ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

DEPARTAMENTO DE ENGENHARIA MECÂNICA

TUPÃ NEGREIROS

**Rastreamento labial:
Aplicação em leitura labial**

São Paulo

2013

TUPÃ NEGREIROS

Rastreamento labial: Aplicação em leitura labial

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção de título de Mestre em Engenharia Mecânica.

Área de Concentração: Engenharia de Controle e Automação Mecânica (3152)

Orientador: Prof.-Dr. Jun Okamoto Jr.

São Paulo

2013

AUTORIZADA A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 7 de Janeiro de 2013

Assinatura do autor _____

Assinatura do orientador _____

**Catálogo na Publicação
Serviço de Documentação
Escola Politécnica da Universidade de São Paulo**

Negreiros, Tupã

Rastreamento Labial: Aplicação em Leitura Labial / T. Negreiros; orientador Jun Okamoto Jr. -- ed. rev. -- São Paulo, 2013.

96 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Leitura labial. 2. Interface homem-computador 3. Rastreamento labial I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II. t.

DEDICATÓRIA

A minha mãe, com saudade.

AGRADECIMENTO

Agradeço a Deus por iluminar meu caminho e me dar força para prosseguir em mais esta etapa de minha jornada.

A Ana, pelo companheirismo e suporte ao longo do tempo em que estamos juntos.

A toda minha família, pelo carinho e apoio.

Ao Professor-Doutor Jun Okamoto Jr. e ao Professor-Doutor Marcos Ribeiro Pereira Barretto, pelas orientações para o desenvolvimento desta dissertação.

A meus amigos, tantos que não tenho espaço para citar todos, cuja paciência e diálogo foram imprescindíveis durante o período que fiquei empenhado neste trabalho.

Às empresas e colegas onde trabalhei pelo apoio em dar continuidade a esta pesquisa.

Aos funcionários do Programa de Pós-Graduação de Engenharia Mecânica – PPGEM.

“Around computers it is difficult to find the correct unit of time to measure progress. Some cathedrals took a century to complete. Can you imagine the grandeur and scope of a program that would take as long?”

*Epigrams in Programming, SIGPLAN
Association for Computing Machinery, Sep. 1982*

RESUMO

NEGREIROS, T. **Rastreamento labial: aplicação em leitura labial**. 2013, 96 p. Dissertação (Mestrado em Engenharia Mecânica) – Escola Politécnica da Universidade de São Paulo (USP). São Paulo, 2013.

Novas interfaces homem-computador têm sido pesquisadas a fim de se tornarem mais naturais e flexíveis. O rastreamento labial, foco deste trabalho, é parte deste contexto, podendo ser utilizado na detecção de emoções, bem como no auxílio ao reconhecimento de voz. Pode assim tornar-se um módulo inicial para a leitura labial, na criação de interfaces voltadas a deficientes auditivos. Os algoritmos disponíveis na literatura foram analisados e comparados, mostrando os prós e contras de cada método. Finalmente foi escolhido desenvolver uma técnica baseada em Active Appearance Model (AAM). O AAM gera um modelo a partir de um conjunto de imagens de treinamento, que pode ser utilizado no rastreamento labial de novas imagens. A técnica proposta baseia-se no uso de algoritmos genéticos para o ajuste do modelo, diferente, portanto, da técnica proposta originalmente pelo AAM. A convergência da técnica proposta foi extensivamente analisada, com a variação de parâmetros, buscando a análise de erro residual da função custo e sua relação com o tempo de convergência e erro de posição.

Palavras-chave: Rastreamento labial. Leitura labial. Interface homem-computador.

ABSTRACT

NEGREIROS, T. **Lip tracking: an application in lip reading.** 2013, 96 p. Dissertation (Master Degree in Mechanics) – Escola Politécnica da Universidade de São Paulo (USP). São Paulo, 2013.

