos Relacionados: apresenta uma revisão sobre os trabalhos relacionados ao tema proposto;
- Sessão 2.3 - Processamento Digital de Imagens: Apresenta uma revisão das técnicas de processamento de imagens digitais utilizadas nesta dissertação;
- Sessão 2.4 - Extração de características: Apresenta uma revisão das técnicas de extração de características utilizadas nesta dissertação;
- Sessão 2.5 - Reconhecimento de padrões: Apresenta uma revisão das técnicas de reconhecimento de padrões utilizadas nesta dissertação.

2.1.2 Trabalhos Relacionados

A maioria dos sistemas que utilizam visão computacional para o reconhecimento de gestos dinâmicos [HK12] são divididos em três módulos: pré-processamento da imagem, extração ou aquisição de características e aprendizagem ou classificação dos gestos. O módulo de pré-processamento contém todas as tarefas de suavização, segmentação e remoção de ruídos na imagem. A tarefa de extração de características se mostra a mais desafiadora pois envolve vários elementos com características dinâmicas e complexas como iluminação, *background* complexos e oclusão. Para resolver a maior parte desses problemas, técnicas de processamento digital de imagens são aplicadas e limitações na captura ou no reconhecimento são impostas. Cada problema requer uma técnica ou conjunto de técnicas a serem aplicadas e a escolha dessas técnicas deve ser feita com cuidado levando em consideração o tipo de gesto, a quantidade de usuários e o ambiente de captura dos gestos, por exemplo. Uma das limitações mais comuns é a observação de somente um membro do corpo humano, e como a mão é o membro que consegue caracterizar uma quantidade maior de gestos, ela é comumente escolhida [PSH97].

Um dos problemas mais comuns no reconhecimento de gestos baseados em visão computacional é a extração de características das imagens para um reconhecimento correto. Várias abordagens já foram aplicadas para isso, como no trabalho de [RMM⁺09] que reconhece um gesto através da posição dos dedos. Cada dedo aberto é reconhecido a partir do contorno da mão. Nesse trabalho os pontos concentrados no centro da mão são eliminados e os que restam são marcados como dedos abertos. Para cada dedo aberto, sua ponta é encontrada através da análise dos pontos mais distantes naquele segmento de pontos. Após encontrar as pontas dos dedos abertos em uma mão, eles são comparados com uma base de dados contendo os gestos conhecidos pelo sistema. A distância entre o gesto a ser

classificado e os gestos já conhecidos é calculada utilizando uma soma da distância de cada ponto. Esse trabalho é aplicado ao reconhecimento de gestos estáticos provenientes da Língua Americana de Sinais obtendo uma taxa de 95% dos gestos classificados corretamente.

No trabalho de Fang et al. [FWCL07] o gesto é reconhecido através da detecção da palma da mão e da orientação dos dedos em relação à palma. Ele utiliza o algoritmo de reconhecimento *Binary Large Objects* (BLOBS) desenvolvido por Lindberg [Lin98] para o reconhecimento da palma da mão e dos dedos em uma imagem com *background* complexo. Utilizando uma base de dados com seis gestos dinâmicos uma média de 90% dos gestos são reconhecidos corretamente.

Stergiopoulou & Papamarkos [SP09] utilizam uma análise da topografia de Redes Neurais Alto Organizadas para o reconhecimento de gestos estáticos realizados com a mão de um usuário. Uma *Self-Growind and Self-Organized Neural Gas* (SGONG) é utilizada. O contorno de uma mão é colocado na rede e a topografia que a rede atinge ao final de sua execução é utilizada como extração de características para aquele gesto. O sistema foi testado no reconhecimento de 31 gestos estáticos e obteve uma taxa de 31% de gestos reconhecidos corretamente.

Murakami & Taguchi [MT91] utilizam uma rede recorrente de Elman para o reconhecimento de posturas de mão que representam 42 símbolos da língua japonesa de sinais. Maraqa & Abu-Zaiter [MAZ08] utilizam uma rede neural recorrente para o reconhecimento de gestos presentes na Língua Árabe de Sinais. Já Chun et al. [SHL⁺98] utiliza uma rede neural recorrente para o reconhecimento de gestos baseados em *templates* espaciais, em que os gestos devem ser realizados em uma área pré-definida de frente à câmera de vídeo.

Ahmad & Seong-Whan [AL06] utilizam o contorno de um ser humano para descrever posições corporais e as classifica utilizando um HMM. Abdul et al. [ATK11] desenvolveram um sistema que utiliza um HMM para cada gesto presente na Língua Malasiana de Sinais, podendo reconhecer os gestos realizados pela mão do usuário.

Duas técnicas utilizadas nesta dissertação são o *Local Contour Sequence* (LCS), descrita por [GM01] e o *Speed Up Robust Features Extraction* (SURF) [BETVG08]. Essas técnicas foram escolhidas por serem técnicas relativamente recentes e que vem sendo utilizadas na área de reconhecimento de gestos dinâmicos e estáticos. Elas apresentam bons resultados e devido à sua natureza de funcionamento, podem ser aplicadas ao reconhecimento de gestos realizados com mãos, como demonstrado no trabalho de Meena [Mee11], que descreve um sistema de reconhecimento de gestos estáticos através dos pontos extraídos do contorno da mão. Meena utiliza a técnica *Local Contour Sequence* (LCS), que extrai distâncias do contorno da mão, para gerar um vetor de características que representa o gesto completo, independente de qual posição a mão possa estar. Esse vetor de características é classificado por uma *Support Vector Machine* (SVM), e um total de 7 gestos são reconhecidos atingindo uma taxa de 90% de gestos classificados corretamente. Já no trabalho de Jiang et al. [JSFJ11] é descrito a utilização dos pontos de interesse extraídos pelo SURF para o reconhecimento de gestos dinâmicos extraídos de um vídeo. Yao et al. [YY12] utiliza os pontos extraídos pelo SURF para reconhecer posturas de

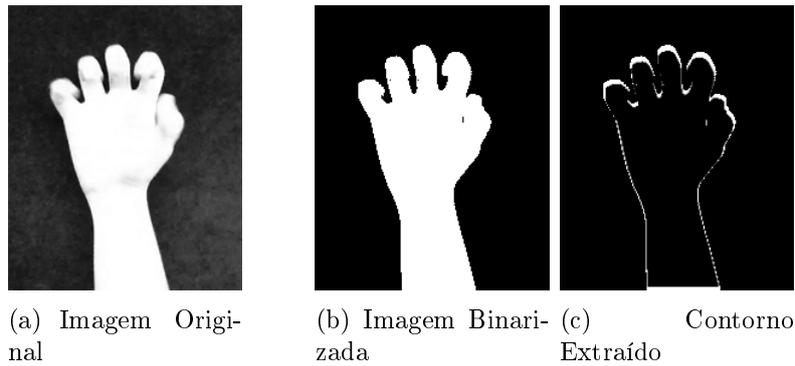


Figura 2.1: Resultado da aplicação do *Otsu Threshold* em uma imagem de postura de mão. (a) exibe a imagem original e (b) a saída do *Otsu Threshold*. (c) exibe o resultado final do processo de segmentação.

mão estáticas.

2.2 Processamento Digital de Imagens - PDI

Por Processamento Digital de Imagens (PDI) entende-se a manipulação de uma imagem por computador de modo que a entrada e a saída do processo sejam imagens. O objetivo de se usar processamento digital de imagens é melhorar o aspecto visual de certas feições estruturais para o analista humano e fornecer outros subsídios para a sua interpretação, inclusive gerando produtos que possam ser posteriormente submetidos a outros processamentos [RG10].

Quando aplicadas ao reconhecimento de gestos, técnicas de PDI ajudam a realçar o formato da mão na imagem, de forma que possa ser descrita por alguma das técnicas de extração de características. Para a maioria das técnicas de extração de características de gestos dinâmicos, como as abordadas neste trabalho, encontrar o contorno da mão é uma tarefa primordial.

2.2.1 Extração de contorno

O processo de extração de contorno de uma mão começa com a segmentação da imagem pela sua separação em dois elementos: objeto e fundo. Para o reconhecimento de gestos realizados por uma mão, a mão é considerada o objeto a ser reconhecido e tudo que não tiver relação com a mão tem que ser eliminado da imagem, sendo classificado como fundo. A Figura 2.1 exibe a imagem após aplicação de cada passo do processo de extração de contorno.

Uma das técnicas mais conhecidas para a segmentação de imagens é o *Otsu Threshold* [Ots79]. Essa técnica consegue segmentar uma imagem, originalmente em tons de cinza, transformando-a em uma imagem binária. Uma imagem binária é aquela cujos *pixels* possuem apenas dois valores, 0 ou 1. A imagem binária por possuir um tamanho menor do que uma imagem comum, diminui o custo computacional necessário para realizar operações, como extração de características.

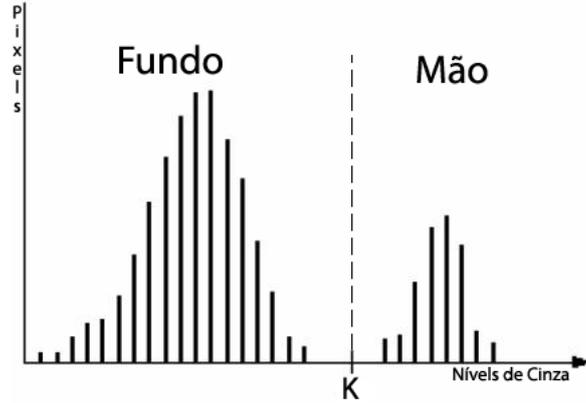


Figura 2.2: Processo de binarização de uma imagem em tons de cinza utilizando K como *threshold*.

O propósito do *Otsu* é encontrar um *threshold* K , que separe uma imagem em níveis de cinza nas duas classes: fundo e objeto, neste caso, a mão. A Figura 2.2 exibe uma ilustração do uso do *threshold* K no processo de binarização da imagem. O *Otsu* inicia-se calculando a quantidade de *pixels* η_i para cada tom de cinza i e denomina-se η como o total de *pixels* na imagem, ou seja, $\eta = \eta_1 + \eta_2 + \eta_3 + \eta_4 \dots + \eta_L$. A distribuição de probabilidade para o nível i é dado por

$$\rho_i = \frac{\eta_i}{\eta}. \quad (2.1)$$

Então, calcula-se a probabilidade $\omega_0(K)$, que define a probabilidade do intervalo de níveis de cinza $[0, K]$ pertencer a classe fundo, e $\omega_1(K)$, que define a probabilidade do intervalo $[K + 1, L]$ pertencer a classe mão, onde L é o último nível de cinza encontrado na imagem. Essas probabilidades são definidas por:

$$\omega_0(K) = \sum_{i=0}^K \rho_i. \quad (2.2)$$

$$\omega_1(K) = \sum_{i=K+1}^L \rho_i = 1 - \omega_0. \quad (2.3)$$

O próximo passo é o cálculo do valor esperado dos níveis de cinza de cada classe, μ_0 para a classe fundo e μ_1 para a classe mão, que serão utilizadas para o cálculo da variância de cada classe. A média das probabilidades é dada por:

$$\mu_0(K) = \frac{\sum_{i=1}^K \rho_i i}{\omega_0(K)}; \quad (2.4)$$

$$\mu_1(K) = \frac{\sum_{i=K+1}^L \rho_i i}{\omega_1(K)}; \quad (2.5)$$

O próximo passo é o cálculo da variância de cada classe, denominada $\sigma_0^2(K)$ para a classe fundo e $\sigma_1^2(K)$ para a classe mão:

$$\sigma_0^2(K) = \frac{\sum_{i=1}^K (\rho_i - \mu_0)^2}{\omega_0(K)} \quad (2.6)$$

$$\sigma_1^2(K) = \frac{\sum_{i=K+1}^L (\rho_i - \mu_1)^2}{\omega_1(K)} \quad (2.7)$$

Por fim a variância entre as duas classes, definida por $\sigma_w^2(K)$, é calculada

$$\sigma_w^2(K) = \omega_0(K)\sigma_0^2(K) + \omega_1(K)\sigma_1^2(K). \quad (2.8)$$

A variância entre as duas classes é calculada para todos os valores de K , ou seja, para todos os níveis de cinza encontrados na imagem. O valor de K que obtiver o maior $\sigma_w^2(K)$ é então escolhido para ser o *threshold* da imagem, e todos os *pixels* pertencentes ao intervalo $[0, K]$ são marcados para ser removidos da imagem, recebendo um valor 0. Já todos os *pixels* no intervalo $[K + 1, L]$ são marcados com o valor 1. A Figura 2.1(b) exibe o resultado de binarização de imagem utilizando o *Otsu*.

Após o processo de binarização, um segundo procedimento é executado, utilizando a imagem binarizada como entrada. Esse processo tem como objetivo eliminar todos os *pixels* que não são utilizados para a determinação da postura da mão. Em outras palavras, esse processo encontra o contorno da mão, reduzindo a quantidade de informação a ser processada pelos algoritmos de extração, otimizando todo o processo de reconhecimento.

Um dos processos mais simples e eficazes para encontrar bordas em uma imagem binária é a análise de diferença de intensidades dos *pixels*. Essa análise permite eliminar todos os *pixels* que não possuem vizinhos com variações 1-0, deixando somente os *pixels* que representam a borda da mão. Inicia-se varrendo a imagem binária à procura do primeiro pixel com valor 1. Ao encontra-lo, verifica-se se ele possui algum dos 8 vizinhos com valor 0, caso positivo marca-se esse pixel como borda. Caso negativo marca-se esse pixel como não-borda. Ao final excluem-se todos os *pixels* marcados como não borda. A Figura 2.1(c) exibe o resultado desse processo.

2.2.2 Algoritmo da extração do contorno

O processo de segmentação da mão em uma imagem consiste na junção dos dois métodos citados: a binarização da imagem com o *Otsu* e a delimitação da borda da imagem binária. O Algoritmo 1 exibe o pseudocódigo que demonstra o *Otsu*. O *Otsu* possui um custo computacional na ordem de $O(N \log N)$. Primeiro encontra-se o histograma da imagem em tons de cinza e calcula-se a probabilidade de cada tom de cinza presente no histograma, entre linhas 5 e 10. O *threshold* a ser aplicado no histograma é encontrado maximizando a variância entre cada um dos tons de cinza, como descrito no *Otsu Threshold*, exemplificado entre as linhas 12 e 20.

Algorithm 1: Algoritmo para segmentação de uma mão.

```
1 Histograma = Histograma(Imagem);
2 Computar a probabilidade  $\rho_i$  para cada nível de cinza de Histograma;
3 VarianciaMaxima = -1;
4 MaximoThreshold = 0;
5 for Nível de cinza K em Histograma do
6   |   variancia =  $\sigma_w^2(K)$ ;
7   |   if variancia > VarianciaMaxima then
8   |     |   VarianciaMaxima = variancia;
9   |     |   MaximoThreshold = K;
10  |   end
11 end
12 for  $y = 0 ; y < ImagemOriginal.Altura; y++$  do
13   |   for  $x = 0 ; x < ImagemOriginal.Largura; x++$  do
14   |     |   if Imagem.pixel(x,y) está em Histograma[0,MaximoThreshold] then
15   |     |     |   Imagem.pixel(x,y) = 0;
16   |     |     end
17   |     |   else
18   |     |     |   Imagem.pixel(x,y) = 1;
19   |     |     end
20   |   end
21 end
22 return Imagem;
```

A imagem binária é utilizada como entrada para a segunda parte do algoritmo, a detecção de borda, demonstrado aqui pelo Algoritmo 2, que possui uma complexidade de $O(N^2)$. Varre-se a imagem e procura-se por uma variação 1-0 entre os *pixels*, ou seja, procura-se por *pixels* que tenham valor 1 e possuam vizinhos 0. Isso é demonstrado nas linhas 2 a 7.

Após isso, uma nova imagem é criada e os *pixels* marcados como borda são adicionados à ela, linhas 9 e 10.

2.3 Extração de características

O processo de extração de características tem como função primária representar a imagem em um espaço de medidas que possa ser utilizado por um classificador. No problema de reconhecimento de gestos feitos pelas mãos de um usuário, as técnicas de extração devem ser capazes de conseguir identificar e descrever a posição e postura da mão, independente de usuário, escala e rotação. As seções a seguir explicam as técnicas de extração de características: LCS e SURF.

Algorithm 2: Algoritmo para segmentação de uma mão

```
1 List<Pixel> Bordas;  
2 for  $y = 0 ; y < Imagem.Altura; y++$  do  
3   for  $x = 0 ; x < Imagem.Largura; x++$  do  
4     if  $Imagem.pixel(x,y)$  for 1 e possuir algum vizinho que seja 0 then  
5       | Bordas.add(Imagem.pixel(x,y));  
6     end  
7   end  
8 end  
9 Imagem Segmentada;  
10 for  $Pixel p$  em Bordas do  
11 | Segmentada.pixel(p) = 1;  
12 end  
13 return Segmentada.
```

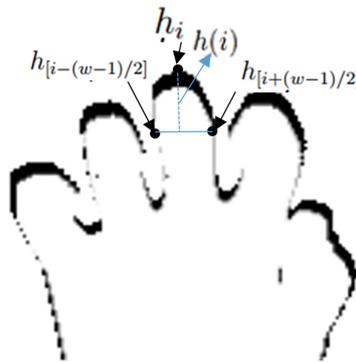


Figura 2.3: Ilustração do cálculo da distância do LCS em uma imagem contendo o contorno de uma mão.

2.3.1 Local Contour Sequence - LCS

O LCS é uma técnica de descrição de gestos baseada no contorno de uma mão, que extrai um vetor de números que representa a posição que a mão está assumindo naquela imagem. O algoritmo recebe como entrada a imagem binária contendo o contorno de uma mão. O LCS encontra o primeiro *pixel* que faz parte do contorno, varrendo-se a imagem de cima para baixo e da esquerda para a direita. A seguir, utilizando esse pixel como referência, todos os demais *pixels* da imagem são ordenados em ordem horária.

Para ordenar os *pixels* em sentido horário a partir do primeiro *pixel*, um algoritmo de ordenação baseado em vizinhança é utilizado. Neste algoritmo, dado que o primeiro *pixel* encontrado é representado pelas coordenadas (x,y) e marcado como p_0 , verifica-se se os *pixels* dos seus vizinhos norte($x,y+1$), nordeste($x+1,y+1$), leste($x+1,y$), sudeste($x+1,y-1$), sul($x,y-1$), sudoeste($x-1,y-1$), oeste($x-1,y$) ou noroeste($x-1,y+1$) são 1. A Figura 2.0001 ilustra os vizinhos do pixel p_0 . A verificação é feita nessa ordem, e o primeiro a ser encontrado é marcado como

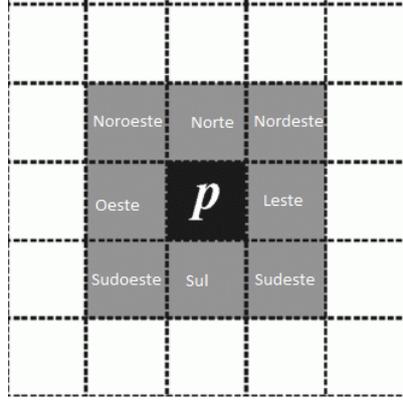


Figura 2.4: Ilustração dos vizinhos do pixel p_0 .

p_1 e o algoritmo é interrompido. Utilizando p_1 como referência, procede-se a verificação de vizinhança novamente, da mesma maneira, seguindo a mesma ordem, e encontra-se p_2 . Esse algoritmo é repetido até que todos os pontos do contorno sejam percorridos, nunca adicionando o mesmo ponto duas vezes.

Após a obtenção de todos os *pixels* ordenados, tem-se h_i como o pixel na posição i , ou seja $h_i = (x_i, y_i)$, de um total de N *pixels*. O LCS retorna um vetor de distâncias que representa a postura da mão. Cada distância é calculada entre o pixel h_i e uma reta formada por $h_{[i-(w-1)/2]}$ e $h_{[i+(w-1)/2]}$, onde w é um parâmetro a ser passado para o algoritmo. A Figura 2.4 exibe a ilustração do cálculo da distância para um ponto $h(i)$. O valor de cada distância, $h(i)$, é calculada através da seguinte equação

$$h(i) = \left| \frac{\mu_i}{v_i} \right|, \quad (2.9)$$

Sendo μ_i e v_i definidos por:

$$\begin{aligned} \mu_i = & x_i [y_{i-(w-1)/2} - y_{i+(w-1)/2}] \\ & + y_i [x_{i+(w-1)/2} - x_{i-(w-1)/2}] \\ & + [y_{i+(w-1)/2}] [x_{i+(w-1)/2}] \\ & - [y_{i-(w-1)/2}] [x_{i-(w-1)/2}], \end{aligned} \quad (2.10)$$

$$v_i = \left[(y_{i-(w-1)/2} - y_{i+(w-1)/2})^2 + (x_{i-(w-1)/2} - x_{i+(w-1)/2})^2 \right]^{1/2}. \quad (2.11)$$

Para todos os pontos presentes no contorno, $i=1,2,3,4,5\dots N$, $h(i)$ é calculado e anexado ao vetor de características. Como vários gestos podem ser executados em diferentes distâncias da câmera, uma padronização da amplitude dos valores extraídos se faz necessária, minimizando o fator escala. Para tanto, divide-se cada valor do vetor de características pelo desvio padrão do mesmo. A Figura 2.5 exibe

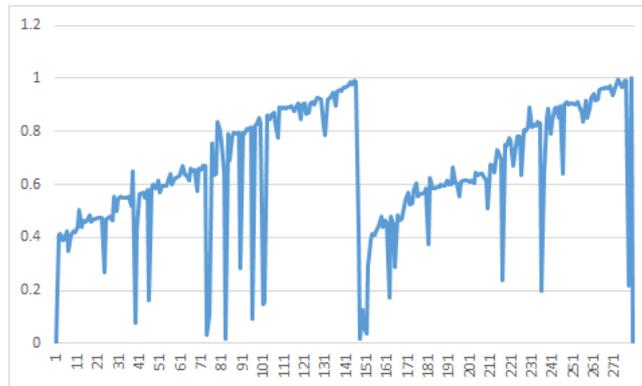


Figura 2.5: Vetor de características que descreve a Figura 2.1(c) obtido através da aplicação do LCS.

o LCS da postura de mão presente na Figura 2.1(c), onde foram obtidos 271 pontos normalizados.

O LCS possui algumas características que auxiliam na representação dos gestos de mão. Como citado anteriormente, através de uma padronização, o LCS se torna invariante à escala, ou seja, o gesto pode ser realizado a distâncias diferentes da câmera, respeitando sempre um limite definido pela resolução da imagem, obtendo-se uma descrição parecida. Respeitando sempre um limite estabelecido pela resolução da câmera. O algoritmo também é invariante a translação, pois o ponto inicial obtido na ordenação pode variar dependendo da posição da mão, mas as distâncias serão as mesmas. A escolha da janela w pode afetar diretamente na construção do vetor de características. Um valor muito alto de w , cria uma amplitude maior para as distâncias, o que aumenta as chances de captação de ruído no contorno. Já a diminuição da janela w pode gerar muitas características, aumentando a quantidade de distâncias extraídas.

Durante o desenvolvimento deste trabalho, um problema observado no LCS é a quantidade de pontos obtidos por cada mão, que pode ser elevado. Esse problema pode fazer com que o LCS sofra da maldição da dimensionalidade, aumentando o tempo e poder computacional requeridos para a classificação do gesto, além de diminuir a taxa de reconhecimento.

2.3.2 Algoritmo LCS

O algoritmo do LCS, Algoritmo 3, é um algoritmo de execução sequencial e não realiza nenhuma computação complexa possuindo uma complexidade $O(N \log N)$. O Algoritmo 3 exibe o pseudocódigo do LCS. O primeiro passo é a ordenação dos pontos em sentido horário, como exibido na linha 1. Uma vez que os pontos já estejam ordenados o cálculo do LCS é realizado, o valor da janela w é escolhido, e a distância para cada ponto é calculada e armazenada em um vetor, como exibido entre as linhas 3 e 7. Por fim, esse vetor é padronizado, dividindo-se cada elemento do vetor pelo seu desvio padrão, como exibido na linha 8.

Algorithm 3: Algoritmo para extração de características da imagem utilizando LCS

```

1 List<Pixel> PontosOrdenados = OrdenacaoHoraria(Imagem);
2 Double LCS;
3 Escolhe-se o valor de w;
4 for Pixel p em PontosOrdenados do
5   | distancia = h(i);
6   | LCS.add(distancia);
7 end
8 LCS = PadronizarLCS(LCS);
9 return LCS.
```

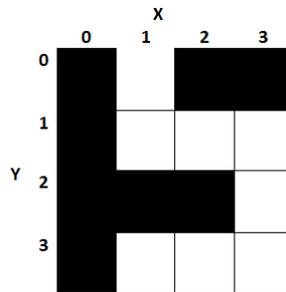


Figura 2.6: Imagem exemplo utilizada como entrada no algoritmo LCS.

Para melhor entendimento do algoritmo do LCS, utilize como entrada a imagem descrita na Figura 2.6. O primeiro passo do LCS é encontrar o *pixel* mais acima na imagem, neste caso o *pixel* (1,0) é o escolhido, sendo assim denominado *pixel* P0. A partir do *pixel* P0, os demais são ordenados em sentido horário.

A ordenação em sentido horário, iniciado a partir do *pixel* P0, inicia-se verificando se o vizinho norte possui um *pixel* com valor 1. Como não existe nenhum, o vizinho nordeste é verificado, depois o leste, sudeste e por fim o sul. Ao visitar o vizinho sul, encontra-se um *pixel* com valor 1, o *pixel* (1,1). Esse *pixel* é então denominado P1. O algoritmo é iniciado novamente, agora centrado em P1. Ao final do algoritmo, ou seja, ao ele retornar à P0 ou não encontrar nenhum vizinho, a ordenação ficará a seguinte: P0(0,1), P1(1,1), P2(2,1), P3(3,1), P4(3,2), P5(3,3), P6(3,2), P7(3,1). A Figura 2.7 exibe a ordenação final dos pontos.

Após a ordenação, uma janela w tem que ser definida. Neste exemplo o valor de w é definido como 4. O cálculo da distância do LCS, $h(i)$ é realizado traçando-se uma reta entre cada *pixel* P_i e uma reta formada por $h_{[i-(w-1)/2]}$ e $h_{[i+(w-1)/2]}$. No exemplo, para cada *pixel* será calculada uma distância. Para o primeiro *pixel*, P0, a distância será calculada em relação à reta que passa entre os *pixels* P6($h_{[0-(4-1)/2]} = -2$) e P2($h_{[0+(4-1)/2]} = 2$). A segunda distância será calculada a partir do ponto P1, em relação à reta formada por P7 e P3. Todas as distâncias são calculadas e adicionadas a um vetor de características. A Figura 2.8 ilustra o cálculo da distância para os dois primeiros *pixels*, P0 e P1.

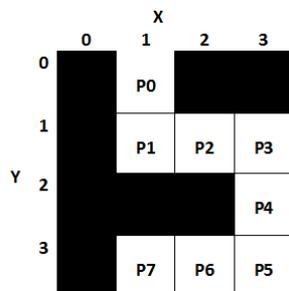
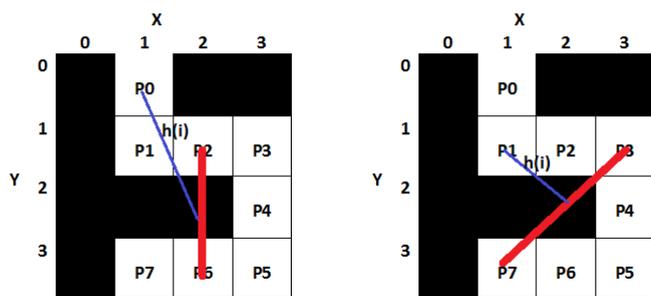


Figura 2.7: Imagem exemplo utilizada como entrada no algoritmo LCS, após o processo de ordenação horária.



(a) Exemplo do cálculo da distância a partir do *pixel* P0.

(b) Exemplo do cálculo da distância a partir do *pixel* P1.

Figura 2.8: (a) Exemplo do cálculo da distância a partir do *pixel* P0. (b) Exemplo do cálculo da distância a partir do *pixel* P1.

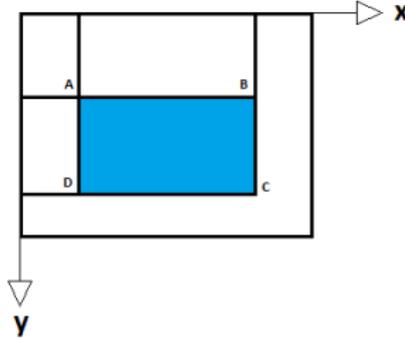


Figura 2.9: O cálculo para soma de intensidades de uma dada região retangular, delimitada pelos *pixels* A, B, C e D, é dado por $(A + C) - (B + D)$.

2.3.3 Speed Up Robust Features Extraction - SURF

O SURF é um algoritmo para descrição de imagens que identifica pontos de interesse e extrai descritores desses pontos, que podem ser utilizados para classificar objetos. Nesta dissertação o SURF é utilizado para classificar gestos realizados com uma mão.

Para encontrar os pontos de interesse em uma imagem, isto é, os pontos dos quais serão obtidos os descritores, o SURF utiliza uma abordagem baseada em uma aproximação da matriz Hessiana [CXC⁺11]. Através do uso de imagens integrais [Cro84], o SURF consegue diminuir o custo computacional do algoritmo, resultando em uma execução rápida de todo o processo de extração dos pontos de interesse.

Uma imagem integral, representa a soma dos *pixels* de uma imagem I com seus vizinhos acima e a esquerda [VJ01]. A equação a seguir ajuda a definir a imagem integral $I_{\Sigma}(X)$

$$I_{\Sigma}(x, y) = \sum I(x - 1, y - 1). \quad (2.12)$$

Uma vez que a imagem integral foi computada, o cálculo da soma das intensidades de uma dada região retangular pode ser facilmente calculada através da operação: $A + C - (B + D)$, onde A , B , C e D representam o valor dos *pixels* dos cantos da região retangular, como exibida na Figura 2.9.

A detecção dos pontos de interesse do SURF está baseada na matriz Hessiana, através da busca por estruturas parecidas, ou seja, nas quais o determinante é máximo. Dado um ponto $\chi = (x, y)$ em uma imagem I , a matriz Hessiana $\chi(x, \sigma)$ em x e com uma escala σ é definida por:

$$\chi(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \quad (2.13)$$

Onde $L_{xx}(x, \sigma)$ é a convolução da derivada Gaussiana de segunda ordem $\frac{\partial^2}{\partial x^2}g(\sigma)$ com a imagem I em x . Da mesma forma descobre-se $L_{xy}(x, \sigma)$ e $L_{yy}(x, \sigma)$. A Figura 2.9(a) exhibe a derivada gaussiana de segunda ordem em y ($L_{yy}(x, \sigma)$). Já a

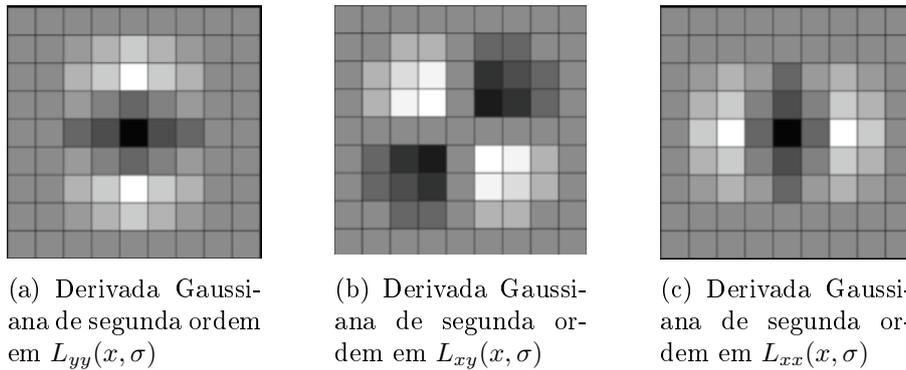


Figura 2.10: (a) Derivada Gaussiana de segunda ordem aplicadas em y ($L_{yy}(x, \sigma)$), (b) Derivada Gaussiana de segunda ordem aplicadas em xy ($L_{xy}(x, \sigma)$) e (c) Derivada Gaussiana de segunda ordem aplicadas em x ($L_{xx}(x, \sigma)$)

Figura 2.10(b) exibe a derivada gaussiana de segunda ordem em xy ($L_{xy}(x, \sigma)$) e a Figura 2.10(c) exibe a derivada da gaussiana de segunda ordem em x ($L_{xx}(x, \sigma)$). Para todas as imagens da figura 2.10, uma escala σ de 1.2 foi utilizada.

Já que a aplicação do filtro Hessiano original não é ideal para aplicações computacionais, um filtro Laplaciano da Gaussiana [Low04] é utilizado, de forma a transformar o resultado obtido pela aplicação da matriz Hessiana em valores computacionais. Um filtro 9×9 normalmente é utilizado, pois adequa-se bem a maioria dos casos. Denomina-se D_{xx} a aplicação do filtro Laplaciano da Gaussiana na Derivada Gaussiana de segunda ordem $L_{xx}(x, \sigma)$. Da mesma forma, encontra-se D_{xy} e D_{yy} da imagem. O resultado da aplicação do filtro Laplaciano da Gaussiana nas imagens da Figura 2.9 pode ser visto nas imagens da Figura 2.11.

Para encontrar as regiões de interesse em um dado ponto x , o SURF utiliza o determinante da matriz Hessiana para o ponto x . Os pontos com determinantes próximos são mapeados em uma região similar, delimitando aquela como uma região de interesse. O determinante da matriz Hessiana pode ser calculado utilizando a aproximação de D_{xx} , D_{xy} e D_{yy} , dado por

$$\det(\chi) = D_{xx}D_{yy} - (0.9D_{xy})^2. \quad (2.14)$$

As regiões de interesse encontradas precisam ser invariantes à escala, de forma que imagens com tamanhos diferentes possam ser comparadas e para tanto um espaço de escala é implementado. Espaços de escala são utilizados geralmente como uma pirâmide de imagens. A cada nível da pirâmide, a imagem é borrada com um filtro Gaussiano e então redimensionada. Como o SURF trabalha com imagens integrais, o custo computacional de aplicar filtros maiores na mesma imagem é menor do que redimensionar toda a imagem, fazendo com que em cada nível da pirâmide seja aplicado um filtro gaussiano em escala maior. No trabalho de Lowe [Low04], ele aplica uma Diferença de Gaussianas em todos os níveis da pirâmide, de forma a extrair contornos e regiões comuns a todas elas.

Para extração dos pontos de interesse, dentro da região de interesse, o espaço de escala é dividido em *Octaves*. Cada *Octave* representa um mapeamento das

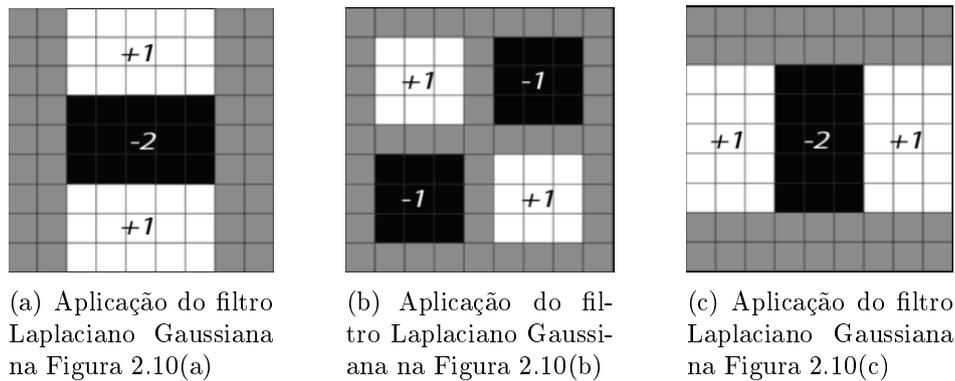


Figura 2.11: Resultado da aplicação do filtro Laplaciano Gaussiana nas imagens da Figura 2.10.

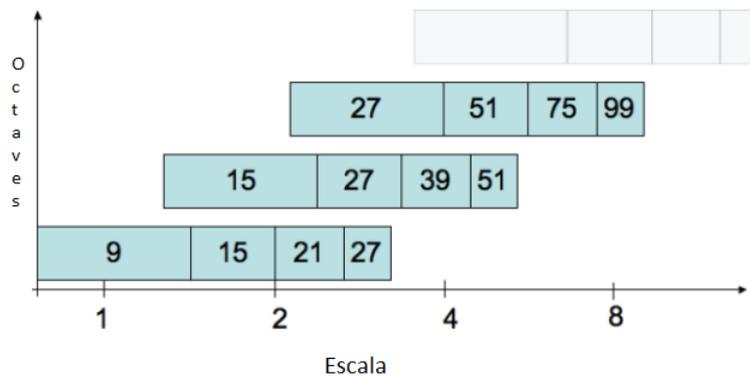


Figura 2.12: Relação entre *Octave* e as escalas de aplicação dos filtros.

respostas obtidas pela convolução da imagem com um filtro, que varia em escala. No total, uma *Octave* contém o mapeamento das respostas de um filtro com escala dobrada, ou seja, o filtro vai crescendo até atingir uma escala que possui o dobro do tamanho original. A Figura 2.12 exhibe a relação entre as *Octave* e as escalas para aplicação dos filtros em cada uma delas.

A imagem original é aplicada dentro do espaço de escalas, através do filtro com escalas diferentes em cada *Octave*. Isso cria uma pirâmide de imagens, contendo a imagem original na base e todas as suas derivadas nos níveis mais acima. A partir das áreas de regiões encontradas em cada *Octave*, uma supressão não-máxima de tamanho 3×3 é aplicada em cada escala de cada *Octave*. Esse processo resulta na extração de intensidades díspares, realçando as regiões de transição da imagem. Para cada escala aplicada, a supressão não-máxima também é aplicada para as regiões acima e abaixo, realçando as áreas de interesse nas diversas escalas. Após essa operação, os pontos de interesse ($\hat{\chi}$) são interpolados utilizando a matriz Hessiana escrita no formato de uma expansão de uma série de Taylor centrada no ponto X :

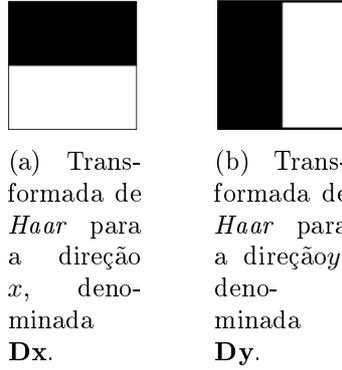


Figura 2.13: Transformada de *Haar* aplicada em ambas as direções, (a) x e (b) y

$$\hat{\chi} = -\frac{\vartheta^2 H^{-1} \vartheta H}{\vartheta X^2} \frac{\vartheta H}{\vartheta X} \quad (2.15)$$

Após extrair os pontos de interesse, o SURF obtém descritores para esse ponto. De forma que esses descritores sejam invariantes à rotação, o SURF extrai a orientação de cada ponto. Para tanto a transformada de *Haar* é calculada para as direções x e y para os vizinhos em um raio de $6s$, onde s delimita a escala em que os pontos foram encontrados. Mais uma vez o uso de imagens integrais auxiliam na aplicação da transformada de *Haar*, agilizando o seu cálculo. A Figura 2.13 exhibe a transformada de *Haar* para as direções x e y , respectivamente D_x e D_y .

Com o resultado da aplicação da transformada em todos os pontos vizinhos, separam-se os valores de D_x e D_y em um plano cartesiano e multiplica-se cada valor pela Gaussiana ($\sigma = 2,5s$), centrada no ponto de interesse. A orientação é calculada através da soma de todas as respostas que pertencem a uma janela, de tamanho previamente informado. Essa janela é deslizada, centrada no ponto de interesse, a uma taxa de $\frac{\pi}{3}$ radianos, e a orientação calculada novamente. A orientação dominante é escolhida pelo valor que obtiver maior soma das coordenadas.

Para extrair a descrição de cada ponto de interesse, o primeiro passo é construir uma região quadrada, de tamanho $20s$, centrada no ponto de interesse e com a orientação baseada na orientação do ponto de interesse. Essa região é dividida em pequenas sub-regiões de tamanho 5×5 . A transformada de *Haar* de cada pixel de cada sub-região é calculada, para ambas as direções, x e y , sempre orientadas pelo ponto de interesse. Para cada conjunto de 4×4 um vetor de descrição com quatro dimensões é extraído. Esse vetor é composto pela soma do valor da transformada de *Haar* de cada pixel da região, em ambas as direções, e a soma do valor absoluto para cada dimensão. Isso é necessário para trazer informações sobre mudança de intensidades na vizinhança do ponto de interesse. O vetor de descritores para cada sub-região, denominado v , é definido como

$$v = \left(\sum Dx, \sum Dy, \sum |Dx|, \sum |Dy| \right). \quad (2.16)$$

Para todas as sub-regiões presentes na região, um vetor contendo 64 caracte-

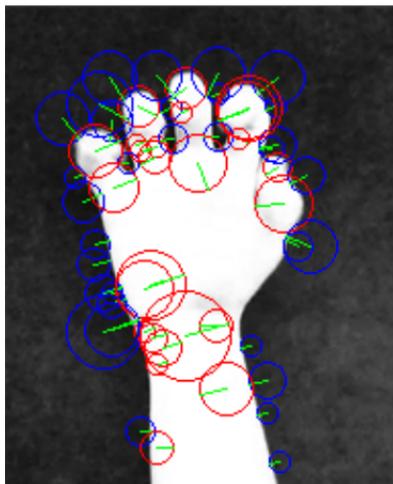


Figura 2.14: Extração dos pontos de interesse e orientação aplicadas à Figura 2.1(a).

rísticas é extraído, já que a cada subregião, com tamanho 5×5 , um vetor com 4 características é extraído. Esse vetor contendo 64 elementos é o descritor do ponto de interesse. A Figura 2.14 exhibe a aplicação da extração de pontos de interesse e a orientação desses pontos aplicados à imagem da Figura 2.1(a).

Para o reconhecimento de gestos dinâmicos, assim como LCS, o SURF também sofre da maldição da dimensionalidade. A quantidade de características extraídas é muito elevada. Se para cada ponto de interesse são extraídas 64 características, uma imagem como a Figura 2.14, que contém 57 pontos, conteria um total de 3648 características. No reconhecimento de gestos dinâmicos, um gesto é formado por uma sequência de imagens. Então, um gesto contendo 15 imagens com posturas diferentes precisaria de 54.720 características para representá-lo. Esse valor se mostrou de grande magnitude para todos os classificadores implementados nessa dissertação, diminuindo a taxa de reconhecimento e aumentando o tempo de processamento.

2.3.4 Algoritmo SURF

O algoritmo do SURF, Algoritmo 4, que possui uma complexidade computacional $O(N^2)$, começa com a transformação da imagem em uma imagem integral, como exibido na linha 1. Depois disso, as *Octaves* são extraídas, através da aplicação de várias convoluções entre a matriz Hessiana e a imagem integral, exemplificado na linha 2. Depois de obtidas as *Octaves*, os pontos de interesse são extraídos, através da aplicação de supressões não-máximas em um espaço de escalas formado por variações da imagem original aplicadas à um filtro Gaussiano com escala variante. Os pontos são interpolados utilizando uma expansão da série de Taylor centrada no ponto X . Essa operação está exemplificada nas linhas 4 à 6.

A orientação dos pontos de interesse é obtida através da soma da aplicação de uma série de transformadas de *Haar* dos eixos x e y nos vizinhos, pertencentes

a uma determinada janela, do ponto de interesse. Ao rotacionar a janela, várias somas são obtidas e a soma que tiver o maior valor é escolhida como orientação do ponto de interesse. Esse processo é demonstrado nas linhas 7 à 9.

Para cada ponto de interesse encontrado, são extraídos 64 descritores, que são a soma, os valores absolutos, das transformadas de *Haar* para os eixos x e y para cada grupo de 5×5 vizinhos. As linhas 10 à 12 exemplificam esse processo.

Algorithm 4: Algoritmo para extração de descritores de imagem utilizando SURF

```

1 ImagemIntegral = ImagemIntegral(Imagem);
2 List<Octaves> Octaves = ExtrairOctaves(ImagemIntegral);
3 List<PontoInteresse> pontosInteresse;
4 for Octave octave em Octaves do
5   | pontosInteresse.add(ExtrairPontosInteresse(octave));
6 end
7 for PontoInteresse ponto em pontosInteresse do
8   | ponto.add(ExtrairOrientacao(ponto));
9 end
10 for PontoInteresse ponto em pontosInteresse do
11   | ponto.add(ExtrairDescritores(ponto));
12 end
13 return pontosInteresse.
```

2.4 Reconhecimento de Padrões

O Segundo módulo de um sistema de reconhecimento de gestos é o módulo de classificação. Neste módulo, técnicas de reconhecimento de padrões capazes de classificar um gesto a partir das características extraídas pelas técnicas de extração são utilizadas. Três técnicas foram escolhidas para serem utilizadas nesta dissertação, e são debatidas nos tópicos a seguir: Modelos Ocultos de Markov (HMM), *Dynamic Time Warping* (DTW) e Rede Neural Recorrente de Elman (Elman RNN). As três técnicas de classificação, HMM, DTW e Elman RNN, foram selecionadas devido à sua natureza para reconhecimento de padrões que evoluem com o tempo.