New human-computer interfaces have been researched to make them more natural and flexible. The lip tracking, emphasis of this work, is part of this setting it can be used for detecting emotions and to aid the speech recognition. It may well become an initial module for lip-reading, in the creation of interfaces directed to hearing impaired people. Algorithms available on literature are analyzed and compared, explaining the upsides and downsides of each method. Finally was chose to develop a technique based on Active Appearance Model (AAM). The AAM generates a model from a set of training images, which can be used in tracking of new lip images. The proposed technique is based on the use of genetic algorithms for fitting the model, different therefore of the technique originally proposed by AAM. The convergence of the proposed technique has been extensively analyzed with the variation of parameters, searching the error analysis of the cost function residual error and its relation to the execution time and position error.

Keywords: Lip tracking. Lip reading. Man machine interface.

LISTA DE FIGURAS

Figura 1:	Sequência de etapas do reconhecimento labial.....	16
Figura 2:	Sequência de etapas deste projeto.....	16
Figura 3:	Princípio básico de funcionamento.....	19
Figura 4:	Detecção do contorno dos lábios	19
Figura 5:	Parametrização dos lábios.....	19
Figura 6:	Rastreamento da face em pessoas do tipo <i>lipless</i>	20
Figura 7:	Aplicação de força em uma <i>Snake</i>	23
Figura 8:	Segmentação de lábios utilizando <i>Snakes</i>	26
Figura 9:	Esquema básico do <i>Template</i>	27
Figura 10:	<i>Template</i> após várias iterações	30
Figura 11:	Modelo para ASM	31
Figura 12:	Variações treinadas para o modelo ASM	32
Figura 13:	Convergência da ASM.....	37
Figura 14:	Face original e face coberta pela textura do AAM.....	38
Figura 15:	Olho demarcado e triangulação dos pontos	39
Figura 16:	Divisão dos segmentos e nova triangulação	40
Figura 17:	Dois primeiros modos de variação da forma	42
Figura 18:	Dois primeiros modos de variação de nível de cinza	43
Figura 19:	Quatro primeiros modos de variação de aparência.....	43
Figura 20:	AAM aplicado ao rastreamento labial	45
Figura 21:	Interface de marcação de pontos.....	53
Figura 22:	Linearidade do AAM.....	58
Figura 23:	Banco de dados Obama (parte 1).....	63
Figura 24:	Banco de dados Obama (parte 2).....	64
Figura 25:	Marcação de pontos do modelo Obama.....	65
Figura 26:	Imagem média do modelo Obama.....	66
Figura 27:	Marcação de pontos do modelo de lábios.....	70
Figura 28:	Imagem média do modelo de lábios	71
Figura 29:	Solução ideal para a imagem média	72
Figura 30:	Sequência de convergência.....	73
Figura 31:	Convergência com erro de 0,215855 para a imagem média.....	73
Figura 32:	Convergência com erro de 0,386069 para a imagem média.....	74
Figura 33:	Convergência com erro de 0,343982 para imagem média	74
Figura 34:	Exemplo de resultado com $-5\sigma_c$ e $+5\sigma_c$	82
Figura 35:	Pais $-0,7\sigma_c$ e $+0,7\sigma_c$ (acima) e a imagem de teste (abaixo)	83
Figura 36:	Convergência com erro de 0,201062.....	84
Figura 37:	Convergência com erro de 0,299886.....	84
Figura 38:	Imagem média do modelo de lábios	87
Figura 39:	Imagem média do modelo de lábios utilizando AAM TOOLS.....	87
Figura 40:	Pai $-0,7\sigma_c$	88
Figura 41:	Pai $+0,7\sigma_c$	88

Figura 42: Convergência típica	88
Figura 43: Estimativa inicial utilizando AAM TOOLS	89
Figura 44: Convergência típica utilizando AAM TOOLS	89