2.4.1 Modelos Ocultos de Markov - HMM

Os Modelos Ocultos de Markov (HMM), são métodos estocásticos *Markovianos* que emitem símbolos, denominados observáveis, de forma indireta como descrito por [Rab90]. Para compreender melhor um HMM é necessário um entendimento preliminar sobre as cadeias de Markov. Modelos de Markov são processos estocásticos onde a probabilidade de um determinado evento ocorrer depende somente do estado atual da variável observada [DdM10]. HMMs são considerados processos

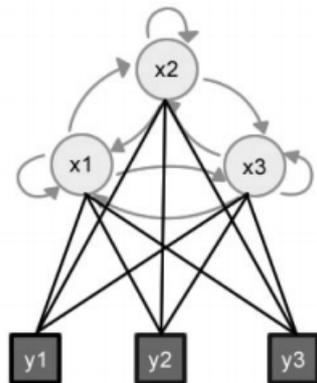


Figura 2.15: Representação gráfica de um HMM contendo três estados (x_1 , x_2 e x_3) e os observáveis que podem ser emitidos por eles (y_1 , y_2 e y_3).

duplamente estocásticos, já que o que são observados não são as transições entre estados e sim um determinado sinal que a variável emite estando em um estado.

Em um HMM a evolução do sistema é observada de forma indireta, captando-se o comportamento dos observáveis, ou seja, o sinal emitido pela variável ao passar por um determinado estado. Dessa forma não é possível saber o caminho ou sequência de passos exatos pela qual a variável percorreu o sistema, mas é possível classificá-la pelos observáveis que ela emite, através de uma função probabilística de transição entre os estados definidos num espaço de estados discreto e finito.

Um HMM pode ser definido por

$$\lambda = (A, B, \pi), \quad (2.17)$$

sendo o HMM com um número finito de estados, A representa sua matriz de probabilidades de transições entre os estados, B representa o conjunto de probabilidades dos observáveis serem emitidos e π é o vetor de probabilidades de distribuição inicial.

Cada elemento a_{ij} , de A , representa a probabilidade de saída do estado i para o estado j . Em π cada elemento h representa a probabilidade do estado inicial ser h . Cada elemento b_{ij} , de B , representa a probabilidade do observável j ser emitido no estado i . Essa matriz pode ser composta por valores discretos ou por distribuições contínuas, como por exemplo na aplicação para reconhecimento de gestos dinâmicos, onde cada observável passa a ser um gesto executado. A distribuição contínua padrão utiliza um conjunto de k distribuições normais $g_{jk}(x)$ e k pesos c_{jk} , onde a probabilidade de emissão é dada por $b_{jk} = \sum_k c_{jk} g_{jk}(x)$. A Figura 2.15 exibe uma representação gráfica de um HMM composta por três estados (x_1 , x_2 e x_3) e o conjunto de observáveis que podem ser emitidos (y_1 , y_2 e y_3).

Para poder classificar uma sequência de observáveis emitida por um HMM, uma sequência de procedimentos é utilizada. Essa sequência resolve os denominados três problemas canônicos, ou fundamentais [Rab90]. Através da solução desses

três problemas é possível obter uma sequência de observáveis ao se inserir uma variável em um HMM, classificar essa sequência de observáveis e maximizar a probabilidade de uma sequência de observáveis, calibrando o modelo para a tarefa de classificação. A definição desses problemas é dada abaixo:

- O primeiro problema procura responder a seguinte questão: dada a sequência de observáveis $O = O_1, O_2, \dots, O_n$, como calcular de forma eficiente a probabilidade dessa sequência ser gerada pelo modelo $\lambda = (A, B, \pi)$, $P(O | \lambda)$?
- Já o segundo problema procura a resposta para: seja um modelo $\lambda = (A, B, \pi)$ e uma sequência de observáveis $O = O_1, O_2, \dots, O_n$, qual a mais provável sequência de estados que poderiam ter gerado esses observáveis?
- O último problema, denominado *Training* procura responder a pergunta: qual o melhor ajuste dos parâmetros do modelo $\lambda = (A, B, \pi)$ para maximizar $P(O | \lambda)$?

Para cada um desses problemas, um algoritmo diferente é utilizado. Para resolução do primeiro problema, o algoritmo *Forward-Backward* é o mais indicado [BPSW70]. Nesse algoritmo, uma variável *Forward* ($\alpha_t(i)$) representa a probabilidade de que para um determinado modelo λ a primeira sequência de observáveis até O_t é gerada ao tempo t e o estado i é alcançado por

$$\alpha_t(i) = P(O_1, O_2, \dots, O_t, S_t = i | \lambda) \quad (2.18)$$

ou seja, $\alpha_t(i)$ representa a probabilidade parcial dos observáveis, de O_1 até O_t , agregada à probabilidade de ocupação do estado S_i no tempo t^1 . Como trabalha-se com sequências em função do tempo, pode-se dizer que trabalha-se com conjuntos ordenados de eventos, o que implica dizer que $\alpha_t(i)$ vale para qualquer instante de tempo dentro do intervalo $0 \leq t \leq T$, onde t representa o tempo parcial de observação e T o tempo total. Para calcular os diversos valores de $\alpha_t(i)$ o seguinte procedimento é aplicado:

- Inicialização:

$$\alpha_0(i) = \pi_i, \quad (2.19)$$

onde i varia de 1 até a quantidade total de estados.

- Indução:

$$\alpha_{t+1}(j) = \sum_{i=1}^N \alpha_t(i) a_{ij} b_{ij}(O_{t+1}), \quad (2.20)$$

onde j varia de 1 até a quantidade total de estados N e t varia de 0 até $T - 1$.

- Finalização:

$$P(O | \lambda) = \sum_{i=1}^N \alpha_T(i). \quad (2.21)$$

Já as variáveis $\beta_t(i)$, denominadas *Backward*, representam a probabilidade de gerar a sequência parcial de observáveis $O_{t+1}, O_{t+2}, \dots, O_T$ a partir do tempo $t + 1$ iniciando em $S_t(i) = 0$ em um determinado modelo λ . Ou seja, a probabilidade de estar no estado S_i no instante t com a probabilidade da sequência parcial de observáveis após t ter sido emitida

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \dots, O_T \mid q_t = S_i, \lambda). \quad (2.22)$$

onde q_t é o estado observado.

Para calcular os diversos valores de $\beta_t(i)$ deve-se utilizar o seguinte procedimento:

- Inicialização:

$$\beta_t(i) = 1, \quad (2.23)$$

onde i varia de 1 até a quantidade total de estados.

- Indução:

$$\beta_t(i) = \sum_{j=1}^N \alpha_{ij} b_{ij}(O_{t+1}) \beta_{t+1}(j), \quad (2.24)$$

onde t começa de $T - 1$ e segue sempre sendo decrescido de 1 até atingir 0, e i varia de 1 até a quantidade total de estados.

A partir de qualquer uma das duas variáveis, $\alpha_t(i)$ e $\beta_t(i)$, é possível calcular a probabilidade da sequência O ter sido emitida pelo modelo λ , dado por

$$P(O \mid \lambda) = \sum_{i=1}^N \alpha_T(i), \quad (2.25)$$

que é equivalente a

$$P(O \mid \lambda) = \sum_{i=1}^N \pi_i b_i(O_1) \beta_1(i). \quad (2.26)$$

Para a solução do segundo problema, o algoritmo de *Viterbi* [For73] é utilizado. O algoritmo de *Viterbi* ajuda a achar a mais provável sequência completa de estados Q que pode ter gerado uma sequência de observáveis O , ou seja maximizar $P(Q \mid O, \lambda)$. Para tanto é dada a seguinte definição

$$\delta_t(j) = \max_{q_1, q_2, \dots, q_{t-1}} P[q_1, q_2, \dots, q_t = S_j, O_1 O_2 \dots O_t \mid q_0 = S_i, \lambda], \quad (2.27)$$

onde $\delta_t(j)$ representa a probabilidade da sequência de estados mais prováveis que leva ao estado S_j no instante t , gerando os primeiros t observáveis. Utilizando o vetor $\psi_t(k)$ para armazenar a sequência de estados no tempo t , o algoritmo de *Viterbi* é executado seguindo procedimento a seguir:

- Inicialização

$$\delta_1(j) = a_{ij}b_{ij}(O_1), \quad (2.28)$$

$$\psi_1(j) = 0, \quad (2.29)$$

onde j varia de 1 até a quantidade de estados.

- Indução

$$\delta_t(k) = \max_{1 \leq j \leq N} [\delta_{t-1}(j)a_{jk}b_{jk}(O_t)], \quad (2.30)$$

$$\psi_t(j) = \operatorname{argmax}_{1 \leq k \leq N} [\delta_{t-1}(j)a_{jk}], \quad (2.31)$$

onde t varia de 2 até T , sendo T o tempo total e K varia de 1 até o total de estados N ;

- Finalização

$$P = \max_{1 \leq k \leq N} [\delta_T(k)], \quad (2.32)$$

$$Q^*_{*T} = \max_{1 \leq k \leq N} [\delta_T(k)]. \quad (2.33)$$

Para recriar o caminho para a melhor sequência, ou seja, aquela para obter o P máximo, basta realizar a seguinte operação:

$$Q_t^* = [\psi_t + 1, \{Q^*_{*t+1}\}], \quad (2.34)$$

para t começando de $T - 1$ e diminuindo unitariamente até atingir 1. Após essa operação Q^* armazenará o caminho mais adequado. Caso ao final do algoritmo, mais de uma sequência possua a mesma probabilidade de ocorrência, uma delas é selecionada arbitrariamente.

O ultimo problema utiliza o algoritmo de *Baum-Welch* [BPSW70]. Esse algoritmo consegue otimizar os parâmetros do HMM de forma que ele possa maximizar a probabilidade de gerar uma sequência completa de observáveis, $P(O | \lambda)$. O algoritmo de *Baum-Welch* funciona otimizando a probabilidade de determinada sequência ter sido gerada pelo modelo. Portanto em cada iteração ele deve gerar um novo conjunto de parâmetros que otimize a probabilidade obtida pela iteração anterior.

Os valores atualizados de cada parâmetro são calculados com base em três funções: probabilidade a posteriori para ocorrência do estado i no tempo t , denominada $\gamma(i)$; probabilidade a posteriori da transição de um estado i para um estado j no tempo t , denominada $\gamma(i, j)$ e a probabilidade de selecionar no estado

j o k -ésimo componente da distribuição de probabilidade no tempo t para gerar a observação contínua O_t , denominada $\varepsilon(j, k)$. Essas três funções são definidas como

$$\gamma(i) = P(S_t = i | O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{P(O | \lambda)}, \quad (2.35)$$

$$\gamma(i, j) = P(S_t = i, S_{t+1} = j | O, \lambda) = \frac{\alpha_t(i)a_{ij}b_{ij}(O_{t+1})\beta_{t+1}(j)}{P(O | \lambda)}, \quad (2.36)$$

e

$$\varepsilon(j, k) = P(S_t = j | O, \lambda) = \frac{\sum_{i=1}^N \alpha_t(i)a_{ij}c_{jk}g_{jk}(O_t)\beta_t(j)}{P(O | \lambda)}. \quad (2.37)$$

Para a otimização dos parâmetros, o algoritmo de *Baum-Welch* utiliza os seguintes procedimentos:

- Inicialização

Um modelo inicial $\lambda = (A, B, \pi)$ é definido, contendo estimativas iniciais para cada parâmetro.

- Otimização

A atualização de cada parâmetro é calculada para a criação de um novo modelo, $\lambda' = \{A', B', \pi'\}$. Para cada elemento, a_{ij} da matriz A' , têm-se

$$a_{ij} = \frac{\sum_{t=1}^{T-1} \gamma_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}. \quad (2.38)$$

Já o valor do vetor π' é atualizado através da seguinte operação

$$\pi' = \gamma_1(i). \quad (2.39)$$

Cada valor dos pesos c das distribuições de probabilidade, assim como μ' e C' são calculados como:

$$c_{jk}' = \frac{\sum_{t=1}^T \varepsilon_t(j, k)}{\sum_{t=1}^T \gamma_t(j)}, \quad (2.40)$$

$$\mu_{jk}' = \frac{\sum_{t=1}^T \varepsilon_t(j, k)x_t}{\sum_{t=1}^T \varepsilon_t(j, k)}, \quad (2.41)$$

e

$$C_{jk}' = \frac{\sum_{t=1}^T \varepsilon_t(j, k)x_t(x_t^T)}{\sum_{t=1}^T \varepsilon_t(j, k)} - \mu_{jk}'\mu_{jk}^{T'}. \quad (2.42)$$

- Validação

Alguns parâmetros para finalização da etapa de otimização devem ser escolhidos, como um resultado superior ao obtido anteriormente, ou ao atingir-se uma quantidade f de iterações no processo.

Os HMMs vêm sendo utilizados para a classificação de padrões temporais, como citado anteriormente, e uma dessas aplicações é para o reconhecimento de gestos dinâmicos. Neste trabalho, o HMM apresenta bons resultados, como exibidos no Capítulo 4.

2.4.2 Algoritmo HMM

O Algoritmo 5 exemplifica o pseudocódigo para treinamento de um HMM. Para iniciar o treinamento é necessário calcular uma distribuição inicial para os parâmetros, exemplificada na linha 2. Após isso, inicia-se o treinamento através do algoritmo de Baum-Welch. Nesse algoritmo, os dados de treino são classificados e o erro é calculado e utilizado como parâmetro para o cálculo das novas probabilidades e atualização dos parâmetros do HMM, como ilustrado entre as linhas 3 e 7. Até a condição de parada ser obtida, o algoritmo continua a atualização dos parâmetros.

Já o Algoritmo 6 exibe a classificação de um padrão utilizando um classificador baseado em modelos HMM. Para cada HMM presente no classificador, os observáveis são calculados utilizando os dados de classificação como entrada, como ilustrado nas linhas 4 e 5. Após isso, a probabilidades do modelo ter gerado esses observáveis é calculada, Linha 6. O modelo que possuir maior probabilidade é escolhido como o que representa os dados inseridos.

Algorithm 5: Algoritmo para treino de um modelo HMM

```
1 Dados dadosTreino;  
2 DefinirDistribuicoesInicias();  
3 while condição de parada não for atingida do  
4   | classificacao = classificar(dadosTreino);  
5   | erro = calculaErro(classificacao);  
6   | probabilidades = calcularProbabilidades(erro);  
7   | atualizarParametros(probabilidades);  
8 end
```

2.4.3 *Dynamic Time Warping - DTW*

Dynamic Time Warping (DTW) é uma técnica para encontrar o melhor alinhamento entre duas sequências, não necessariamente possuindo o mesmo tempo e espaço. O DTW vem sendo utilizado na área de reconhecimento de voz, mas algumas recentes aplicações utilizam sua capacidade de comparação entre sequências para o reconhecimento de gestos dinâmicos [KZ07] [HL10] [HR12].

Algorithm 6: Algoritmo para a classificação utilizando HMM

```
1 Dados dados;
2 double maiorProbabilidade = 0;
3 HMM hmmEscolhido;
4 for HMM hmm no classificador do
5   | Observaveis obs = hmm(dados);
6   | double probabilidade = calcularProbabilidade(obs,hmm);
7   | if maiorProbabilidade = probabilidade; hmmEscolhido = hmm; then
8   |   | probabilidade > maiorProbabilidade
9   |   | end
10 end
11 return hmmEscolhido.
```

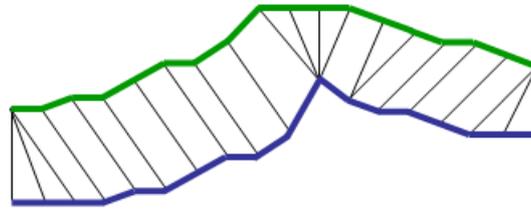


Figura 2.16: Duas seqüências com tamanhos diferentes sendo alinhadas utilizando o DTW.

O objetivo primário do DTW é a comparação de duas seqüências $X = (x_1, x_2, x_3, \dots, x_N)$ de tamanho N , e $Y = (y_1, y_2, y_3, \dots, y_M)$ de tamanho M . Essas seqüências podem ser sinais discretos ou uma seqüência de características dinâmicas, extraídas de padrões que evoluem com o tempo. Para calcular a distância entre essas duas seqüências X e Y , uma medida de custo local, denominada c é necessária. Normalmente c possui um valor menor quando X e Y são parecidas e um valor maior quando são muito diferentes. Uma matriz de custo C é definida pela diferença de cada par de elementos presente em X e Y , ou seja cada elemento c_{ij} é composto pela diferença entre o elemento i de X e o elemento j de Y , e o objetivo do DTW é encontrar o melhor alinhamento para X e Y , obtendo o menor custo para a matriz C . A Figura 2.16 exibe o alinhamento entre duas seqüências de tamanhos diferentes, utilizando o DTW.

O DTW define como $P = (p_1, p_2, \dots, p_L)$ como o caminho de menor custo da matriz C , onde $p_l = c_{ij}$. A Figura 2.17 exemplifica a matriz de custos C para duas seqüências. Para tanto, P tem que satisfazer as seguintes condições:

- **Monotonicidade:** O caminho percorrido não pode voltar no tempo, ou seja, p_{x+1} está sempre à frente de p_x . Essa condição garante que não existam elementos com distâncias repetidas. Isso implica dizer que: $n_1 \leq n_2 \leq \dots \leq n_L$ e $m_1 \leq m_2 \leq \dots \leq m_L$.
- **Continuidade:** O caminho percorrido não pode saltar no tempo. Isso quer

Algorithm 7: Algoritmo para execução do cálculo de distâncias através do DTW

```
1 Sequencia X;
2 Sequencia Y;
3 MatrizCusto C;
4 for Cada elemento  $x_i$  em X do
5   | for Cada elemento  $y_j$  em X do
6   |   |  $C_{ij} = \text{distancia}(x_i, y_j)$ ;
7   | end
8 end
9 MenorDistancia P = menorDistancia(C);
10 return P.
```

2.4.5 Rede Neural Recorrente de Elman - Elman RNN

Redes Neurais [AH08] vem sendo utilizadas na área de reconhecimento de padrões há muito tempo. Sua capacidade de generalização, treinamento para novos padrões e robustez de resultados garantem sua funcionalidade para várias aplicações, entre elas reconhecimento de gestos [NSPF08] [ASKH12] [HA11] [CC02]. Um dos problemas envolvidos em reconhecimento de gestos dinâmicos é a composição do gesto, formado por várias posições de uma mão que varia com o tempo. Para esse problema, uma rede neural *Multilayer Perceptron* não é indicada, pois ela não é capaz de reconhecer padrões em um fluxo de tempo. Para tanto, uma rede neural recorrente é mais apropriada, pois a saída da camada oculta da rede para um dado padrão num instante de tempo t é utilizada como entrada para a rede no instante t_1 , permitindo a aprendizagem de padrões que variam com o tempo.

Para compreender uma rede neural recorrente, é necessário o entendimento do conceito de rede neural multi-camadas, a *Multilayer Perceptron* (MLP). Uma MLP é uma rede neural composta por neurônios artificiais, perceptrons, interligados e compondo várias camadas. Um perceptron simula o funcionamento de um neurônio biológico, assim sendo os dendritos foram substituídos por entradas, cujas ligações com o corpo celular artificial são realizadas através de elementos chamados pesos, simulando as sinapses. Os estímulos captados pelas entradas são processados pela função de soma e o limiar de disparo do neurônio biológico foi substituído pela função de transferência [F.58]. A Figura 2.18 exhibe uma ilustração de um perceptron.

Um perceptron exhibe uma saída baseada na soma das suas entradas e se o resultado dessa soma atinge ou não um determinado limiar. Ele é capaz de separar linearmente vetores de entrada em classes de padrões através de hiperplanos. A função soma, a , é definida pela soma das entradas, X , ponderadas pelos pesos, W . Cada entrada representa um elemento no vetor de características que se quer classificar e uma entrada a mais é adicionada, denominada *bias*, que possui sempre valor 1 e serve como uma forma de ajustar o desvio dos pesos de um perceptron. O campo receptivo do perceptron pode ser descrito matematicamente como

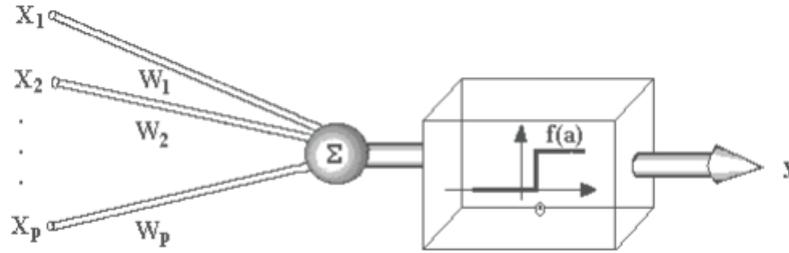


Figura 2.18: Representação gráfica de um peceptron, onde X_1, X_2, X_p representam as entradas, W_1, W_2, W_p representam os pesos, Σ representa a função soma e $f(a)$ representa o limiar de ativação. Fonte: [Hay99]

$$a = \sum_{i=0}^p X_i W_p, \quad (2.44)$$

onde p é a quantidade total de entradas para o neurônio. O resultado dessa função é comparado com um limiar previamente definido l , e o resultado expelido pela saída do neurônio. A função $f(a)$ delimita o limiar de ativação do perceptron

$$f(a) = \begin{cases} 1, & a > l \\ 0, & a \leq l \end{cases} \quad (2.45)$$

Para a classificação de padrões mais complexos, uma arquitetura MLP é utilizada. Por convenção a primeira camada dessa arquitetura é denominada camada de entrada, as camadas subsequentes são chamadas de camadas escondidas e a última camada é denominada camada de saída [AH08].

Cada camada de uma MLP é formada por um ou mais perceptrons. A entrada de cada camada é dada pela saída dos neurônios da camada anterior, exceto a camada de entrada, onde a entrada é dada por cada elemento do vetor de características que se deseja classificar. Cada valor do vetor de características deve ser passado a um neurônio na camada de entrada e a saída desses neurônios são propagadas para todos os neurônios da camada subsequente até que cheguem à camada de saída. A Figura 2.19 exhibe a arquitetura de uma MLP.

Nas camadas escondidas de uma MLP, uma função de ativação, $f(a)$, é utilizada para delimitar a saída de cada neurônio. A função sigmoide é muito utilizada para a maioria dos casos de classificações, sendo definida como

$$f(a) = \frac{1}{1 + e^{-\lambda a}}. \quad (2.46)$$

Em outros casos essa função também pode ser escrita como segue

$$f(a) = \frac{1}{2} + \frac{1}{2} \tanh\left(\frac{\lambda a}{2}\right). \quad (2.47)$$

Em uma rede neural o conhecimento para classificação está armazenado diretamente nos pesos das entradas de cada neurônio. Isso implica dizer que um conjunto

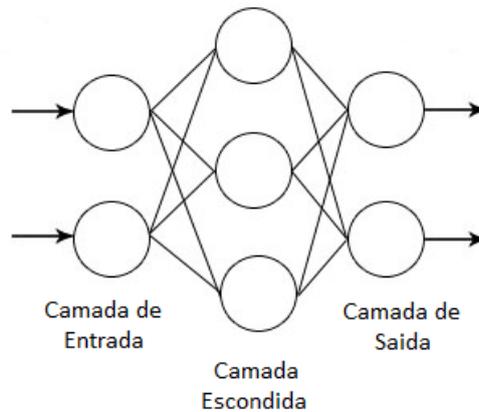


Figura 2.19: Representação gráfica de uma MLP contendo as camadas de entrada, uma camada escondida e a camada de saída.

de pesos precisa ser escolhido para o sistema de forma que a rede possa classificar os padrões corretamente, para isso o algoritmo *Backpropagation* [RHW86], é uma solução apropriada.

O *Backpropagation* consiste em um algoritmo que encontra a melhor distribuição de pesos para uma rede neural dado um certo conjunto de dados. Ou seja, para um conjunto de dados de treinamento, ele encontra os pesos que possibilitam que a rede neural generalize uma nova entrada, classificando-a. Para tanto o *Backpropagation* utiliza o conceito de retropropagação do erro pela rede, ajustando os pesos quando necessário. O seguinte procedimento é realizado durante a execução do *Backpropagation*:

- Inicialização: todos os pesos dos neurônios na rede são inicializados com valores aleatoriamente escolhidos no intervalo $[-1,1]$.
- Exemplos de treino: um conjunto de dados é selecionado para se treinar a rede, ou seja, a rede irá ser ajustada de acordo com esse conjunto de treino. Esse conjunto T , é composto pelos vetores de características, \vec{x} , e a saída esperada para esse vetor de características, d . Assim T é definido por

$$T = (\vec{x}_1, d_1), (\vec{x}_2, d_2), \dots, (\vec{x}_n, d_n); \quad (2.48)$$

- Propagação: utilizando cada entrada \vec{x} do conjunto de treino calcule a saída da rede neural, $O_j(\vec{x})$. A saída é dada pela função de ativação da camada de saída, j .
- Sensibilidade do erro na camada de saída: utilizando a saída $O_j(\vec{x})$, o erro na camada de saída $e_j(\vec{x})$ é dado por:

$$e_j(x) = d_j(\vec{x}) - O_j(\vec{x}) \quad (2.49)$$

sendo que $d_j(\vec{x})$ representa a saída esperada da rede. Esse sinal será utilizado para calcular os valores dos erros nas camadas anteriores e fazer a correção dos pesos.

- Retropropagação: para cada neurônio, os erros locais σ são calculados. Esse erro é calculado a partir da camada de saída e é propagado em ordem inversa até a camada de entrada. O erro local para a camada de saída é calculado como

$$\sigma_j(x) = e_j(x)O_j(x)(1 - O_j(x)). \quad (2.50)$$

Já para as demais camadas o erro é calculado como:

$$\sigma_j(x) = O_j(x)(1 - O_j(x)) \sum \sigma_k w_{jk}, \quad (2.51)$$

onde σ_k representa o erro de cada neurônio da camadas anterior que está conectado ao neurônio j , e w_{jk} representa o peso anexados a essas conexões. Depois de calcular o erro para todos os neurônios, o ajuste dos pesos é necessário. Para tanto, o valor do peso atual w_{jk} é alterado pelo valor da correção do peso Δw_{jk} , onde Δw_{jk} é definido por

$$\Delta w_{jk} = \alpha w_{jk} + \eta \sigma_j O_j(x), \quad (2.52)$$

onde α é a constante de *momentum* que determina o efeito das mudanças passadas dos pesos na direção atual do movimento no espaço de pesos. η é a taxa de aprendizado, uma constante no intervalo $[0,1]$, que indica quão rápido o modelo aprende a generalizar os exemplos passados: σ_j representa o erro no neurônio j , calculado anteriormente e $O_j(x)$ a saída do neurônio.

- Iteração: após o ajuste do peso para um elemento do conjunto de treino, outro elemento é passado para um novo ajuste. Essa iteração ocorre até que a taxa de erro total da rede seja menor que um determinado limiar ou que um número máximo de ciclos de ajuste ocorram. Essa condição é denominada condição de parada e deve ser escolhida com cuidado pois uma rede pode ser treinada até que ela apresente um erro 0 para os padrões exibidos. Essa condição é denominada *overfitting* o que quer dizer que a rede superajustou os pesos, decorando os melhores pesos para os padrões dados, fazendo com que a capacidade de generalização de um novo padrão seja prejudicada. Outra condição, denominada *underfitting*, pode acontecer quando o ajuste dos pesos é parado antes que a rede possua a capacidade de generalizar os padrões, fazendo com que a classificação de um novo padrão seja dada incorretamente.

Para o reconhecimento de um padrão estático uma rede do tipo MLP se mostra adequada, mas para o reconhecimento de um padrão que evolui em uma determinada linha de tempo, uma MLP não é indicada. Para tanto, uma rede neural

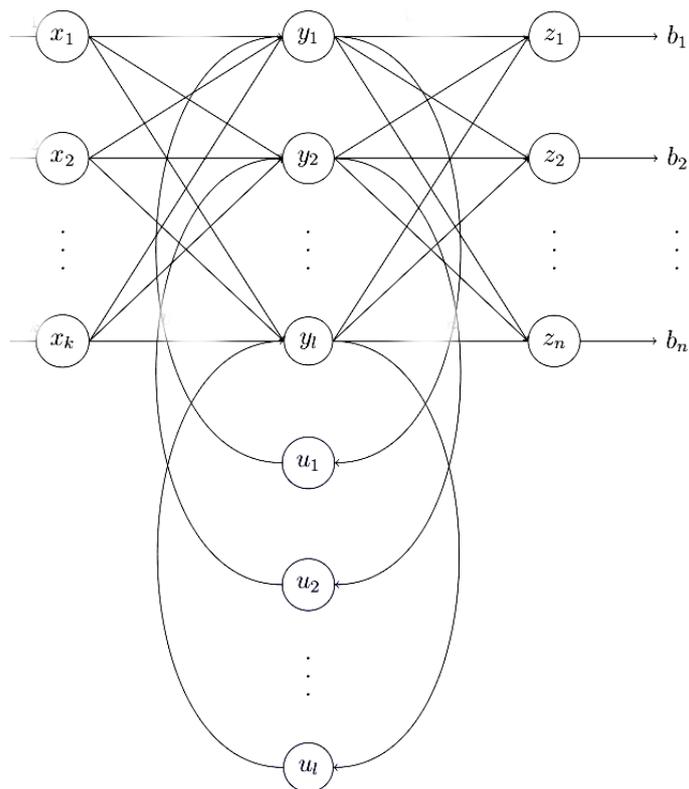


Figura 2.20: Representação gráfica de uma Elman RNN contendo as camadas de entrada, escondida, de saída e de contexto.

recorrente é a escolha mais coerente. Uma rede neural recorrente é uma MLP com uma camada a mais, denominada camada de recorrência ou contexto, que permite que a rede consiga generalizar padrões que são dependentes entre si.

Uma das redes neurais utilizadas para o reconhecimento de gestos é a Rede Neural Recorrente de Elman (Elman RNN), descrita por Jain [Jai01]. Uma Elman RNN define uma nova camada de contexto, no qual cada neurônio dessa nova camada recebe como entrada um neurônio da camada escondida. A saída desse neurônio na camada de contexto é utilizada como entrada no neurônio da camada escondida para o próximo padrão. A Figura 2.20 exibe uma Elman RNN graficamente.

Essa camada de contexto na Elman RNN garante que o padrão anterior vai influenciar na classificação do padrão atual. A propagação dos padrões na rede funciona da mesma forma que em uma MLP, sendo a única diferença que um padrão só será classificado após uma certa quantidade de padrões terem sido apresentados previamente à rede. Por exemplo, um gesto dinâmico é composto por 15 posições de mão. Para classifica-lo com uma Elman RNN, é necessário que as quinze posições sejam passadas pela rede, uma de cada vez, e ao final da 15ª o padrão é classificado.

Outra grande diferença da Elman RNN é no treinamento. O algoritmo *Back-propagation* não funciona muito bem para esse fluxo de padrões, ficando preso em

mínimos locais na sua execução. Para resolver tal problema, Zhang et al. [ZWD11] utiliza a técnica *Simulated Annealing* para gerar um espaço de busca maior, retirando o *Backpropagation* dos mínimos locais encontrados. A mesma técnica pode ser vista no trabalho de Amato et al. [AAC⁺91].

Annealing é uma técnica utilizada para fundição de metais, onde este é aquecido a uma temperatura elevada e em seguida resfriado lentamente, de modo que o produto final seja uma massa homogênea. O *Simulated Annealing* [KGV83] se baseia nos princípios do *Annealing* para problemas de otimização combinatória. Nesse contexto, o processo de otimização, $\min_x f(x)$, é realizado em níveis, simulando os níveis de temperatura do resfriamento. Em cada nível, dado um ponto μ , vários pontos na vizinhança de μ são gerados e o correspondente valor de f é calculado. Cada ponto é aceito ou rejeitado de acordo com certa probabilidade. Essa probabilidade de aceitação decresce de acordo com o nível do processo, ou equivalente, de acordo com a temperatura.

A ideia por trás da utilização do *Simulated Annealing* no *Backpropagation* para a Elman RNN é evitar os mínimos locais. Então, quando o *Backpropagation* atingir um ponto de estabilização no erro, o *Simulated Annealing* é executado, procurando por pontos próximos e aumentando o nível da busca até que o erro diminua, evitando o mínimo local. No contexto do *Simulated Annealing*, os parâmetros otimizados são os parâmetros passados para o *Backpropagation*, cada nível de temperatura representa um novo erro obtido com os parâmetros do *Backpropagation*, ou seja, no lugar de aumentar o nível de temperatura, o algoritmo do *Simulated Annealing* tentará diminuí-lo, diminuindo o erro obtido.

2.4.6 Algoritmo Elman RNN

O algoritmo de treinamento de uma Elman RNN, Algoritmo 8, começa com a seleção de um conjunto de dados para treino, e a definição de suas saídas esperadas, como exibidos nas linhas 1 e 2. Após a criação do modelo, o *Backpropagation* é utilizado para treinar o modelo, ajustando os pesos de acordo com o conjunto de treino e as saídas esperadas. Esse treino ocorre até que uma condição de parada seja definida. Caso o *Backpropagation* esteja preso em um mínimo local, ou seja, o erro não diminui, o *Simulated Annealing* é chamado, expandindo a área de busca. O *Simulated Annealing* precisa de uma condição de parada, já que pode-se ter sido atingido um mínimo global. Esse procedimento está exibido entre as linhas 4 a 11.

Após o treinamento, o Algoritmo 8, de classificação, é executado. Um novo padrão é propagado pela rede, e a saída é considerada sua classificação, exemplificado na linha 3 do Algoritmo 9.

Algorithm 8: Algoritmo para treino uma Elman RNN.

```
1 List<Dados> Treino;
2 List<Classe> SaidasEsperadas;
3 RNN rnn = construirModelo();
4 for Dado i em Treino do
5     while condição de parada do
6         Erro = rnn.Backpropagation(i, SaidasEsperadas(i));
7         while Erro não diminui ou a condição de parada do Simulated
           Annealing seja atingida do
8             rnn.SimulatedAnnealing();
9         end
10    end
11 end
```

Algorithm 9: Algoritmo para classificação em uma Elman RNN.

```
1 ElmanRNN elmanRNN;
2 treinar(elmanRNN);
3 classe = elmanRNN.classificar(padrao);
4 return classe;
```

Capítulo 3

Reconhecimento de Gestos Dinâmicos Através da Abordagem Convexa

3.1 Introdução

Os sistemas para reconhecimento de gestos normalmente são divididos em três partes: pré-processamento da imagem, extração de características e reconhecimento de padrões. Sistemas baseados em visão computacional recebem dados a partir de câmeras de vídeo, que são pré-processados eliminando ruído ou realçando áreas de interesse. Esses dados devem passar por um módulo de extração de características, de forma que possam ser computacionalmente representáveis. Um módulo de classificação recebe as características e estima a qual classe ela pertence, normalmente utilizando uma base de dados previamente classificada. Algumas técnicas de extração de características pulam a primeira parte, o pré-processamento de imagem. Uma dessas técnicas, o *Speed Up Robust Feature* - SURF utilizada nesta dissertação, é explicada no capítulo anterior.

O sistema descrito nesse trabalho não é diferente do sistema padrão, sendo composto pelas três etapas: pré-processamento, extração e classificação. Como se trata de um sistema para o reconhecimento de gestos dinâmicos, o sistema de extração recebe várias imagens capturadas por uma câmera de vídeo. Os *frames* capturados são enviados ao módulo de extração e as características descritas são enviadas ao módulo de classificação para então o gesto ser identificado. A Figura 3.1 exibe um fluxograma de funcionamento desse sistema.

O módulo de extração de características é responsável por receber os *frames* contendo as posturas da mão e por gerar um arquivo de texto contendo os descritores das imagens. Agindo dessa forma, o módulo de extração pode conter vários tipos de técnicas de extração, o que permite que o sistema consiga ser adaptável ao tipo de gesto que se quer reconhecer. As imagens utilizadas como entrada devem ser as imagens recebidas pela câmera, de forma que o próprio método de extração seja responsável pelo seu pré-processamento e extração dos descritores.

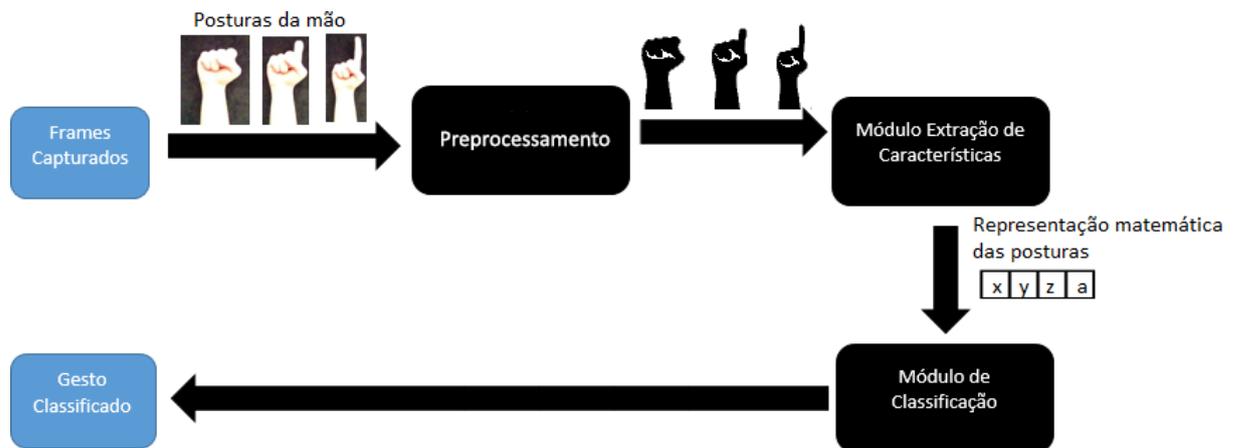


Figura 3.1: Arquitetura que descreve o sistema de reconhecimento de gestos proposto.

O módulo de classificação utiliza os descritores extraídos pelo módulo de extração como forma de entrada. Esse módulo também pode implementar várias técnicas diferentes, de forma a ser adaptável ao problema em questão. As características podem ser enviadas em lote, ou por *frames* individuais, dependendo da técnica de classificação utilizada. Quando as características são extraídas em lote, todos os *frames* que compõem o gesto são enviados ao mesmo tempo para o extrator e ao classificador. Já por *frames* individuais, um *frame* é enviado ao sistema de cada vez, e o sistema vai classificando o gesto *frame* por *frame*.

De forma a resolver o problema de muitas características extraídas em um gesto dinâmico, como problema que caracteriza a maldição da dimensionalidade, uma nova técnica para extração de características é proposta. Essa técnica, denominada Abordagem Convexa, seleciona os pontos presentes no contorno da mão que melhor podem representá-la, sem a necessidade de utilizar o contorno completo. Essa técnica foi descrita formalmente nos nossos trabalhos [BJB⁺13a] e [BJB⁺13b]. A sessão a seguir detalha o funcionamento da Abordagem Convexa.

3.1.1 Organização do capítulo

Uma leitura sequencial de todas as sessões deste capítulo é indicada para um melhor entendimento. Este capítulo está estruturado da seguinte forma:

- Sessão 3.1 - Introdução: apresenta uma análise introdutória sobre o sistema de reconhecimento de gestos dinâmicos proposto;
- Sessão 3.2 - Abordagem Convexa: detalha a técnica de extração de características proposta;

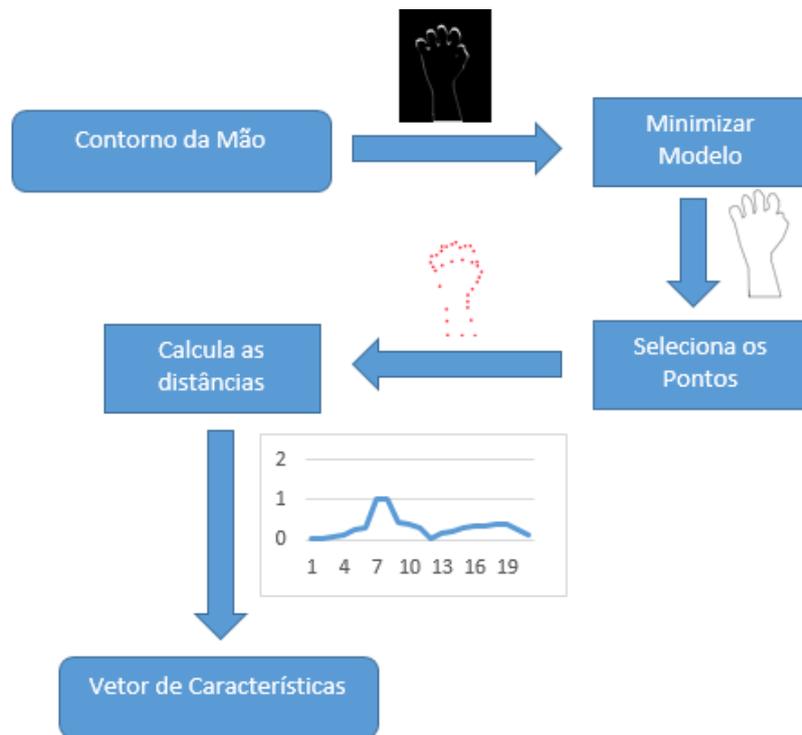


Figura 3.2: Fluxograma que descreve as etapas da Abordagem Convexa, método de extração de características para reconhecimento de gestos.

3.2 Abordagem Convexa

Como visto no capítulo anterior, algumas técnicas de extração de características representam um gesto com um vetor de tamanho elevado, o que pode acarretar em alguns problemas na hora da classificação, como tempo elevado para classificação, altas taxas de erros altas e até mesmo a não viabilidade do uso dessas técnicas para reconhecimento de tempo real. Para resolver tal problema, esse trabalho propõe uma técnica extração de características que selecionar uma quantidade mínima de pontos para representar uma determinada postura de mão. Esse procedimento reduz a quantidade de informações necessárias para descrever aquela postura, aumentando a taxa de classificação correta dos gestos e diminuindo o tempo de processamento.

A Abordagem Convexa extrai as características de uma imagem por vez, sendo necessária que essa imagem contenha somente o contorno da mão. O primeiro passo do algoritmo é reduzir o modelo geométrico da mão, eliminando curvas e o segundo passo é encontrar uma quantidade mínima de pontos, escolhidos levando em consideração cada postura de mão, que possam representar o modelo geométrico. O último passo é extrair a distância entre os pontos selecionados e compôr o vetor de características que descreverá a mão. A Figura 3.2 exhibe um fluxograma de funcionamento da Abordagem Convexa.

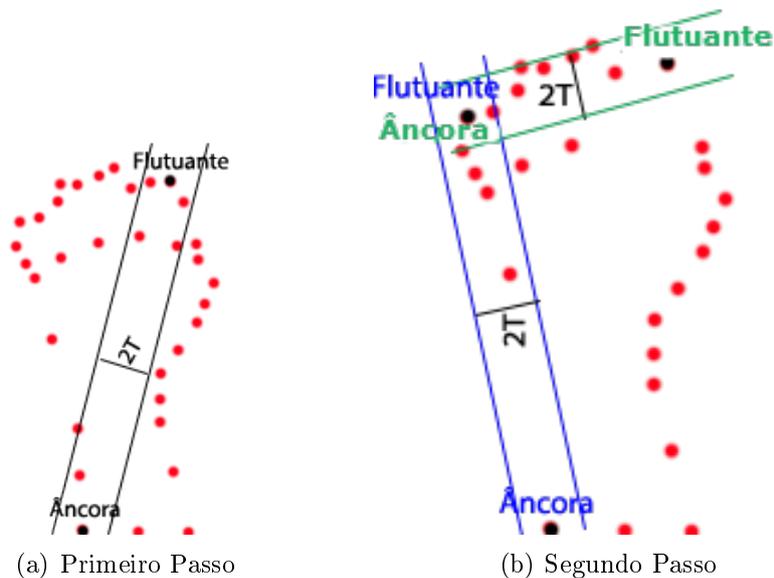


Figura 3.3: Execução do algoritmo Douglas-Peucker. (a) exibe o primeiro passo, delimitando o corredor da distância $2T$. O segundo passo (b) seleciona o ponto mais distante do segmento de linha formados por âncora e flutuante e cria dois subgrupos, onde o algoritmo será executado novamente.