LISTA DE GRÁFICOS

Gráfico 1: Erro x probabilidade m_1 de ocorrência de mutação	75
Gráfico 2: Erro x probabilidade m_2 de ocorrência de mutação	75
Gráfico 3: Erro x probabilidade m_3 de ocorrência de mutação	76
Gráfico 4: Erro x probabilidade m_4 de ocorrência de mutação	76
Gráfico 5: Erro x fator de variação f_1 de mutação	77
Gráfico 6: Erro x fator de variação f_2 de mutação.....	77
Gráfico 7: Erro x Fator de variação f_3 de mutação	78
Gráfico 8: Erro x Parâmetro α do crossover blend.....	78
Gráfico 9: Erro x Parâmetro β do crossover blend.....	79
Gráfico 10: Erro x Quantidade de filhos por geração	79
Gráfico 11: Tempo de execução (s) x Quantidade de filhos por geração	80
Gráfico 12: Erro médio x Número de gerações	81
Gráfico 13: Erro x Número de gerações	81
Gráfico 14: Erro médio com pais $-5\sigma_c$ e $+5\sigma_c$ a cada iteração.....	82
Gráfico 15: Erro médio com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ a cada iteração.....	83
Gráfico 16: Erros finais com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução	84
Gráfico 17: Erros de posição (<i>pixel</i>) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução.....	85
Gráfico 18: Tempo de execução (s) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução.....	86
Gráfico 19: Erro de posição (<i>pixel</i>) x Tempo de execução (s) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$...	86

LISTA DE TABELAS

Tabela 1:	Variação de forma do modelo Obama	66
Tabela 2:	Variação de nível de cinza do modelo Obama	68
Tabela 3:	Variação de aparência do modelo Obama	69
Tabela 4:	Variação de aparência do modelo de lábios	72

LISTA DE SIGLAS

AAM	Active Appearance Model
ASM	Active Shape Model
ROI	Região de Interesse (Region of Interest)

SUMÁRIO

1 INTRODUÇÃO.....	13
1.1 MOTIVAÇÃO.....	13
1.2 OBJETIVO.....	15
1.2.1 Contextualização	15
1.2.2 Objetivo Específico.....	17
1.3 ESTRUTURA DO TRABALHO	18
2 REVISÃO BIBLIOGRÁFICA	19
2.1 RASTREAMENTO LABIAL.....	22
2.1.1 Active Contour Model (Snakes).....	23
2.1.2 Templates	27
2.1.3 Active Shape Model (ASM).....	30
2.1.4 Active Appearance Model (AAM)	38
2.1.5 Thresholding.....	46
2.1.6 Outras formas de rastreamento labial	48
2.2 ANÁLISE DA BIBLIOGRAFIA	49
2.2.1 Determinação da Técnica	50
2.2.2 Treinamento	51
3 DESENVOLVIMENTO.....	52
3.1 TREINAMENTO	52
3.2 PRINCIPAL	53
3.3 RASTREAMENTO.....	58
4 RESULTADOS OBTIDOS	63
4.1 TESTE DO MODELO	63
4.2 TESTE DE CONVERGÊNCIA	69
4.2.1 Testes Iniciais.....	72
4.2.2 Análise de Sensibilidade	74
4.2.3 Análise dos Resultados.....	82
4.3 COMPARAÇÃO COM A PROPOSTA ORIGINAL	86
5 CONSIDERAÇÕES FINAIS	90
REFERÊNCIAS	92

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Os primeiros computadores dispunham apenas do teclado para permitir realizar operações, mais tarde surgiu o *mouse* como dispositivo para entrada. Inúmeras novas tecnologias vêm sendo desenvolvidas para aumentar a capacidade de interface entre homem e computador, considerada por alguns autores como peça-chave na satisfação do usuário. Substituição de teclados padrões por outras interfaces como telas *touchscreen*, reconhecimento de voz e de gestos, buscam uma interface mais natural e flexível.

Métodos padrão como teclado e *mouse* não podem ser usados por uma pessoa com alguma deficiência nos braços ou nas mãos; deficientes visuais por outro lado não têm como usar telas ou textos impressos. Entretanto, tais pessoas geralmente têm a habilidade da fala e audição, e isso pode ser instrumentalizado na utilização do computador (YAU; WEGHORN, 2007).

Já as interfaces de reconhecimento/geração de voz não têm utilidade para uma pessoa com limitações de escuta e/ou fala. Surge então a possibilidade da leitura labial, comumente usada por pessoas com deficiência auditiva que podem levar uma vida social relativamente normal com esse recurso. Apenas observando os movimentos da boca podem compreender o que uma pessoa está falando, sem nenhuma informação auditiva (YAU; WEGHORN, 2007; SUGAHARA; KISHINO; KONISHI, 2000). Mais de 80% da comunicação visual entre pessoas sem deficiências conversando é pelo movimento dos lábios (BEAUMESNIL; LUTHON, 2006).