3.2.1 Minimização do modelo geométrico

A primeira etapa do algoritmo garante que qualquer informação excedente seja extraída. No contexto de gestos realizados pelas mãos, entende-se como informação excedente aquela que é redundante ou que pode ser excluída sem que haja perda na descrição da mão. Um exemplo disso é uma curva qualquer, presente no contorno da mão, que pode ser facilmente representada por três pontos, um em cada extremidade e um no centro da parábola. Para criar um modelo minimizado da mão o algoritmo de simplificação poligonal Douglas-Peucker [DP73] é utilizado. Esse algoritmo foi utilizado para o reconhecimento de placas de trânsito [SS11], minimização de modelos por compressão [NPRK05] e aplicações geográficas [YG10].

O Algoritmo de Douglas-Peucker é recursivo e a cada passo processa o intervalo de pontos contidos entre um vértice inicial, denominado âncora, e um final, denominado flutuante. Na primeira execução do algoritmo, os pontos mais distantes entre si do polígono são marcados como âncora e flutuante. Uma distância mínima, denominada tolerância T tem que ser previamente definida. Duas linhas paralelas são traçadas contendo uma largura igual a $2T$, de forma que os pontos âncora e flutuante estejam no centro delas. A Figura 3.3(a) exibe o passo inicial do algoritmo Douglas-Peucker.

Caso todos os pontos do modelo processado estejam contidos dentro do corredor, isso quer dizer que a aproximação inicial formada pelo segmento de reta \overrightarrow{AF} entre o ponto âncora e o flutuante é boa e todos os pontos são eliminados sobrando somente os pontos âncora e flutuante. Caso existam pontos fora do corredor, a

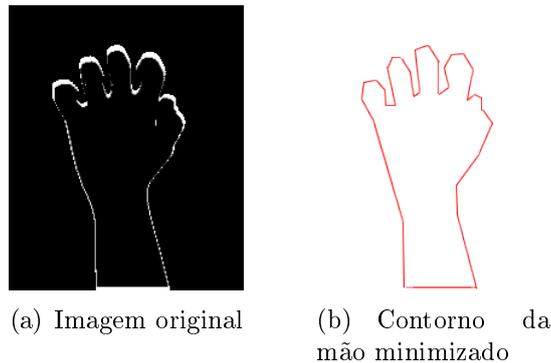


Figura 3.4: Resultado da aplicação do algoritmo de minimização de polígonos Douglas-Peucker na imagem original (a). (b) exibe a saída do algoritmo, com o modelo minimizado, contendo menos curvas e mais vértices.

distância entre esses pontos e o segmento de reta \overrightarrow{AF} são calculadas. O ponto que obtiver a maior distância é selecionado e marcado como mais distante. O algoritmo agora divide o processamento em duas instâncias, uma onde o ponto inicial é o âncora anterior e o ponto flutuante é o ponto mais distante, e a outra onde o ponto âncora se torna o ponto mais distante e o ponto flutuante permanece. O processo é novamente executado, dessa vez em paralelo nessas duas instâncias. O algoritmo é executado até o fim, ou seja, até que todas as aproximações dos subconjuntos sejam consideradas aproximações boas. A Figura 3.3(b) exibe a continuação do processo iniciado na Figura 3.3(a), mostrando os subconjuntos selecionados.

Ao final da execução do algoritmo Douglas-Peucker, o polígono estará minimizado contendo somente os pontos que formarão os segmentos de reta do polígono minimizado. Um dos pontos importantes no algoritmo é a escolha da distância de tolerância T . Com uma distância muito grande, o algoritmo irá ignorar picos e depressões que podem ser importantes na descrição da mão, e uma distância muito pequena poderá capturar informações desnecessárias. A Figura 3.4 ilustra o resultado da aplicação do algoritmo de minimização de polígonos.

3.2.2 Extração dos pontos mais significativos

O segundo passo da Abordagem Convexa é a escolha dos pontos mais significativos no modelo minimizado. Para isso, a seleção dos pontos é realizada em duas etapas: a primeira é a seleção das arestas do polígono que mais se destacam no modelo. O segundo é a escolha das arestas que mais detalham o modelo, baseada nas arestas selecionadas anteriormente.

A primeira etapa da seleção dos pontos mais significativos é realizado através da execução de um fecho convexo ao redor do modelo. Ao se aproximar o fecho convexo do modelo, verifica-se quais vértices o tocam. Esses vértices são marcados, e todos os outros que não tocam o fecho são eliminados. Um fecho convexo, $F(x)$ do conjunto $x \subset \mathbb{R}^2$, $x = x_1, x_2, x_3 \dots x_n$, é a menor região convexa de \mathbb{R}^2 que contém o conjunto x . O algoritmo de Sklansky [Skl82], corrigido posteriormente

por [Mel87], consegue implementar um fecho convexo utilizando o processo descrito no Algoritmo 10.

Algorithm 10: Algoritmo para Extração dos pontos que tocam o fecho convexo de um polígono

```

1  Encontra-se o ponto mais externo do polígono, o com maior valor de
   coordenada y;
2  Ordena-se os pontos restantes em ordem horária, e nomeia-os começando
   por  $P_0$ ;
3  Coloca-se três marcadores nos vértices  $P_0$ ,  $P_1$  e  $P_2$ , e nomeia-os como
   "Atrás", "Centro" e "Frente" respectivamente;
4  while Enquanto "Frente" não é o vértice  $P_0$  e "Atrás", "Centro", e
   "Primeiro" fizerem uma curva para a direita do
5  |   if "Terceiro", "Segundo" e "Primeiro" formarem uma curva à direita ou
   |   forem vértices colineares then
6  |   |   Pega-se "Atrás" e o coloca no vértice subsequente à "Frente";
7  |   |   Coloca-se "Atrás" no lugar de "Frente", "Frente" no lugar de
   |   |   "Centro" e "Centro" no lugar "Atrás";
8  |   end
9  |   else
10 |   |   Coloca-se "Centro" no vértice antes de "Atrás";
11 |   |   Remove o vértice em que "Centro" está;
12 |   |   Coloca-se "Centro" no lugar de "Atrás" e "Atrás" no lugar de
   |   |   "Centro";
13 |   end
14 end
15 Ordena-se os polígonos restantes em ordem horária.

```

O Algoritmo de Sklansky coloca três moedas em cima dos três primeiros vértices do polígono. As três moedas, são alteradas no decorrer do algoritmo, sempre avançando por vértices que formam uma curva à direita. Quando as três moedas realizam uma curva à esquerda, o vértice em que a primeira moeda se encontra deve ser retirado do modelo, volta-se um passo para trás e continua a busca. Isso ocorre até que todo o polígono possa ser representado por vértices que façam curva à direita, eliminando assim qualquer ponto não-convexo no polígono. A Figura 3.5(a) exhibe o resultado da aplicação do algoritmo de Sklansky na imagem da Figura 3.4(b).

Após a execução do fecho convexo, os pontos que detalham o fecho convexo precisam ser encontrados. Como o fecho convexo encontra somente os pontos mais externos, delimitando a forma máxima da configuração da mão, alguns detalhes são perdidos, o que pode prejudicar a identificação do modelo da mão pelas técnicas de classificação. Esses detalhes são encontrados na segunda etapa, onde os pontos internos são detalhados.

Para detalhar os pontos internos utiliza-se os pontos extraídos pelo fecho con-

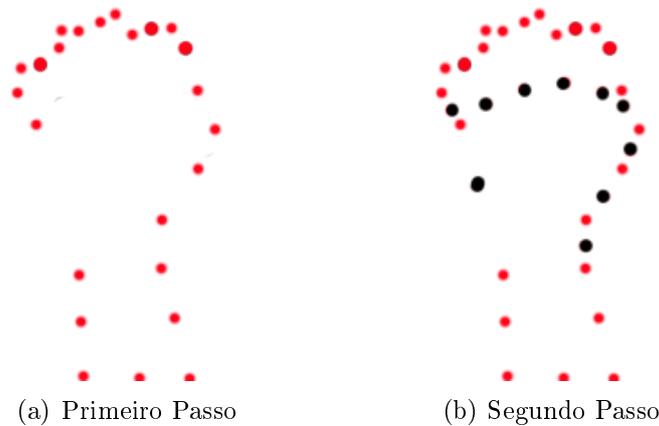


Figura 3.5: A aplicação do algoritmo de Sklansky em um polígono minimizado da imagem de uma mão, gera uma representação que enaltece os vértices externos(a). Já a aplicação do realce nos vértices internos, gera um modelo mais detalhado (b).

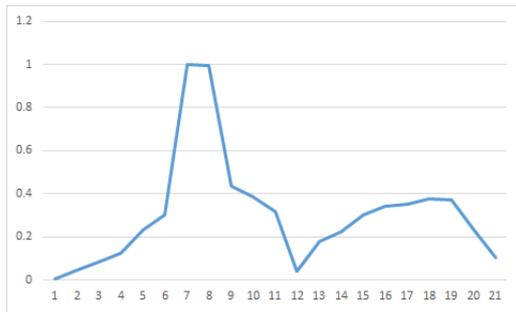
vexo e o modelo original com o contorno da mão. Para cada par de pontos extraídos pelo fecho convexo, uma reta é traçada. A distância entre todos os pontos contidos entre o par de pontos selecionados e a reta traçada é calculada, e o ponto que obtiver a maior distância é selecionado como ponto de detalhe. Caso não exista nenhum ponto ou a distância seja igual a 0, nenhum ponto é considerado. A Figura 3.5(b) exhibe a aplicação do realce dos pontos internos.

3.2.3 Composição do vetor de características

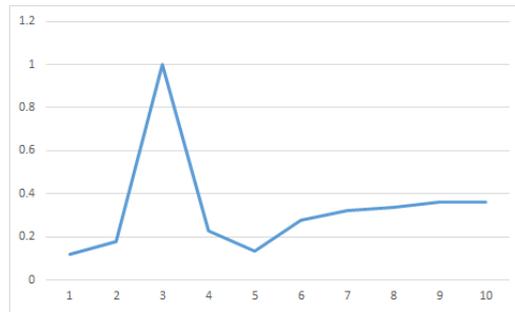
O último passo da Abordagem Convexa é a composição do vetor de características. Esse vetor é utilizado pelos classificadores para identificarem a postura de mão em um gesto, por isso precisam conter informações que descrevam com a postura da mão, sem conter informações redundantes ou irrelevantes. Para tanto um cálculo de distância nos pontos obtidos pelo passo anterior é executado.

A distância calculada entre os pontos permite que a Abordagem Convexa tenha as mesmas características do LCS, porém com menos pontos a serem considerados. A cada par de pontos externos extraídos no passo anterior, verifica-se se existe algum ponto interno entre eles. Caso exista, a distância entre uma reta traçada entre os pontos externos e o ponto interno é calculada e armazenada no vetor de características. Esse procedimento é realizado em todos os pontos, de forma que toda a configuração da mão seja representada. Isso garante que os pontos internos possam ser utilizados para detalhar os pontos externos, delimitando uma dependência na descrição do modelo da posição dos pontos internos e externos. Essa dependência assegura que pontos externos e internos, mesmo que sejam parecidos geometricamente, possam ser descritos de forma diferente no vetor de características.

Com a função de transformar o algoritmo invariante à escala, uma padronização



(a) Gesto extraído com a Abordagem Convexa



(b) Vetor normalizado

Figura 3.6: (a) Resultado da composição do vetor de características para a postura de mão exibida na Figura 3.5(b). Já (b) exibe o mesmo vetor de características normalizado com 10 elementos.

de amplitude no vetor de características é necessária. Isso auxilia no processo de classificação, já que a mesma configuração de mão pode ser executada a distâncias diferentes da câmera, necessitando serem caracterizadas de forma igual. O processo de padronização de amplitude é simples, dividindo o valor de cada distância no vetor de características pelo desvio padrão. A Figura 3.6(a) exibe o vetor de características normalizado em amplitude extraído da postura da Figura 3.5(b).

Para algumas técnicas de classificação, como no caso das redes neurais, os vetores de características precisam estar normalizados em tamanho. Para resolver esse problema um algoritmo de normalização é utilizado. Esta dissertação propõe a utilização de um método para normalização baseado em amostragem. Primeiro o tamanho que se deseja normalizar é escolhido. Se o vetor de características possuir um tamanho menor do que o tamanho definido, valores 0 são adicionados ao final do vetor até que ele atinja o tamanho escolhido. Caso o vetor de características possua um tamanho maior que o tamanho escolhido, um algoritmo de seleção de amostras é utilizado. Esse algoritmo define uma janela w como o quociente da divisão do tamanho do vetor de características pelo tamanho desejado. O vetor de características então é visitado, e cada posição que seja múltiplo de w é adicionada em um novo vetor de saídas. Se o novo vetor de saídas ainda possuir um tamanho menor do que o desejado, as posições restantes, que não foram adicionadas no novo vetor, são visitadas aleatoriamente até que o tamanho desejado seja obtido.

Para exemplificar essa técnica de normalização, imagine que o vetor de características possua cem elementos. Deseja-se um vetor de características com onze elementos. Como o vetor possui mais elementos do que o tamanho desejado, a janela w é calculada como $100/10 = 10$. Ao se visitar o vetor de características as posições 10, 20, 30, 40, 50, 60, 70, 80, 90 e 100 são adicionadas no novo vetor. A Figura 3.6(b) exibe o vetor da Figura 3.5(b) com aplicação da normalização de tamanho para 10 elementos.

3.2.4 Algoritmo da Abordagem Convexa

O algoritmo da Abordagem Convexa, Algoritmo 11, com uma complexidade $O(N^2)$ começa com a minimização do modelo, exibido na linha 1, utilizando o algoritmo de Douglas-Peucker. Com o modelo minimizado, a escolha dos pontos significantes é realizada, linha 2. Essa escolha separa os pontos em pontos externos, aqueles que definem o modelo, obtidos pelo algoritmo de Sklankys, e pontos internos, aqueles que detalham o modelo. O vetor de características é definido, adicionando-se sempre a distância entre um par de pontos externos e algum ponto interno que esteja entre eles, como exibido entre as linhas 3 e 6. Uma normalização de amplitude é necessária, linha 9, de forma que a classificação seja invariante à escala. Por último, se necessário, uma normalização de tamanho do vetor é realizada, utilizando um algoritmo baseado em escolha por amostragem, linha 10.

Algorithm 11: Algoritmo para Extração de características com a Abordagem Convexa

```
1 List<pontos> pontos = MinimizarModelo(Imagem);
2 pontos = ExtrairPontosSignificantes(pontos);
3 List<double> distancias;
4 for Par de pontos externos em pontos do
5   | if Possui ponto interno then
6   | | distancias.add(pontosExternos,pontoInterno);
7   | end
8 end
9 distancias = NormalizarAmplitude(distancias);
10 distancias = NormalizarTamanho(distancias,TamanhoEscolhido);
11 return distancias
```

Capítulo 4

Experimentos e Resultados

4.1 Introdução

De modo a avaliar a técnica de extração de características proposta e a arquitetura implementada, uma série de testes são executados. Esses testes tem como finalidade comparar a Abordagem Convexa com outras duas técnicas de extração, o *Local Contour Sequence* (LCS) e o *Speed Up Robust Features Extraction* (SURF). Três técnicas de classificação são implementadas no módulo de classificação: Rede Neural de Elman (Elman RNN), Modelo Oculto de Markov (HMM) e um classificador baseado na comparação de sequências realizadas pelo *Dynamic Time Warping* (DTW).

Os testes são realizados com cada uma das técnicas de extração e classificação para gestos estáticos, dinâmicos e previsão de gestos. Uma base de gestos dinâmicos foi desenvolvida durante o trabalho da dissertação, e é utilizada durante todos os testes. As seções a seguir descrevem a base de gestos dinâmicos e os testes, assim como os resultados dos experimentos.

4.1.1 Organização do capítulo

Uma leitura sequencial de todas as sessões deste capítulo é indicada para um melhor entendimento. Este capítulo está estruturado da seguinte forma:

- Sessão 4.1 - Introdução: apresenta uma análise introdutória sobre os experimentos executados;
- Sessão 4.2 - Base de dados de gestos dinâmicos RPPDI: apresenta a base de gestos dinâmicos proposta por esta dissertação;
- Sessão 4.3 - Experimentos : exhibe a metodologia utilizada para todos os experimentos executados durante esta dissertação assim como seus resultados;



Figura 4.1: Ilustração dos sete gestos dinâmicos contidos na base de gestos dinâmicos RPPDI.

Tabela 4.1: Quantidade de sequências gravadas para cada gesto da base de gestos dinâmicos RPPDI

Gesto	Quantidade de sequências
Gesto 1	24
Gesto 2	24
Gesto 3	31
Gesto 4	18
Gesto 5	26
Gesto 6	33
Gesto 7	32

4.2 Base de dados de gestos dinâmicos RPPDI

A base de gestos dinâmicos RPPDI¹ é composta por sete gestos realizados pela mão de um usuário. Várias sequências para cada gesto foram gravadas, de forma que existem vários exemplos de execução de cada gesto. A Figura 4.1 exibe exemplos dos gestos utilizados.

Nota-se que várias posturas se repetem em gestos diferentes, de forma que seja possível testar o reconhecimento e previsão de gestos dinâmicos, ou seja, através de posturas que variam com o tempo. Os sete gestos refletem várias posições de mãos diferentes realizadas de acordo com um fluxo de tempo. Uma pulseira de cor preta está presente no pulso para facilitar a separação entre mão e pulso, o que ajuda na execução de alguns algoritmos de extração de características. Cada gesto possui um total de 14 *frames*, e para cada gesto uma certa quantidade de exemplos foram gravadas. Cada exemplo reflete a execução do gesto de uma maneira diferente, simulando a variação na execução do gesto por pessoas diferentes. A Tabela 4.1 exibe a quantidade de sequências gravadas para cada gesto.

¹Disponível em <http://rppdi.ecomp.poli.br/gesture/database/>

Todos os gestos foram gravados utilizando a mesma câmera, com $5MP$ de resolução. Depois de gravados, os gestos foram divididos em imagens, *frames*, sendo cada gesto é formado por 14 *frames* com um tamanho de 640×480 *pixels*. Isso permite que a base de gestos possa ser utilizada tanto para o reconhecimento de gestos dinâmicos, onde todos os 14 *frames* são passados como uma sequência para os classificadores, como para a previsão de gestos, onde só o início de cada gesto é enviado para ser previsto. Isso também facilita nos testes para o reconhecimento de gestos estáticos, já que cada sequência de posições de mão pode ser separada em posições fixas a serem classificadas.

4.3 Experimentos

De forma a avaliar a eficiência e eficácia da técnica de extração de características proposta, uma série de testes utilizando a arquitetura proposta é realizada. Esses testes procuram comparar os resultados obtidos pela Abordagem Convexa com duas outras técnicas, o LCS e o SURF. Os testes são realizados em três diferentes experimentos: reconhecimento de gestos estáticos, reconhecimento de gestos dinâmicos e previsão de gestos.

A Abordagem Convexa utiliza o contorno da mão para extrair as informações que melhor descrevem a postura. De forma a obter resultados mais consistentes, a extração do contorno para a Abordagem Convexa é realizada de duas maneiras diferentes. A primeira utilizando uma série de técnicas de processamento de imagens, as mesmas utilizadas para gerar o LCS, compostas pelo contorno da mão. A segunda utiliza os pontos de interesse extraídos pelo SURF, formando um contorno da postura da mão com esses pontos. A aplicação da Abordagem Convexa nessas duas técnicas é denominada, respectivamente de CLCS e CSURF.

Os testes são realizados utilizando a arquitetura proposta no Capítulo 3. O módulo de extração de características é implementado com quatro diferentes técnicas, o LCS, o SURF, o CLCS e o CSURF. As características extraídas por cada técnica são encaminhadas para o módulo de classificação, que implementa três técnicas de classificação diferentes: HMM, DTW e Elman RNN.

4.3.1 Experimento 1: Reconhecimento de gestos estáticos

O primeiro experimento, denominado Reconhecimento de gestos estáticos, tem como finalidade a comparação de taxa de classificação, tempo de execução e tempo de treino para cada par de algoritmos testados. O experimento utiliza posturas de mão extraídas da base de gestos dinâmicos RPPDI e separa-as em oito classes. A Figura 4.2 exhibe as oito posturas a serem reconhecidas. A Tabela 4.2 exhibe a quantidade de exemplos para cada postura.

Devido ao caráter estático desse experimento, visto que não existe evolução em uma linha de tempo nos gestos executados, a técnica de classificação HMM não é utilizada. O mesmo ocorre com a Elman RNN, que apresenta resultados melhores em padrões que evoluem com o tempo. Retirando a camada de recursão



Figura 4.2: Ilustração das oito posturas de gestos estáticos extraídos da base de gestos dinâmicos RPPDI.

Tabela 4.2: Quantidade de seqüências presentes para cada classe de posturas no experimento Reconhecimento de gestos estáticos.

Postura	Quantidade de amostras
Postura 1	156
Postura 2	193
Postura 3	129
Postura 4	100
Postura 5	100
Postura 6	118
Postura 7	117
Postura 8	55

da Elman RNN ela se torna uma *Multilayer Perceptron*(MLP) , que será utilizada nesse experimento. Nesse experimento o algoritmo utilizado para treinamento da MLP é o *Backpropagation*.

Os testes ocorrem em três diferentes divisões das seqüências de posturas: 1/3 das seqüências para treinamento e 2/3 para o teste, 1/2 para o treinamento e 1/2 para o teste e 2/3 para o treinamento e 1/3 para o teste. Cada teste é executado trinta vezes, e os resultados exibem a média dessas execuções. A configuração dos parâmetros para cada algoritmo foi escolhida de forma empírica, utilizando-se aproximação por tentativa e erro. A Tabela 4.3 exibe a melhor configuração para as técnicas de classificação.

Utilizando LCS no módulo de extração combinado com DTW no módulo de classificação, a melhor taxa de reconhecimento obtida foi de 78,52%, com uma divisão de 2/3 da base de gestos para treino. Com uma divisão de 1/2, a taxa

Tabela 4.3: Configurações utilizadas para cada par de algoritmos

Technique	Parameters	CLCS	CSURF	LCS	SURF
MLP	Neurons Input Layer	140	210	5600	1400
	Hidden Layers	12	14	75	37
	Neurons Output Layer	4	4	4	4
DTW	Features	140	210	5600	1400

Tabela 4.4: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o LCS no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	70,03	2,32	11,02	2,69
1/2 Treino	73,02	3,00	11,00	3,82
2/3 Treino	78,52	1,70	11,61	3,62

Tabela 4.5: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o SURF no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	57,16	2,32	120,42	52,00
1/2 Treino	63,01	4,08	130,40	53,38
2/3 Treino	70,72	7,10	190,43	50,56

obtida foi de 73,02%. Dividindo-se a base em 1/3 dos dados para treino, a taxa obtida foi de 70,03%. A Tabela 4.4 exibe o resultado encontrado.