O reconhecimento labial por computador pode vir a se tornar importante na nova geração de interfaces. Pessoas com as deficiências citadas acima podem utilizar computadores

normalmente, a fim de ter a mesma possibilidade de melhorar a educação e oportunidade de carreira das demais. Computadores que interpretam o movimento dos lábios de um interlocutor poderão ser muito úteis para ensinar deficientes auditivos a fazer leitura labial e transformar essa leitura em palavras, bem como ensinar pessoas com deficiência de fala a falar naturalmente.

Além disso, o reconhecimento labial tem a vantagem de não necessitar que o usuário emita qualquer som, isso é útil para fins de privacidade, tanto para uso pessoal, corporativo ou militar. Métodos que utilizam sinais visuais têm a vantagem de não serem intrusivos, não dependem da colocação de nenhum sensor no usuário (YAU; WEGHORN, 2007).

O reconhecimento de voz ainda hoje é pouco aproveitado. Até mesmo equipamentos portáteis como celulares podem oferecer reconhecimento de voz com bons resultados. Em geral, têm desempenho significativo em aplicações bem definidas, como ditados e médios vocabulários em ambientes relativamente controlados; mas ainda não oferecem desempenho necessário para ser uma interface completa com o usuário. O sistema tradicional de reconhecimento de voz tem bom desempenho quando existem dados de treinamento do *software* adequados e quando o ambiente é suficientemente silencioso e apropriado aos dados de treinamento coletados. Na prática, essas condições nem sempre dão precisão suficiente ao reconhecimento de voz.

Em situações adversas ainda é difícil obter alta taxa de reconhecimento utilizando apenas informações de áudio (SUGAHARA; KISHINO; KONISHI, 2000). Pesquisas têm sido feitas a fim de dar mais confiabilidade, utilizando a leitura labial em conjunto com o reconhecimento de voz em ambientes com alto nível de ruído. Tal processo é chamado de bimodal e tem apresentado bons resultados, afinal tanto a produção quanto a percepção da voz humana são bimodais (POTAMIANOS et al., 2003). Foi demonstrado que esse processo pode dar mais robustez e acurácia em condições desfavoráveis ao reconhecimento de voz

(LUETTIN; THACKER; BEET, 1996; ZHANG; MERSEREAU, 2000). A leitura labial é comumente utilizada pelas pessoas, com deficiência auditiva ou não, em grande variedade de condições visuais adversas. Isso contribui para a habilidade de reconhecer um interlocutor mesmo em condições complicadas acusticamente. De fato a maioria das pessoas tem dificuldade de comunicação em ambientes com pouca iluminação e muito ruído ou a grandes distâncias (PETAJAN; GRAF, 1996); a parte visual da comunicação é efetivamente usada para melhorar o entendimento da voz (LIEVIN; LUTHON, 1998).

O rastreamento labial, por si só, tem usos importantes como biometria, parametrização para codificadores de vídeo como MPEG4 e para detecção de emoções através dos lábios (REHMAN; LI LIU; HAIBO LI, 2007b).

1.2 OBJETIVO

1.2.1 Contextualização

Este trabalho tem por objetivo desenvolver uma técnica de rastreamento labial de um vídeo de um interlocutor, gerando parâmetros que descrevam os movimentos dos lábios e outros pontos da face. Sendo a aplicação principal deste trabalho a leitura labial, segue na Figura 1 a sequência básica de funcionamento aplicado à leitura labial. Já na Figura 2, é mostrada a sequência de funcionamento deste projeto, igual à Figura 1, porém terminando na parametrização.

Figura 1: Sequência de etapas do reconhecimento labial

Figura 2: Sequência de etapas deste projeto

O *software* deverá primeiramente procurar por faces na sequência de imagens (a). Então, decide se existe uma face e qual face é a mais importante do vídeo para iniciar o processo de procura pelos lábios desta face (b). No caso deste projeto, o escopo deste trabalho é a partir da etapa (b), considerando que já foram detectadas faces na imagem, por essa razão na Figura 2 essa etapa está acinzentada. Os pontos pertencentes ao lábio determinam uma região que pode ser descrita por meio de um conjunto de parâmetros (c). Tais parâmetros são então listados para posterior classificação (d), (e) e (f) (YANG; WAIBEL, 1996). Como o foco desta pesquisa é o rastreamento, a técnica desenvolvida é mostrada nos itens (b) e (c) da Figura 1 e Figura 2. O resultado obtido da parametrização (c) pode ser utilizado para outros fins além da leitura labial, citados no capítulo 1.1.

O sistema deve ser capaz de extrair o contorno externo dos lábios com variações na pose, luminosidade, maquiagem ou pelos faciais. É importante que o sistema seja capaz de detectar e recuperar o funcionamento caso ocorra erro de detecção (CHAN; ZHANG; HUANG, 1998).

A técnica deve ter um bom desempenho mesmo sujeito a condições adversas, como pouca luminosidade, movimentação da face, variações do interlocutor (óculos, barba e diferença de tom da pele), padrões variados de imagem de fundo ou a combinações desses.

Uma câmera montada sobre o monitor de um computador fornece uma boa visão de todas as características da face, mesmo com alguns movimentos normais do rosto (PETAJAN; GRAF, 1996).

Apesar de não ser escopo deste trabalho, a partir dos parâmetros obtidos do rastreamento dos pontos labiais do interlocutor é possível recriar os movimentos dos lábios em uma face animada sintetizada em vídeo por computador.

1.2.2 Objetivo Específico

O foco deste trabalho é o rastreamento labial definido na Figura 2 (item 1.2.1). Sendo o rastreamento facial necessário para o funcionamento da técnica desenvolvida, considera-se esta etapa como executada e a imagem de teste como a uma única face ao centro.

Detectada a face, assume-se que os lábios estão somente na metade inferior da face; não podendo haver, portanto, inversões ou rotações da imagem (GOSWAMI; CHRISTMAS; KITTLER, 2006).

Foi definido adotar basicamente a técnica de Active Appearance Model (AAM), com alterações na forma de rastreamento. Na proposta original é utilizada uma regressão linear multi-variável para buscar a convergência para a solução ideal, enquanto que neste trabalho foi utilizado Algoritmo Genético.

1.3 ESTRUTURA DO TRABALHO

No Capítulo 1 temos a introdução do texto, com a motivação deste trabalho, a caracterização do problema, os objetivos gerais e específicos.

O Capítulo 2 – “Revisão Bibliográfica” traz um breve resumo histórico das técnicas disponíveis na bibliografia para o Rastreamento Labial.

Já no Capítulo 3 – “Desenvolvimento” é explicado o desenvolvimento do software fruto deste trabalho, com as descrições matemáticas e de programação.

No Capítulo 4 temos os resultados obtidos com dois testes, primeiro um teste do modelo gerado e segundo pelo teste de convergência. Neste capítulo também são analisados os resultados, principalmente através de uma análise de sensibilidade do algoritmo, bem como com a comparação dos resultados com a proposta original do Active Appearance Model.

Por fim no Capítulo 5 temos o encerramento do texto com as considerações finais e em seguida as Referências.

2 REVISÃO BIBLIOGRÁFICA

Os princípios desenvolvidos neste trabalho podem ser extrapolados para outros rastreamentos, como o rastreamento de gestos (YANG; WAIBEL, 1996).

É possível ver na Figura 3 primeiramente a imagem normal de um rosto, a seguir a imagem depois de uma transformação e finalmente a região dos lábios detectados. Após esta etapa ainda faz-se necessário detectar o contorno dos lábios, como nas Figuras 4 e 5.

Figura 3: Princípio básico de funcionamento (YANG; WAIBEL, 1996)

Figura 4: Detecção do contorno dos lábios (CHAN; ZHANG; HUANG, 1998)

Figura 5: Parametrização dos lábios (CHAN; ZHANG; HUANG, 1998)

Essencialmente os parâmetros interessantes a serem coletados são: separação vertical, separação horizontal, perímetro e área da boca, bem como as suas derivadas no tempo (CHAN; ZHANG; HUANG, 1998). Outros parâmetros podem ser interessantes, no entanto não foram utilizados por Chan, Zhang e Huang (1998), pois havia preocupação em manter o sistema funcionando em tempo real.