A combinação SURF, no módulo de extração, e DTW no módulo de classificação obteve um resultado de 70,72% para uma divisão de 2/3 dos dados para treino. Uma divisão de 1/2 obteve um total de 63,01% e uma divisão de 1/3 obteve um resultado de 57,16% dos gestos classificados corretamente. A Tabela 4.5 exibe o resultado encontrado.

Utilizando-se o CSURF no módulo de extração e o DTW no módulo de classificação, o sistema conseguiu uma taxa de 82,53% dos gestos reconhecidos, para uma divisão de 2/3 dos dados para treinamento. Com uma divisão de 1/2, o resultado obtido foi de 78,46%. Uma divisão de 1/3 dos dados para treino obteve um resultado de 73,32% dos gestos reconhecidos corretamente. A Tabela 4.6 exibe o resultado encontrado.

A combinação CLCS, no módulo de extração, e DTW, no módulo de classificação, foi capaz de obter um resultado de 87,52% de gestos reconhecidos, com uma divisão de 2/3 dos dados para treino. Com uma divisão de 1/2 da base para treino, o resultado obtido foi de 83,01%. Uma divisão de 1/3 conseguiu uma taxa de 78,00%. A Tabela 4.7 exibe o resultado encontrado.

Combinando o LCS, no módulo de extração, com a MLP, no módulo de classificação resultou em uma taxa de reconhecimento de 73,52% para uma divisão de 2/3 dos dados para treino. Já uma divisão de 1/2 obteve um total de 65,02% de gestos reconhecidos. Uma divisão de 1/2, obteve um total de 62,03% de dados reconhecidos corretamente. A Tabela 4.8 exibe o resultado encontrado.

Utilizando uma MLP no módulo de classificação, e SURF no módulo de extra-

Tabela 4.6: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CSURF no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	73,32	1,82	4,09	1,09
1/2 Treino	78,46	6,08	4,21	0,99
2/3 Treino	82,53	4,01	6,33	1,23

Tabela 4.7: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CLCS no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	78,00	3,09	4,98	0,98
1/2 Treino	83,01	2,07	5,37	0,92
2/3 Treino	87,52	3,56	6,00	0,97

Tabela 4.8: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o LCS no módulo de extração e a MLP no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	62,03	2,33	20110,11	1,15
1/2 Treino	65,02	1,98	20545,32	1,00
2/3 Treino	73,52	1,24	20910,45	1,11

ção, uma taxa de 67,87% de gestos reconhecidos foi obtida, com uma divisão de 2/3 dos dados para treino. Dividindo os dados em 1/2 para treino e 1/2 para teste, um resultado de 60,19% foi obtido. Com uma divisão de 1/3 da base para treino um resultado de 55,12% foi obtida. A Tabela 4.9 exibe o resultado encontrado.

A utilização da combinação CSURF, no módulo de extração, e MLP no módulo de classificação conseguiu obter um resultado de 79,34% de reconhecimento para uma divisão de 2/3 da base para treino. Com uma divisão de 1/2 da base para treino, uma taxa de 73,09% foi obtida. Para uma divisão de 1/3, uma taxa de 69,81% foi obtida. A Tabela 4.10 exibe o resultado encontrado.

Utilizando CLCS no módulo de extração e MLP no módulo de classificação, com uma divisão de 2/3 da base para treino, a taxa de gestos classificados ficou em 80,65%. Já utilizando uma divisão de 1/2, obteve-se uma taxa de 74,03%. Uma divisão de 1/3 obteve uma taxa de 70,09%. A Tabela 4.11 exibe o resultado encontrado.

Os resultados obtidos mostram que a aplicação da Abordagem Convexa no LCS e no SURF, respectivamente CLCS e CSURF, obtiveram taxas de reconhecimento maiores do que o LCS e o SURF. O tempo de classificação e treino também foi menor para ambas as técnicas de classificação testadas, DTW e MLP.

4.3.2 Experimento 2: Reconhecimento de gestos dinâmicos

O segundo experimento, denominado Reconhecimento de gestos dinâmicos, tem a função de testar a Abordagem Convexa e a arquitetura proposta com gestos dinâmicos. Todos os sete gestos presentes na base de gestos dinâmicos RPPDI são

Tabela 4.9: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o SURF no módulo de extração e a MLP no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	55,12	4,97	23072,42	1,67
1/2 Treino	60,19	2,58	25398,40	1,43
2/3 Treino	67,87	3,12	29053,45	1,56

Tabela 4.10: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CSURF no módulo de extração e a MLP no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	69,81	5,87	13009,43	0,79
1/2 Treino	73,09	4,12	16523,12	0,70
2/3 Treino	79,34	3,91	15053,98	0,85

Tabela 4.11: Resultados obtidos para o reconhecimento de gestos estáticos utilizando o CLCS no módulo de extração e a MLP no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	70,09	3,09	10911,11	0,78
1/2 Treino	74,03	2,07	12032,61	0,87
2/3 Treino	80,65	3,56	13552,87	0,83

utilizados nesse experimento.

As técnicas de extração de características utilizadas nesse experimento são: LCS, SURF, CLCS e CSURF. Para o módulo de classificação, as técnicas implementadas são: Elman RNN, HMM e DTW.

As técnicas de extração são aplicadas em todos os *frames* de cada gesto, no módulo de extração, que são passados em lote, ou seja, todas as características extraídas dos 14 *frames*, para módulo de classificação.

Cada vetor de características, representando um *frame*, é passado na Elman RNN individualmente. A cada 14 *frames* passados, a rede apresenta uma classificação. O treinamento dessa rede se dá através do *Backpropagation* combinado com o *Simulated Annealing*, e ocorre utilizando uma janela de 14 *frames*.

Cada gesto presente na base de gestos é representado por um HMM diferente. Para se classificar um novo gesto, todos os 14 vetores de características são apresentados a cada HMM. O gesto representado pelo HMM que gerar a maior probabilidade de saída, dado o vetor de características apresentado, é escolhido como o gesto classificado.

No DTW, separa-se cada conjunto vetor de característica que representa os gestos em grupos, cada grupo representando um gesto. Um novo gesto é classificado comparando-o com todos os elementos de cada grupo. O grupo que obtiver a menor média de distâncias é escolhido como o gesto classificado.

Os testes ocorrem em três diferentes divisões das sequências de posturas: 1/3 das sequências para treinamento e 2/3 para a teste, 1/2 para o treinamento e 1/2 para a teste e 2/3 para o treinamento e 1/3 para a teste. Cada teste é executado trinta vezes, e os resultados exibem a média dessas execuções. A configuração dos parâmetros para cada algoritmo foi escolhida de forma empírica, utilizando-se aproximação por tentativa e erro. A Tabela 4.12 exhibe a melhor configuração para as técnicas de classificação, bem como a quantidade de características extraídas para cada sequência de gestos.

Utilizando a combinação do LCS, no módulo de extração, e o HMM, no módulo de classificação, a mais alta taxa de reconhecimento obtida foi 52,55 %. Essa taxa foi obtida utilizando a divisão de 2/3 da base de gestos para treino e 1/3 para

Tabela 4.12: Configuração de cada par de técnica de extração e classificação

Técnica	Parâmetros	CLCS	CSURF	LCS	SURF
Elman RNN	Neurônios camada entrada	140	210	5600	1400
	Camadas Escondidas	12	14	75	37
	Neurônios camada de saída	4	4	4	4
HMM	Estados	3	3	3	3
	Iterações Baum-Welch	100	100	10	10
	Características	10	10	5600	1400
DTW	Características	140	210	5600	1400

Tabela 4.13: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o HMM no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	27,48	7,55	496,52	3,07
1/2 Treino	39,29	6,33	636,07	3,11
2/3 Treino	52,55	5,70	772,41	3,07

teste. Com uma divisão de 1/3 dos dados para treino, a taxa obtida foi de 27,48% . com a divisão de 1/2 dos dados para treino a taxa foi de 39,29%. Observa-se também que o tempo de classificação médio é praticamente o mesmo para todas as divisões, o que não acontece com o tempo de treino, que aumenta de acordo com o tamanho da base de treino. A Tabela 4.13 exhibe o resultado obtido por cada divisão, bem como o desvio padrão e o tempo médio para treino e classificação de cada padrão.

Utilizando a combinação entre SURF, no módulo de extração, e HMM no módulo de classificação, os resultados obtidos foram aleatórios. Como a quantidade de características extraídas pelo SURF para cada gesto é muito alta, 1400 características, como exibido na Tabela 4.12, e cada característica do SURF é composta por 64 dimensões, o HMM não consegue convergir em nenhuma divisão da base de treino. A Tabela 4.14 descreve o resultado obtido.

A combinação entre CLCS, no módulo de extração, e HMM, no módulo de classificação, mostra resultados superiores, chegando a uma taxa de reconhecimento de 81,66% dos gestos para divisão de base de dados em 2/3 para treino e 1/3 para teste. Já com a divisão 1/2 para treino e 1/2 para teste, a taxa atingida foi de 76,70%. Utilizando a divisão 1/3 da base para treino e 2/3 para teste, a taxa encontrada foi de 68,13%. Observa-se uma evolução quando da aplicação da Abordagem Convexa no contorno extraído pelo CLCS, quando comparado ao

Tabela 4.14: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e o HMM no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	12,59	0,0	1221,13	7,58
1/2 Treino	12,63	0,0	1601,37	7,79
2/3 Treino	12,50	0,0	1902,83	7,77

Tabela 4.15: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e o HMM no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	68,13	4,87	124,15	0,769
1/2 Treino	76,70	5,24	159,06	0,77
2/3 Treino	81,66	5,41	193,99	0,79

Tabela 4.16: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e o HMM no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	54,40	5,05	192,08	1,19
1/2 Treino	69,89	4,50	243,72	1,17
2/3 Treino	77,44	4,81	299,65	1,19

resultado obtido pelo LCS. Essa evolução aparece também na taxa de classificação. O tempo gasto para classificação de cada padrão apresenta uma diminuição em relação ao tempo encontrado com a utilização do LCS. A Tabela 4.15 descreve esses resultados.

A utilização da Abordagem Convexa no contorno formado pelos pontos extraídos pelo SURF, CSURF, obteve resultados parecidos com os resultados do CLCS. Uma taxa de reconhecimento de 77,44% foi obtida para a divisão de 2/3 da base para treino e 1/3 para teste. Já com a divisão 1/2 para treino e 1/2 para teste, a taxa atingida foi de 69,89%. Utilizando a divisão 1/3 da base para treino e 2/3 para teste, a taxa encontrada foi de 54,40 %. A Tabela 4.16 descreve esses resultados.

Utilizando LCS no módulo de extração e DTW no módulo de classificação, as taxas de classificação obtidas foram mais altas do que as obtidos com HMM no módulo de classificação. Para uma divisão de 2/3 da base de dados para treino, uma taxa de 89,06 % foi obtida. Já com uma divisão de 1/2 da base de dados para treino, uma taxa de 83,59 % foi obtida. Com uma divisão de 1/3 da base de dados para treino, uma taxa de 79,47 % foi obtida. A Tabela 4.17 descreve o resultado.

O SURF, como método de extração de características, obteve resultados baixos, devido à quantidade de características extraídas para o reconhecimento de gestos dinâmicos. Utilizando o DTW no módulo de classificação, o resultado não foi diferente. Utilizando uma divisão da base de dados de 1/3 para treino, o DTW não conseguiu generalizar um reconhecimento, e a taxa foi de apenas 12,5%. Já para uma divisão de 1/2, o resultado sobe um pouco, para 25,3%. Para uma divisão de 2/3 para a base de treino, o resultado foi de 37,2%. A Tabela 4.18

Tabela 4.17: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	79,47	4,19	2820,32	792,33
1/2 Treino	83,59	4,89	2840,91	1003,49
2/3 Treino	89,06	4,88	2880,04	1237,47

Tabela 4.18: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	12,5	1,0	3590,05	1589,02
1/2 Treino	25,3	1,5	3750,35	1370,07
2/3 Treino	37,2	2,87	3710,079	1799,58

Tabela 4.19: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	85,30	4,10	252,61	75,19
1/2 Treino	87,99	3,63	250,53	73,25
2/3 Treino	94,08	3,20	257,09	79,75

descreve os resultados encontrados.

Utilizando a Abordagem Convexa aplicada aos pontos interesse do SURF, o CSURF, no módulo de extração e o DTW no módulo de classificação, os resultados obtidos foram melhores do que utilizando os descritores do SURF. Para uma divisão de 2/3 da base de dados para treino, uma taxa de 85,30 % de gestos reconhecidos foi obtida. Já com uma divisão de 1/2 da base de dados para treino, uma taxa de 87,99 % foi obtida. Com uma divisão de 1/3 da base de dados para treino, uma taxa de 94,08 % foi obtida. A Tabela 4.19 descreve o resultado.

As mais altas taxas de classificação foram encontradas utilizando o CLCS no módulo de extração e o DTW no módulo de classificação. Para uma divisão de 1/3 da base de dados para treino, uma taxa de 93,70% foi obtida. Para uma divisão de 1/2, uma taxa de 95,60% foi obtida e para uma divisão de 2/3, uma taxa de 97,00% encontrada. A Tabela 4.20 detalha os resultados obtidos.

Utilizando LCS, no módulo de extração, e Elman RNN no módulo de classificação, uma taxa de 51,0% de gestos reconhecidos foi obtida, utilizando uma divisão de 1/3 dos dados para treino. Com uma divisão de 1/2 para treino, a taxa obtida foi de 57,51%. Utilizando uma divisão de 2/3 dos gestos para o treino, foi possível obter uma taxa de 63,32%. A Tabela 4.21 detalha os resultados obtidos.

Com o SURF no módulo de extração e Elman RNN no módulo de classificação, os resultados continuaram baixos, como ocorreu com o uso do DTW. Para uma divisão de 1/3 dos dados para treino, a taxa de reconhecimento foi de 17,18%. Com uma divisão de 1/2, a taxa obtida foi de 27,98%. Para uma divisão de 2/3, a taxa obtida foi de 36,45%. A Tabela 4.22 detalha os resultados obtidos.

Com o uso das técnicas CSURF no módulo de extração e Elman RNN no

Tabela 4.20: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e o DTW no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	93,70	1,47	243,78	65,13
1/2 Treino	95,60	2,70	246,90	62,97
2/3 Treino	97,00	2,70	244,85	63,37

Tabela 4.21: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o LCS no módulo de extração e o Elman RNN no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	51,0	5,32	189872,89	1,41
1/2 Treino	57,51	6,12	201212,12	1,00
2/3 Treino	63,32	2,32	262121,33	1,23

Tabela 4.22: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o SURF no módulo de extração e a Elman RNN no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	17,18	2,36	439144,43	1,44
1/2 Treino	27,98	1,09	471212,12	1,43
2/3 Treino	36,45	4,21	502122,92	1,37

módulo de classificação os resultados obtidos conseguiram atingir uma taxa de 76,53% de gestos reconhecidos para uma divisão de 2/3 dos dados para treino. Com uma divisão de 1/2, a taxa obtida foi de 71,05%. Para uma divisão de 2/3, a taxa obtida foi de 63,91%. A Tabela 4.23 detalha os resultados obtidos.

Utilizando CLCS no módulo de extração, e Elman RNN no módulo de classificação, a melhor taxa de classificação obtida foi de 80,00%, com uma divisão de 2/3 dos dados para o treino. Já com uma divisão de 1/2, a taxa obtida foi de 75,12%. Para uma divisão de 1/3, a taxa obtida foi de 69,98%. A Tabela 4.24 detalha os resultados obtidos.

A aplicação da Abordagem Convexa nas técnicas LCS e SURF, formando respectivamente o CLCS e o CSURF, gerou taxas de reconhecimento maiores do que quando não aplicada. Os tempos para treinamento e classificação obtidos foram menores, em todas as técnicas de classificação. Esse resultado demonstra que a redução do vetor de características, proposto pelo algoritmo da Abordagem Convexa, é eficaz e eficiente, no domínio aplicado.

Para a classificação, todos os resultados obtidos com o CLCS e o CSURF foram maiores que os obtidos pelo LCS e o SURF. A Figura 4.3 exibe um resumo das classificações obtidas. Isso acontece pelo fato das técnicas com a Abordagem Convexa possuírem menos características, já que a aplicação do algoritmo seleciona somente as distâncias que influenciam na representação da forma da mão. A taxa obtida pelo SURF foi menor do que a das outras técnicas de extração, o que é causado pela quantidade elevada de características extraídas de cada *frame* que compõe um gesto.

O CLCS foi a técnica que obteve maiores resultados, em todas as técnicas.

Tabela 4.23: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CSURF no módulo de extração e a Elman RNN no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	63,91	2,38	12921,32	1,21
1/2 Treino	71,05	1,59	15923,23	1,00
2/3 Treino	76,53	1,33	17233,73	0,91

Tabela 4.24: Resultados obtidos para o reconhecimento de gestos dinâmicos utilizando o CLCS no módulo de extração e a Elman RNN no módulo de classificação

Divisão Dados	Class. Correta(%)	Desv. pad.	Tempo treino(ms)	Tempo class.(ms)
1/3 Treino	69,98	1,24	10854,31	0,99
1/2 Treino	75,12	5,79	16847,82	1,23
2/3 Treino	80,00	2,79	18592,16	0,92

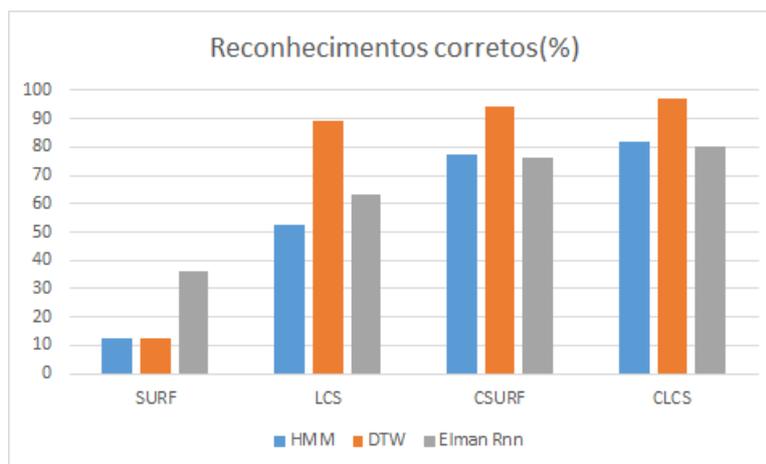
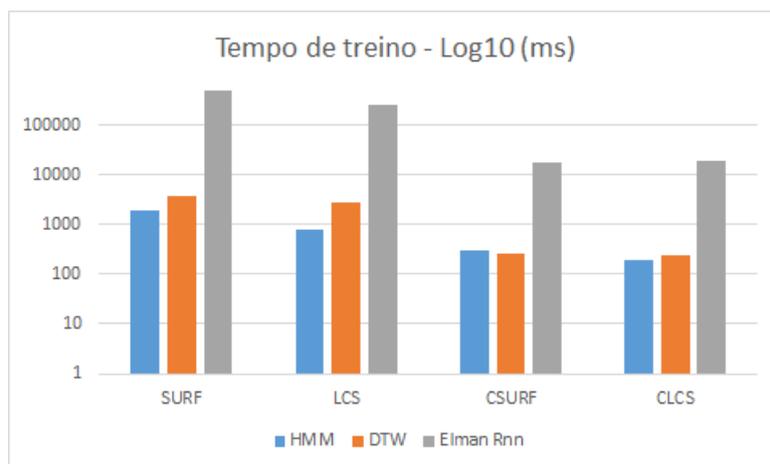


Figura 4.3: Gráfico exibindo os resultados de classificação para todas as técnicas testadas.

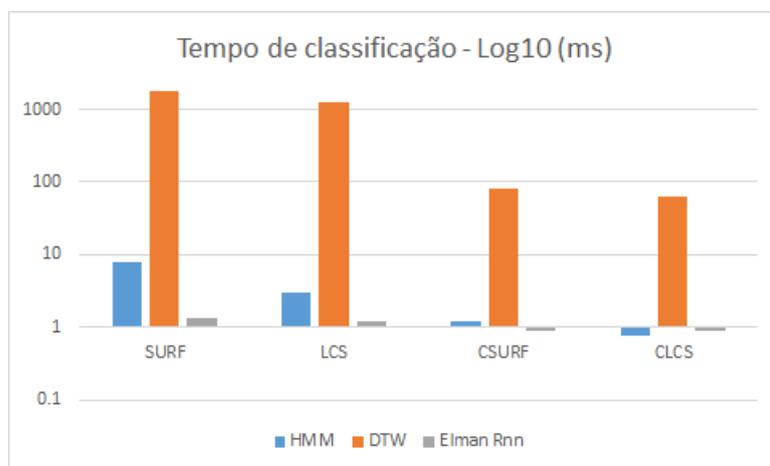
De todas as combinações, aquelas realizadas com o DTW obtiveram as melhores taxas. Isso ocorre, devido à capacidade do DTW comparar sequências distintas em tamanho e em velocidade, o que o torna uma poderosa técnica de comparação para gestos dinâmicos.

Levando em consideração o tempo de treinamento e de classificação de um padrão, HMM é a técnica que possui melhores resultados. A Figura 4.4(a) exibe um gráfico mostrando a comparação entre os tempos para treinamento de todas as combinações. Já a Figura 4.4(b) exibe o gráfico comparando o tempo de classificação de um gesto para todas as combinações de técnicas.

Os gráficos exibidos na Figura 4.4 estão em escala logarítmica de base 10 de forma a melhorar a percepção das diferenças entre os valores. O DTW, apesar de obter uma taxa de classificação melhor, apresenta tempos elevados para a classificação dos gestos. Isso ocorre por que o DTW calcula a distância ponto-a-ponto de cada elemento do vetor de características, culminando em um custo diretamente proporcional ao tamanho do vetor. Isso fica claro ao observar que o tempo de classificação do DTW com o SURF, que possui muitos elementos no seu vetor de características, é muito elevado em comparação à outras técnicas. Os tempos de treino da Elman RNN são os mais elevados devido à própria natureza do treinamento de uma rede neural recorrente, que precisa executar várias iterações até que o erro da rede diminua à um valor previamente definido.



(a) Gráfico exibindo o tempo de treinamento de todas as combinações de técnicas, exibidos em escala logarítmica na base 10.



(b) Gráfico exibindo o tempo de classificação de um padrão para todas as combinações de técnicas, exibidos em escala logarítmica na base 10.

Figura 4.4: Gráficos exibindo os tempos de treinamento e classificação para todas as combinações de técnicas.

Tabela 4.25: Resultado da previsão adicionando-se os quatro primeiro *frames* no vetor de características para o CLCS.

Técnica Classificação	Resultado	F1	F2	F3	F4
HMM	Taxa predição(%)	37,9	43,6	49,9	54,4
HMM	Desvio padrão	6,5	5,6	4,1	5,3
HMM	Tempo de predição(ms)	0,10	0,17	0,26	0,34
DTW	Taxa predição(ms)(%)	17,18	17,42	17,06	17,34
DTW	Desvio padrão	0,0	0,27	0,70	0,47
DTW	Tempo de predição(ms)	9,7	9,5	29,0	40,1

4.3.3 Experimento 3: Previsão de gestos dinâmicos

No último experimento, a capacidade da arquitetura de prever gestos por extrapolação é avaliada. A previsão de gestos é definida como uma generalização de gestos incompletos, que pode ajudar no reconhecimento de gestos em tempo real.

Para esse experimento, as quatro técnicas de extração de características são aplicadas no módulo de extração: CLCS, CSURF, LCS e HMM. Para o módulo de classificação, são utilizadas: DTW e HMM. A Elman RNN não foi documentada nesse experimento pois os resultados apresentados não foram suficientes para a previsão dos gestos. Os testes foram executados somente com a divisão de 2/3 dos dados para treino, pois o objetivo do teste é a comparação entre as técnicas na tarefa de previsão de gesto.

O processo de previsão se dá quando um gesto incompleto é classificado pelo módulo de classificação. Para tanto, as técnicas de classificação são treinadas utilizando-se toda a base de gestos dinâmicos. A diferença ocorre na hora da classificação, onde, para cada teste, são passados gestos incompletos, ou seja, com menos *frames* do que um gesto completo. Os testes foram executados com a mesma configuração do experimento anterior, porém, são executados para intervalos de *frames*. Cada teste é iniciado apenas com 1 *frame*, o *frame* inicial, e a cada iteração um novo *frame* é adicionado ao intervalo, até que todos os 14 *frames* sejam adicionados.

Nesse experimento os resultados foram coletados para cada novo *frame* adicionado no vetor de características. Esse procedimento torna possível observar a evolução da taxa de predição e do tempo necessário para classificação de cada novo *frame* adicionado.

A Tabela 4.25 exhibe os resultados obtidos utilizando a técnica CLCS no módulo de extração, com os *frames* de 1 a 4 no vetor de características. Nesse resultado, o HMM demonstra uma evolução contínua na previsão, chegando a 54,4% de gestos previstos, diferente do DTW, que não é capaz de prever o gesto com tão poucos *frames*, resultando em uma previsão completamente aleatória.

A Tabela 4.26 exhibe os resultados da aplicação do CLCS com a adição dos *frames* 5 à 8 no vetor de características. A taxa de previsão com HMM continua aumentando, passando de 66,0% de gestos previstos com 5 *frames*, para 75,98% com 8 *frames*. A partir da adição do sexto *frame*, o DTW começa a apresentar um aumento na taxa de previsão, em uma razão de crescimento maior do que a

Tabela 4.26: Resultado da previsão adicionando-se os *frames* cinco à oito no vetor de características para o CLCS.

Técnica Classificação	Resultado	F5	F6	F7	F8
HMM	Taxa predição(%)	66,0	68,38	73,12	75,98
HMM	Desvio padrão	5,5	4,8	6,0	4,7
HMM	Tempo de predição(ms)	0,42	0,50	0,59	0,67
DTW	Taxa predição(%)	17,29	19,16	23,43	35,20
DTW	Desvio padrão	0,3	2,0	3,4	4,5
DTW	Tempo de predição(ms)	48,3	59,0	68,8	80,6

Tabela 4.27: Resultado da previsão adicionando-se os *frames* de nove à doze no vetor de características para o CLCS.

Técnica Classificação	Resultado	F9	F10	F11	F12
HMM	Taxa predição(%)	76,7	80,0	82,2	81,7
HMM	Desvio padrão	5,2	4,6	6,0	5,8
HMM	Tempo de predição(ms)	0,76	0,83	0,92	1,00
DTW	Taxa predição(ms)(%)	49,58	64,06	76,77	94,84
DTW	Desvio padrão	4,9	5,3	6,8	4,0
DTW	Tempo de predição(ms)	90,23	98,06	104,87	80,6

do HMM, chegando à 35,20% dos gestos previstos. Nota-se também uma curva de aumento com grande inclinação no tempo de predição gasto pelo DTW, o que não ocorre com o HMM, onde o aumento do tempo de previsão é suave.

Os resultados obtidos com a adição dos *frames* de nove à doze no vetor de características utilizado como entrada no módulo de extração com CLCS são exibidos na Tabela 4.27, Os resultados exibem uma estabilização na taxa de previsão obtida para o HMM, chegando a um patamar próximo de 80,00% de previsão. Essa taxa é obtida com a adição do décimo *frame*. O DTW continua apresentando uma taxa de crescimento na previsão, partindo de 49,58% com a adição do *frame* nove e chegando até 94,84% com a adição do *frame* doze.

Os resultados após a adição dos dois últimos *frames*, ao vetor de características utilizado no módulo de extração utilizando o CLCS, são exibidos na Tabela 4.28. Esses resultados confirmam a estabilização do HMM na previsão dos gestos. A taxa de previsão continua a mesma dos resultados da Tabela 4.27, próximo a 80% de gestos previstos corretamente, mesmo com a adição de todos os *frames*. O DTW também atinge uma estabilização, elevando um pouco a taxa de reconhecimento

Tabela 4.28: Resultado da previsão adicionando-se os *frames* treze e catorze no vetor de características para o CLCS.

Técnica Classificação	Resultado	F13	F14
HMM	Taxa predição(%)	80,15	80,72
HMM	Desvio padrão	5,9	5,4
HMM	Tempo de predição(ms)	1,09	1,18
DTW	Taxa predição(%)	96,76	97,00
DTW	Desvio padrão	1,14	1,11
DTW	Tempo de predição(ms)	115,34	123,04

Tabela 4.29: Resultado da previsão adicionando-se os quatro primeiro *frames* no vetor de características para o CSURF.

Técnica Classificação	Resultado	F1	F2	F3	F4
HMM	Taxa predição(%)	35,57	41,87	50,15	55,52
HMM	Desvio padrão	4,6	4,4	5,9	5,9
HMM	Tempo de predição(ms)	0,09	0,18	0,26	0,35
DTW	Taxa predição(ms)(%)	16,53	17,18	17,18	20,31
DTW	Desvio padrão	0,0	0,0	0,0	6,98
DTW	Tempo de predição(ms)	11,37	46,22	68,65	91,26

Tabela 4.30: Resultado da previsão adicionando-se os *frames* cinco à oito no vetor de características para o CSURF.

Técnica Classificação	Resultado	F5	F6	F7	F8
HMM	Taxa predição(%)	62,91	67,65	72,23	73,69
HMM	Desvio padrão	5,56	5,64	4,60	5,64
HMM	Tempo de predição(ms)	0,44	0,51	0,61	0,69
DTW	Taxa predição(%)	18,43	24,37	39,37	63,12
DTW	Desvio padrão	1,3	2,8	4,7	7,8
DTW	Tempo de predição(ms)	113,52	135,41	157,05	180,65

para 97,00 % com todos os *frames*.

Aplicando a técnica CSURF no módulo de extração, os resultados coletados foram similares aos obtidos com a aplicação do CLCS. Adicionando os quatro primeiros *frames* ao vetor de características, o HMM apresenta a mesma curva de evolução na taxa de previsão, iniciando de 35,57% de gestos previstos, até 55,52% com quatro *frames*. O DTW apresenta pouca evolução e resultados muito baixos, chegando a obter 20,31% dos gestos previstos com quatro *frames*. A Tabela 4.29 exibe esse resultado.

Os resultados obtidos ao adicionar os *frames* 5 à 8 no vetor de características, utilizando o CSURF, são exibidos na Tabela 4.30. Os resultados exibem um crescimento constante na taxa de previsão com o HMM. Já com o DTW, a previsão começa a subir rapidamente, a cada novo *frame* adicionado. Ao adicionar o *frame* 8, a taxa de previsão para o DTW é de 63,12% e a do HMM 73,69%. Como acontece com o CLCS, a taxa de crescimento do tempo gasto para a previsão no DTW também sobe rapidamente, diferente do que ocorre com o HMM.

O HMM começa a estabilizar quando os *frames* 9 à 12 são adicionados ao vetor

Tabela 4.31: Resultado da previsão adicionando-se os *frames* nove à doze no vetor de características para o CSURF.

Técnica Classificação	Resultado	F9	F10	F11	F12
HMM	Taxa predição(%)	72,13	75,41	74,84	77,76
HMM	Desvio padrão	5,64	4,6	5,6	5,7
HMM	Tempo de predição(ms)	0,78	0,86	0,95	1,04
DTW	Taxa predição(ms)(%)	80,31	85,31	88,12	92,81
DTW	Desvio padrão	5,6	2,3	4,0	5,4
DTW	Tempo de predição(ms)	202,29	225,76	262,84	267,46

Tabela 4.32: Resultado da previsão adicionando-se os *frames* treze e catorze no vetor de características para o CSURF.

Técnica	Classificação	Resultado	F13	F14
HMM		Taxa previsão(%)	76,25	77,44
HMM		Desvio padrão	5.3	4,8
HMM		Tempo de previsão(ms)	1.13	1.23
DTW		Taxa previsão(%)	93.12	94.08
DTW		Desvio padrão	1.3	2.0
DTW		Tempo de previsão(ms)	289.87	312.66

Tabela 4.33: Resultado da previsão adicionando-se os quatro primeiro *frames* no vetor de características para o LCS.

Técnica	Classificação	Resultado	F1	F2	F3	F4
HMM		Taxa previsão(%)	36,14	35,20	34,53	40,46
HMM		Desvio padrão	6,67	6,49	4,79	6,45
HMM		Tempo de previsão(ms)	0,27	0,48	0,72	0,94
DTW		Taxa previsão(ms)(%)	12,50	14,37	13,75	14,37
DTW		Desvio padrão	2,47	5,11	4,33	1,71
DTW		Tempo de previsão(ms)	89,39	179,87	294,65	356,69

de características, utilizando o CSURF. A Tabela 4.31 exibe esse resultado. O HMM atinge uma taxa de 77,76% de gestos previstos com 12 *frames*. O DTW continua aumentando sua taxa de previsão, partindo de 80,31% no *frame* número nove, e chegando até 92,81% no décimo segundo *frame*. A Tabela 4.31 exibe esse resultado.

A estabilização do HMM, assim como ocorre com a utilização do CLCS, é confirmada para o CSURF ao adicionar-se os dois últimos *frames* ao vetor de características. A Tabela 2.32 exibe o resultado. O DTW continua crescendo, dessa vez menos, atingindo 94,08% de previsão, com todos os *frames*.

Os últimos resultados obtidos foram com a utilização do LCS no módulo de extração. A Tabela 4.33 exibe os dados adicionando os quatro primeiros *frames*. Observa-se que, assim como CLCS e CSURF, o HMM apresenta uma evolução constante na taxa de previsão, começando de 36,14% e atingindo 40,46% no quarto *frame*. O DTW não apresenta evolução no início, obtendo resultados aleatórios.

Os resultados obtidos ao adicionar os *frames* cinco à 8 no vetor de características extraído pelo LCS mostram um avanço do HMM. Diferente dos resultados obtidos pelo CLCS e CSURF, a taxa com o LCS se mostra mais baixa, chegando à 52,23% de gestos previstos no oitavo *frame*. A partir do quarto *frame*, a taxa de previsão do DTW passa a subir, chegando a 65,00% no *frame* 8, já ultrapassando o resultado obtido pelo HMM. Isso ocorre devido ao fato de que com 8 *frames* o DTW já conseguir comparar com sucesso a sequência incompleta com a sequência completa. Pode ser observado também um alto tempo na previsão do gesto com o DTW, muito mais alto do que em qualquer outro resultado, chegando a 700ms no oitavo *frame*. A Tabela 4.34 detalha esse resultado.

Adicionando os *frames* nove à doze ao vetor de características extraído pelo LCS, percebe-se a estagnação no resultado da previsão de gestos obtidos pelo

Tabela 4.34: Resultado da previsão adicionando-se os *frames* cinco à oito no vetor de características para o LCS.

Técnica Classificação	Resultado	F5	F6	F7	F8
HMM	Taxa predição(%)	44,27	48,80	49,47	52,23
HMM	Desvio padrão	6,54	5,97	7,03	6,02
HMM	Tempo de predição(ms)	1,16	1,38	1,60	1,83
DTW	Taxa predição(ms)(%)	21,87	32,18	47,50	65,00
DTW	Desvio padrão	2,20	4,22	8,0	3,92
DTW	Tempo de predição(ms)	444,30	529,94	624,53	700,98

Tabela 4.35: Resultado da previsão adicionando-se os *frames* nove à doze no vetor de características para o LCS.

Técnica Classificação	Resultado	F9	F10	F11	F12
HMM	Taxa predição(%)	53,43	52,65	53,34	54,42
HMM	Desvio padrão	5,98	7,68	5,37	6,11
HMM	Tempo de predição(ms)	2,06	2,25	2,46	2,70
DTW	Taxa predição(ms)(%)	75,93	78,43	80,31	82,50
DTW	Desvio padrão	7,46	3,00	6,11	4,73
DTW	Tempo de predição(ms)	785,25	876,76	962,54	1060,92

HMM. A taxa de previsão chega a 54,42% no *frame* 12. O DTW obtém uma taxa de 82,50% no *frame* 12, com o tempo de classificação chegando a 1060,92 também no *frame* 12. A Tabela 4.35 exhibe esse resultado.

Ao adicionar os dois últimos *frames* ao vetor de características extraído pelo LCS, observa-se a estabilização do resultado em ambas as técnicas, o DTW atingindo 89,06% e o HMM 52,55%. A Tabela 4.36 detalha esse resultado.

Observando a taxa de classificação, percebe-se que o HMM apresenta uma subida constante, porém suave. Ao obter metade dos *frames* de um gesto completo, o HMM já está perto de uma taxa de previsão estabilizada. Já o DTW oferece melhores resultados após a inserção de metade dos *frames* no vetor de características. A Figura 4.5 ilustra essa tendência.

Quando o tempo de classificação de um padrão é observado, percebe-se como a previsão de gestos é importante. A Figura 4.6 exhibe a evolução do tempo de predição para todas as combinações. Percebe-se que o HMM possui um tempo de classificação baixo. Ao chegar à metade dos *frames* inseridos no vetor de características, o HMM já obtém uma taxa de previsão estabilizada, que permanece muito

Tabela 4.36: Resultado da previsão adicionando-se os *frames* treze e catorze no vetor de características para o LCS.

Técnica Classificação	Resultado	F13	F14
HMM	Taxa predição(%)	53,59	52,55
HMM	Desvio padrão	6,0	5,7
HMM	Tempo de predição(ms)	2,70	2,91
DTW	Taxa predição(%)	89,68	89,06
DTW	Desvio padrão	4,6	1,1
DTW	Tempo de predição(ms)	1131,02	1237,47

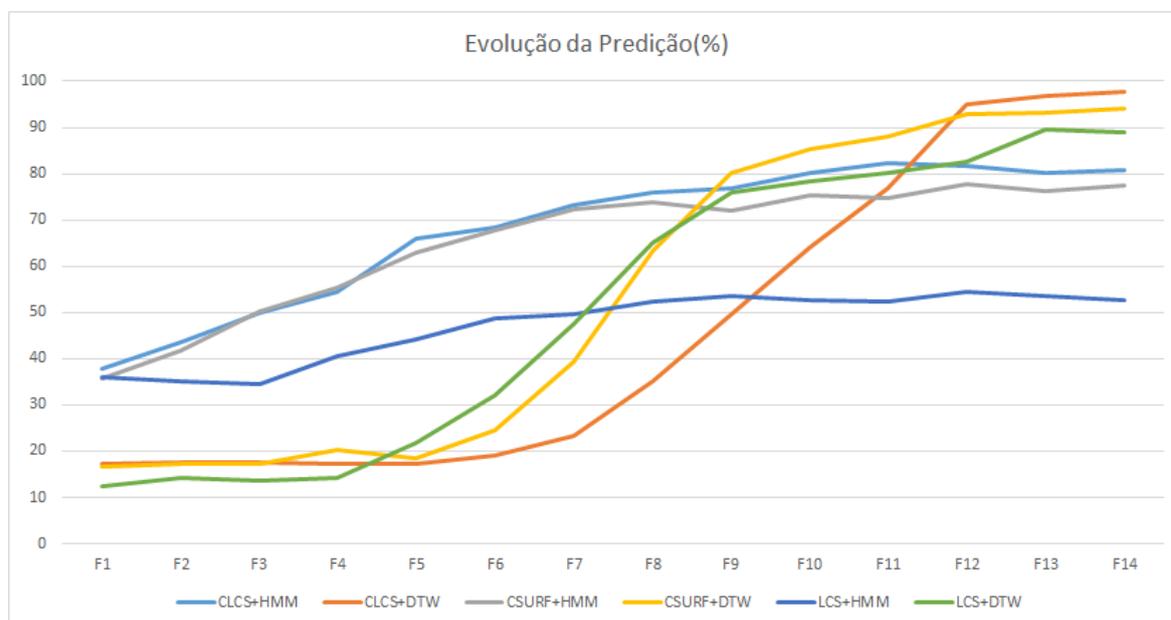
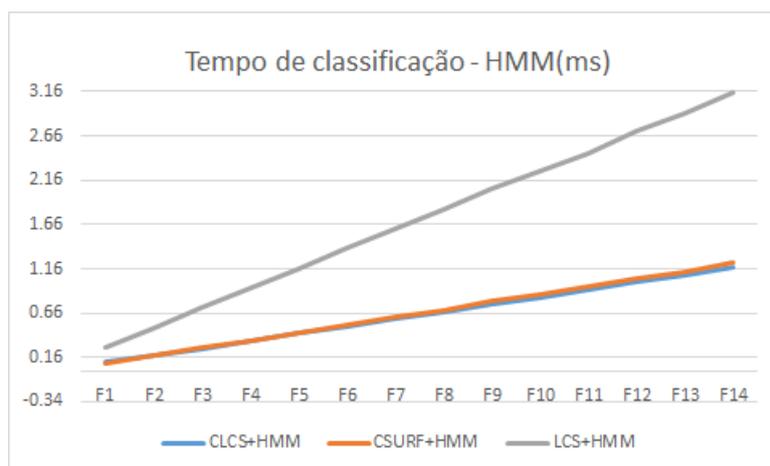
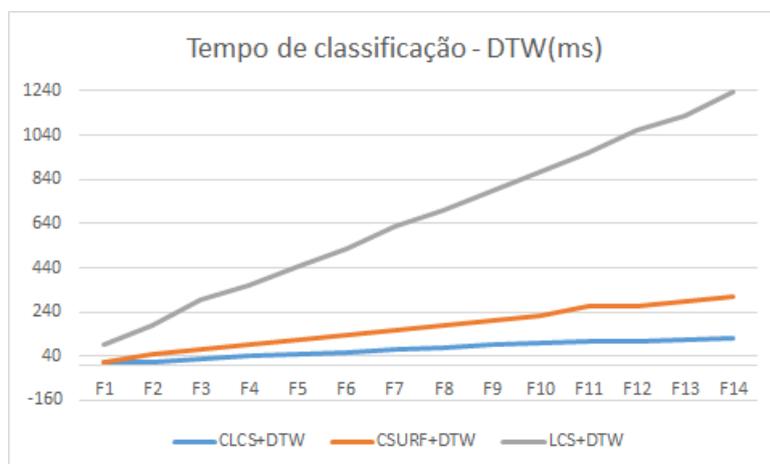


Figura 4.5: Gráfico exibindo a evolução da predição de gestos corretos para todas as combinações testadas.

similar à taxa encontrada com todos os *frames*. Apesar disso, possui um tempo de previsão de cada padrão menor, chegando a prever até dois gestos no mesmo tempo que se classificaria um gesto completo. O mesmo fenômeno pode ser observado no DTW, ainda de forma mais drástica, já que o número de elementos no vetor de características influencia diretamente no tempo de processamento desta técnica.



(a) Gráfico exibindo a evolução do tempo de classificação para previsão de gestos utilizando o HMM.



(b) Gráfico exibindo a evolução do tempo de classificação para previsão de gestos utilizando o DTW.

Figura 4.6: Gráficos de comparação do tempo de treinamento e classificação para todas as combinações de técnicas.

Capítulo 5

Conclusão e Trabalhos Futuros

5.1 Conclusão

Esse trabalho de dissertação descreve a proposta, implementação e testes de um sistema para reconhecimento e previsão de gestos dinâmicos realizados com a mão de um usuário. A área de reconhecimento de gestos dinâmicos, embora muito abordada por diversos trabalhos na última década, vem esbarrando em alguns problemas, como o processamento de imagens em um tempo hábil para ser utilizada em tempo real e uma base de gestos em que se possa realizar diversos *benchmarks*.

Para resolver o problema citado, uma técnica de extração de características é proposta e analisada. Essa técnica, denominada Abordagem Convexa, seleciona pontos no contorno da mão que podem representar todo o contorno. A escolha desses pontos se dá dinamicamente, sendo selecionados os pontos que melhor descrevem a forma da mão na imagem.

Ao se utilizar a Abordagem Convexa nos contornos obtidos pelas técnicas de extração *Local Contour Sequence* (LCS) e *Speed Up Robust Feature*(SURF), o vetor de característica extraído se mostra menor, o que reflete em uma boa taxa de reconhecimento, além de diminuir o tempo necessário para reconhecer gestos corretamente.

O sistema proposto possui dois módulos, um para extração de características e um outro para a classificação do gesto. De forma a realizar testes de validação no sistema foram implementadas quatro técnicas no módulo de extração e três no módulo de classificação. Para extração foram implementadas as técnicas LCS e SURF, bem como a aplicação da Abordagem Convexa nos contornos de mão extraídos por essas duas técnicas, gerando o CLCS e o CSURF respectivamente.

Devido a necessidade de uma base de dados que suportasse os experimentos, uma base de gestos dinâmicos foi desenvolvida, denominada Base de Gestos Dinâmicos RPPDI, e distribuída gratuitamente pela internet. Essa nova base de gestos dinâmicos contém sete gestos, com vários exemplos para cada gesto. Os gestos são realizados pela mão de um usuário sob um fundo escuro e são compostos por 14 imagens, representando uma evolução no tempo da postura da mão.

Foram realizados três experimentos, o primeiro denominado Reconhecimento

de gestos estáticos, o segundo Reconhecimento de gestos dinâmicos e o terceiro Previsão de gestos dinâmicos por extrapolação. No primeiro experimento, algumas poses de mão da base de gestos foram selecionadas, de forma a simular gestos estáticos. As quatro técnicas de extração foram utilizadas em dois classificadores: uma Rede Neural MLP e um classificador que utiliza o *Dynamic Time Warping* (DTW) para comparar sequências. No segundo experimento, os sete gestos completos da base de gestos são utilizados. No módulo de extração são utilizadas as quatro técnicas de extração citadas anteriormente. No módulo de extração são utilizadas as técnicas DTW, Modelo Oculto de Markov (HMM) e Rede Neural Recorrente de Elman (Elman RNN). No terceiro experimento, os sete gestos da base de gesto dinâmicos são utilizados. O sistema utiliza como entrada gestos incompletos, somente alguns *frames* de cada gesto. Vários testes são realizados, cada um contendo uma certa quantidade de *frames*, de forma a avaliar a capacidade de predição de gestos para cada combinação de técnica. No módulo de classificação, o terceiro experimento utiliza um DTW e um Modelo Oculto de Markov.

Os resultados obtidos demonstram que o CLCS e o CSURF, técnicas derivadas do LCS e SURF utilizando a Abordagem Convexa, se mostraram superiores em taxas de classificação e tempo de processamento aos das técnicas originalmente propostas. Esse fator se deve ao tamanho e aos elementos do vetor de característica extraído pela Abordagem Convexa. Como os pontos selecionados para a extração conseguem representar melhor a postura da mão, as taxas de classificação em todos os classificadores testados foram mais elevadas. Dentre os classificadores o DTW é aquele que apresenta a melhor taxa de reconhecimento, tendo como desvantagem um maior tempo necessário para a classificação dos padrões. O HMM foi a técnica que conseguiu os melhores tempos para a classificação.

Os resultados obtidos no terceiro experimento, previsão, demonstram que o uso da Abordagem Convexa pode ser utilizada para o reconhecimento de padrões incompletos em conjunto com o DTW e o HMM. Os valores obtidos deixam claro que a previsão de gestos pode ser utilizada para agilizar o trabalho de reconhecimento, já que com metade dos *frames* capturados a quantidade de gestos reconhecidos corretamente tende a estabilizar, quando se usa o HMM, enquanto o tempo para a classificação é metade do que o necessário para classificar um gesto completo.

5.2 Contribuições

Como contribuições desta dissertação podemos citar o sistema proposto para o reconhecimento e previsão de gestos, a Abordagem Convexa e a base de gestos dinâmicos RPPDI. O sistema proposto se mostrou robusto e flexível para o reconhecimento de gestos estáticos e dinâmicos, bem como para o reconhecimento de gestos incompletos, previsão de gestos.

A Abordagem Convexa é uma técnica de extração de características para gestos dinâmicos que consegue descrever um gesto com uma quantidade reduzida de elementos. Essa técnica se mostrou eficaz, pois consegue atingir taxas de classificação melhores do que técnicas utilizadas em reconhecimento de gestos, LCS e

SURF, e eficiente pois consegue a classificação em tempos menores.

A base de gestos dinâmicos RPPDI possui sete gestos dinâmicos, compostos por 14 *frames* contendo posturas de mão que variam com o tempo. Essa base de gestos possui várias amostras para cada gesto, trazendo diversidade para os gestos. Isso garante que um mesmo gesto seja executado de várias formas diferentes.

Durante o desenvolvimento deste trabalho de dissertação, alguns artigos e resumos foram publicados utilizando os resultados obtidos. Seguem os trabalhos publicados:

Evento: 28th Symposium On Applied Computing.

Título: Convexity Local Contour Sequences for Gesture Recognition.

Evento: 23rd International Conference on Artificial Neural Networks.

Título: An Effective Dynamic Gesture Recognition System based on the Feature Vector Reduction for SURF and LCS.

Evento: French-Brazilian Workshop on Numerical and Symbolic Methods of Data Analysis 2013.

Título: Using a Dynamic Time Wrapper to Recognize Dynamic Gestures.

Evento: 1st BRICS Countries Congress.

Título: A Dynamic Gesture Prediction System Based on The CLCS Feature Extraction.

Além dos trabalhos citados, uma publicação para o *Journal Expert Systems With Applications*, da *Elsevier*, está sendo compilada. Uma evolução deste trabalho será executada como forma do projeto de PHD intitulado "*Using Computer Vision to provide Real-Time Sign Language Interaction with a Robot.*" a ser realizado no laboratório *Knowledge Technoly Group*, na Universidade de Hamburgo - Alemanha.

5.3 Trabalhos Futuros

Esse trabalho pode ser estendido em várias linhas de pesquisa, como por exemplo:

- Expandir a base de gestos proposta, capturando mais gestos;
- Criar uma base de sinais de LIBRAS, a Língua Brasileira de Sinais;
- Implementar métodos de extração de contorno de mão em fundos complexos;
- Implementar uma abordagem mista, utilizando a previsão de gestos de forma à auxiliar o reconhecimento;
- Realizar testes com gestos obtidos em tempo real;

- Incrementar os tipos de características extraídas, além do contorno, como características biométricas.
- Implementar as técnicas em um robô humanoide para fins de comunicação homem-robô.
- Testar a Abordagem Convexa para a descrição e reconhecimento de outros objetos, além de mãos.

Referências Bibliográficas

- [AAC⁺91] S. Amato, B. Apolloni, G. Caporali, U. Madiesani, and A. Zanaboni. Simulated annealing approach in backpropagation. *Neurocomputing*, 3(5–6):207 – 220, 1991.
- [AC99] J. K. Aggarwal and Q. Cai. Human motion analysis: a review. *Comput. Vis. Image Underst.*, 73(3):428–440, March 1999.
- [AH08] Ienkaran Arasaratnam and Simon Haykin. Nonlinear bayesian filters for training recurrent neural networks. In *MICAI’08*, pages 12–33, 2008.
- [AL06] M. Ahmad and Seong-Whan Lee. Hmm-based human action recognition using multiview image sequences. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 263–266, 2006.
- [ASKH12] Y. Araga, M. Shirabayashi, K. Kaida, and H. Hikawa. Real time gesture recognition system using posture classifier and jordan recurrent neural network. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8, 2012.
- [ATK11] Y.F.G. Abdul, F.W.H. Tze, and K.T.T. Kin. Feature extraction from 2d gesture trajectory in malaysian sign language recognition. In *Mechatronics (ICOM), 2011 4th International Conference On*, pages 1–6, may 2011.
- [BAESS11] S. Bilal, R. Akmeliawati, M.J. El Salami, and A.A. Shafie. Vision-based hand posture detection and recognition for sign language x2014; a study. In *Mechatronics (ICOM), 2011 4th International Conference On*, pages 1–6, 2011.
- [BDH13] S. Bodiroza, G. Doisy, and V.V. Hafner. Position-invariant, real-time gesture recognition based on dynamic time warping. In *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*, pages 87–88, 2013.
- [BETVG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

- [BJB⁺13a] Pablo V. A. Barros, Nestor T. M. Junior, Juvenal M. M. Bisneto, Bruno J. T. Fernandes, Byron L. D. Bezerra, and Sergio M. M. Fernandes. Convexity local contour sequences for gesture recognition. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13*, pages 34–39, New York, NY, USA, 2013. ACM.
- [BJB⁺13b] Pablo V. A. Barros, Nestor T. M. Junior, Juvenal M. M. Bisneto, Bruno J. T. Fernandes, Byron L. D. Bezerra, and Sergio M. M. Fernandes. An effective dynamic gesture recognition system based on the feature vector reduction for surf and lcs. In *Proceedings of the 23rd International Conference on Artificial Neural Networks, ICANN '13*, New York, NY, USA, 2013. Elsevier.
- [BNT09] J. Bernardes, R. Nakamura, and R. Tori. Design and implementation of a flexible hand gesture command interface for games based on computer vision. In *Games and Digital Entertainment (SBGAMES), 2009 VIII Brazilian Symposium on*, pages 64–73, 2009.
- [BPSW70] Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.
- [CC02] A. Corradini and P.R. Cohen. Multimodal speech-gesture interface for handfree painting on a virtual paper using partial recurrent neural networks as gesture recognizer. In *Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on*, volume 3, pages 2293–2298, 2002.
- [CH11] F.M. Ciaramello and S.S. Hemami. A computational intelligibility model for assessment and compression of american sign language video. *Image Processing, IEEE Transactions on*, 20(11):3014–3027, 2011.
- [COB⁺04] Sung-Jung Cho, Jong-Koo Oh, Won-Chul Bang, Wook Chang, Eun-seok Choi, Yang Jing, Joonkee Cho, and Dong-Yoon Kim. Magic wand: a hand-drawn gesture input device in 3-d space with inertial sensors. In *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*, pages 106–111, 2004.
- [CR11] D. Chivers and P. Rodgers. Gesture-based input for drawing schematics on a mobile device. In *Information Visualisation (IV), 2011 15th International Conference on*, pages 127–134, 2011.
- [Cro84] Franklin C. Crow. Summed-area tables for texture mapping. *SIG-GRAPH Comput. Graph.*, 18(3):207–212, January 1984.

- [CXC⁺11] Yitao Chi, Zhang Xiong, Qing Chang, Chao Li, and Hao Sheng. Improving hessian matrix detector for surf. *IEICE Transactions*, 94-D(4):921–925, 2011.
- [DdM10] Evgueni Dodonov and Rodrigo Fernandes de Mello. A novel approach for distributed application scheduling based on prediction of communication events. *Future Gener. Comput. Syst.*, 26(5):740–752, May 2010.
- [DP73] David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, October 1973.
- [F.58] Rosenblatt F. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.
- [FAL10] LUIZ ALBERICO FALCAO. *SURDEZ, COGNIÇÃO VISUAL E LIBRAS ESTABELECENDO NOVOS DIALOGOS*. Luiz Alberico, 1st edition, 2010.
- [For73] Jr. Forney, G.D. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.
- [FWCL07] Yikai Fang, Kongqiao Wang, Jian Cheng, and Hanqing Lu. A real-time hand gesture recognition method. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 995–998, 2007.
- [Gav99] D. M. Gavrilu. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73:82–98, 1999.
- [GM01] L. Gupta and Suwei Ma. Gesture-based interaction and communication: automated classification of hand gesture contours. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 31(1):114–120, feb 2001.
- [HA11] H. Hikawa and Y. Araga. Study on gesture recognition system using posture classifier and jordan recurrent neural network. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 405–412, 2011.
- [Han10] Youngmo Han. A low-cost visual motion data glove as an input device to interpret human hand gestures. *Consumer Electronics, IEEE Transactions on*, 56(2):501–509, 2010.

- [Hay99] Simon Haykin. *Neural networks : a comprehensive foundation*. Prentice Hall, Upper Saddle River (N.J.), 1999. La page de couv. porte en plus : International edition.
- [HJAJG09] A.J. Husssain, A.J. Jameel, D. Al-Jumeily, and R. Ghazali. Speech prediction using higher order neural networks. In *Innovations in Information Technology, 2009. IIT '09. International Conference on*, pages 294–298, 2009.
- [HK12] Haitham Sabah Hasan and S.Abdul Kareem. Human computer interaction for vision based hand gesture recognition: A survey. In *Advanced Computer Science Applications and Technologies (ACSAT), 2012 International Conference on*, pages 55–60, 2012.
- [HL10] B. Hartmann and N. Link. Gesture recognition with inertial sensors and optimized dtw prototypes. In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 2102–2109, 2010.
- [HMWA11] Yu Huang, Dorothy Monekosso, Hui Wang, and J.C. Augusto. A concept grounding approach for glove-based gesture recognition. In *Intelligent Environments (IE), 2011 7th International Conference on*, pages 358–361, 2011.
- [HN07] E.E. Helander and J. Nurminen. A novel method for prosody prediction in voice conversion. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–509–IV–512, 2007.
- [HR12] S.M.A. Hussain and A.B.M.H. Rashid. User independent hand gesture recognition by accelerated dtw. In *Informatics, Electronics Vision (ICIEV), 2012 International Conference on*, pages 1033–1037, 2012.
- [IK12] Noor Adnan Ibraheem and Rafiqul Zaman Khan. Article: Survey on various gesture recognition technologies and techniques. *International Journal of Computer Applications*, 50(7):38–44, July 2012. Published by Foundation of Computer Science, New York, USA.
- [Jai01] L. Jain. *Recurrent Neural Networks*. CRC Press, 1st edition, 2001.
- [JJS⁺12] Soonmook Jeong, Jungdong Jin, Taehoun Song, Keyho Kwon, and Jae Wook Jeon. Single-camera dedicated television control system using gesture drawing. *Consumer Electronics, IEEE Transactions on*, 58(4):1129–1137, 2012.
- [JLK11] Eunseok Jeong, Jaehong Lee, and DaeEun Kim. Finger-gesture recognition glove using velostat (iccas 2011). In *Control, Automation*

- and Systems (ICCAS), 2011 11th International Conference on*, pages 206–210, 2011.
- [JSFJ11] Xinghao Jiang, Tanfeng Sun, Bing Feng, and Chengming Jiang. A space-time surf descriptor and its application to action recognition with video words. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on*, volume 3, pages 1911–1915, july 2011.
- [KE11] N. Kouiroukidis and G. Evangelidis. The effects of dimensionality curse in high dimensional knn search. In *Informatics (PCI), 2011 15th Panhellenic Conference on*, pages 41–45, 2011.
- [KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
- [KK10] S. Koceski and N. Koceska. Vision-based gesture recognition for human-computer interaction and mobile robot’s freight ramp control. In *Information Technology Interfaces (ITI), 2010 32nd International Conference on*, pages 289–294, 2010.
- [KSA11] D. Kohlsdorf, T. Starner, and D. Ashbrook. Magic 2.0: A web tool for false positive prediction and prevention for gesture recognition systems. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 1–6, 2011.
- [KTK09] Ji-Hwan Kim, Nguyen Duc Thang, and Tae-Seong Kim. 3-d hand motion tracking and gesture recognition using a data glove. In *Industrial Electronics, 2009. ISIE 2009. IEEE International Symposium on*, pages 1013–1018, 2009.
- [KZ07] A. Kuzmanic and V. Zanchi. Hand shape classification using dtw and less as similarity measures for vision-based gesture recognition system. In *EUROCON, 2007. The International Conference on 34; Computer as a Tool 34;*, pages 264–269, 2007.
- [LBM01] C. Leubner, C. Brockmann, and H. Muller. Computer-vision-based human-computer interaction with a back projection wall using arm gestures. In *Euromicro Conference, 2001. Proceedings. 27th*, pages 308–314, 2001.
- [Lee06] Seong-Whan Lee. Automatic gesture recognition for intelligent human-robot interaction. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 645–650, 2006.

- [LH10] Doe-Hyung Lee and Kwang-Seok Hong. Game interface using hand gesture recognition. In *Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on*, pages 1092–1097, 2010.
- [Lin98] Tony Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, November 1998.
- [LNV⁺12] M.A.S. Lima, P.F.R. Neto, R.R. Vidal, G.H.E.L. Lima, and J.F. Santos. Libras translator via web for mobile devices. In *Telematics and Information Systems (EATIS), 2012 6th Euro American Conference on*, pages 1–4, 2012.
- [Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.
- [MAZ08] M. Maraqa and R. Abu-Zaiter. Recognition of arabic sign language (arsl) using recurrent neural networks. In *Applications of Digital Information and Web Technologies, 2008. ICADIWT 2008. First International Conference on the*, pages 478 –481, aug. 2008.
- [Mee11] S. Meena. A study on hand gesture recognition technique. Master’s thesis, National Institute Of Technology, Rourkela, India, 2011.
- [Mel87] Avraham A. Melkman. On-line construction of the convex hull of a simple polyline. *Inf. Process. Lett.*, 25(1):11–12, April 1987.
- [MT91] Kouichi Murakami and Hitomi Taguchi. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology, CHI ’91*, pages 237–242, New York, NY, USA, 1991. ACM.
- [MUK⁺06] A. Mori, S. Uchida, R. Kurazume, R.-I. Taniguchi, T. Hasegawa, and H. Sakoe. Early recognition and prediction of gestures. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 560–563, 2006.
- [NPRK05] G. N. Nandakumar, N. Patel, R. Reddy, and M. Kothandaraman. Application of douglas-peucker algorithm to generate compact but accurate ibis models. In *VLSI Design, 2005. 18th International Conference on*, pages 757–761, 2005.
- [NSPF08] M.N. Neris, A.J. Silva, S.M. Peres, and F.C. Flores. Self organizing maps and bit signature: A study applied on signal language recognition. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 2934–2941, 2008.

- [Ots79] N. Otsu. A threshold selection method from gray-level histograms. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(1):62–66, jan. 1979.
- [PKF00] B.-U. Pagel, F. Korn, and C. Faloutsos. Deflating the dimensionality curse using multiple fractal dimensions. In *Data Engineering, 2000. Proceedings. 16th International Conference on*, pages 589–598, 2000.
- [PSH97] V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):677–695, 1997.
- [QM06] S.W. Qaiyumi and D.T. Mirikitani. An extension of the k-mean algorithm for dealing with dimensionality curse. In *Engineering of Intelligent Systems, 2006 IEEE International Conference on*, pages 1–5, 2006.
- [QT09] Haiou Qin and Shixi Tang. A solution to dimensionality curse of bp network in pattern recognition based on rs theory. In *Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on*, volume 1, pages 636–638, 2009.
- [RA11] S.S. Rautaray and A. Agrawal. Interaction with virtual game through hand gesture recognition. In *Multimedia, Signal Processing and Communication Technologies (IMPACT), 2011 International Conference on*, pages 244–247, 2011.
- [Rab90] Lawrence R. Rabiner. Readings in speech recognition. chapter A tutorial on hidden Markov models and selected applications in speech recognition, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.
- [RG10] Richard Woods Rafael Gonzalez. *Processamento Digital de Imagens*. Longman do Brasil, 3rd edition, 2010.
- [RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.
- [RMM⁺09] J. Ravikiran, Kavi Mahesh, Suhas Mahishi, R. Dheeraj, S. Sudheender, and Nitin V. Pujari. Finger detection for sign language recognition. In *Proc. International Conference on Computer Science, International Multi-Conference of Engineers and Computer Scientists – IMECS 2009. ICCS-2009, IAENG, Hong Kong, 2009*.

- [SA07] T. Shanableh and K. Assaleh. Two tier feature extractions for recognition of isolated arabic sign language using fisher's linear discriminants. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II-501-II-504, 2007.
- [SCH⁺05] M. Sen, I. Corretjer, F. Haim, S. Saha, S.S. Bhattacharyya, J. Schlessman, and W. Wolf. Computer vision on fpgas: Design methodology and its application to gesture recognition. In *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, pages 133-133, 2005.
- [SCO11] J. Schlecht, B. Carque, and B. Ommer. Detecting gestures in medieval images. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1285-1288, 2011.
- [SGS11] K.V. Satya, A.K. Gogoi, and G. Sahu. Regressive linear prediction with doublet for speech signals. In *Control System, Computing and Engineering (ICCSCE), 2011 IEEE International Conference on*, pages 33-36, 2011.
- [SHL⁺98] Mu-Chun Su, Hi Huang, Chia-Hsien Lin, Chen-Lee Huang, and Chi-Da Lin. Application of neural networks in spatio-temporal hand gesture recognition. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, volume 3, pages 2116-2121 vol.3, 1998.
- [Sk182] Jack Sklansky. Finding the convex hull of a simple polygon. *Pattern Recogn. Lett.*, 1(2):79-83, December 1982.
- [SP09] E. Stergiopoulou and N. Papamarkos. Hand gesture recognition using a neural network shape fitting technique. *Eng. Appl. Artif. Intell.*, 22(8):1141-1158, December 2009.
- [SS11] D. Soendoro and I. Supriana. Traffic sign recognition with color-based method, shape-arc estimation and svm. In *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, pages 1-6, 2011.
- [ST07] D.G. Stavrakoudis and J.B. Theocharis. A recurrent fuzzy neural network for adaptive speech prediction. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 2056-2061, 2007.
- [TS07] C.-C. Teoh and G.B. Sheble. Lattice method of real option analysis - solving the curse of dimensionality and strategic planning. In *Power Tech, 2007 IEEE Lausanne*, pages 825-830, 2007.

- [VH99] E. Varoglu and K. Hacioglu. Speech prediction using recurrent neural networks. *Electronics Letters*, 35(16):1353–1355, 1999.
- [VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I-511–I-518 vol.1, 2001.
- [XGDT10] Wang Xinyu, Sun Guan, Han Dong, and Zhang Ting. Data glove gesture recognition based on an improved neural network. In *Control Conference (CCC), 2010 29th Chinese*, pages 2434–2437, 2010.
- [YG10] Dong Youfu and Tang Guoan. Comparison of popular methods for simplifying terrain based on grid dems. In *Environmental Science and Information Application Technology (ESIAT), 2010 International Conference on*, volume 3, pages 227–230, 2010.
- [YY12] Chang-Tsun Li Yi Yao. Hand posture recognition using surf with adaptive boosting. In *British Machine Vision Conference*, 2012.
- [ZCZ+08] Yu Zhou, Xilin Chen, Debin Zhao, Hongxun Yao, and Wen Gao. Mahalanobis distance based polynomial segment model for chinese sign language recognition. In *Multimedia and Expo, 2008 IEEE International Conference on*, pages 317–320, 2008.
- [ZMP10] Leigang Zhao, Yimin Mo, and Lina Pa. Research of characteristic vector based on hht in rotary machine failure diagnostics. In *Mechanic Automation and Control Engineering (MACE), 2010 International Conference on*, pages 2272–2275, 2010.
- [ZWD11] Hui Zhang, Yongqi Wang, and Chen Deng. Application of gesture recognition based on simulated annealing bp neural network. In *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, volume 1, pages 178–181, aug. 2011.
- [ZZ08] Guofeng Zhang and Dongming Zhang. Research on vision-based multi-user gesture recognition human-computer interaction. In *System Simulation and Scientific Computing, 2008. ICSC 2008. Asia Simulation Conference - 7th International Conference on*, pages 1455–1458, 2008.