Alguns estudos, entretanto, indicam que uma região aumentada, contendo a parte inferior do rosto do interlocutor (incluindo as bochechas e o queixo), aumenta a percepção da leitura labial, especialmente humana (POTAMIANOS; NETI, 2001). Rehman, Li Liu e Haibo Li (2007a) propõem rastrear pontos da face próximos aos lábios para obter resultados em pessoas que possuem lábios muito finos (*lipless*), como no exemplo da Figura 6. Saitoh, Morishita e Konishi (2008) e Petajan e Graf (1996) propõem rastrear também os dentes e as narinas devido à facilidade. A maioria das técnicas rastreia somente o contorno externo dos lábios, o que resulta em mau desempenho da leitura labial, pois o contorno interno dos lábios carrega informações importantes (RAMAGE; LINDSAY, 2009). Alguns autores (YI WU et al., 2007) sugerem como melhoria utilizar várias câmeras, não somente na posição frontal, a fim de obter uma estrutura tridimensional da face.

Figura 6: Rastreamento da face em pessoas do tipo *lipless* (REHMAN; LI LIU; HAIBO LI, 2007a)

Os lábios humanos são formas geometricamente complexas, que suavemente variam em múltiplos graus de liberdade, de acordo com os músculos faciais do interlocutor. A ideia

consiste em extrair informação desses graus de liberdade de configurações instantâneas da face (BREGLER; OMOHUNDRO, 1995).

Vários padrões deformáveis foram propostos para rastrear os lábios, mas justamente por causa da deformação o modelo geralmente não consegue representar os vários contornos em um nível de detalhamento suficiente (LUETTIN; THACKER; BEET, 1996). Vários algoritmos para estimar movimentos funcionam para objetos rígidos, mas não para a face e os lábios (YANG; WAIBEL, 1996).

O rastreamento deste projeto pode ser dividido em duas etapas: rastreamento facial e rastreamento labial (CHAN; ZHANG; HUANG, 1998).

No rastreamento facial, a entrada é um vídeo com uma ou mais faces. O *software* deve detectar essas faces a fim de iniciar a etapa seguinte. A saída desta etapa é uma região onde há grande probabilidade de ser encontrada uma face, chamada Região de Interesse ou Region of Interest (ROI), que já deve estar devidamente orientada com a face.

Esta etapa facial pode ser suprimida caso haja garantia que o interlocutor tenha posição fixa na sequência de imagens. Isso pode ser feito utilizando uma câmera acoplada a um capacete fixo ao interlocutor (GRAF; COSATTO; POTAMIANOS, 1997; LIEVIN; LUTHON, 1998; 1999). No entanto existem alguns algoritmos que podem executar essa tarefa, como o método proposto por Viola-Jones (2001) capaz de detectar objetos, extremamente rápido e com altas taxas de detecção.

O princípio básico do Viola-Jones (2001) é utilizar pequenos, simples e fracos classificadores chamados AdaBoost. Para a combinação de classificadores podem ser utilizados dois métodos, em série ou paralelo. Em paralelo, seria como combinar os classificadores por peso ou sistema de votação. Em série, como é utilizado pelo Viola-Jones (2001), o funcionamento é em cascata; caso um classificador dê resultado positivo, o teste é feito pelo classificador seguinte; caso o resultado seja negativo, o teste é interrompido

(NGUYEN; MILGRAM; NGUYEN, 2008). Isso garante grande velocidade, pois nem todos os testes precisam ser realizados para cada região, bem como garante taxa alta de detecção, pois da combinação dos classificadores fracos se tem um classificador forte. A fim de obter uma rápida classificação, o sistema descarta a maioria das características e foca somente nas críticas.

A segunda etapa, o rastreamento labial, que é o escopo deste trabalho, consiste em, conhecida uma região da imagem onde há grande possibilidade de haver lábios, a região de interesse ROI, retornar a posição desses lábios.

2.1 RASTREAMENTO LABIAL

Os métodos mais comuns encontrados na bibliografia para esta tarefa são Active Contour Model (ou Snakes), Templates, Active Shape Model (ASM) e Active Appearance Model (AAM). Outras opções de segmentação são possíveis (POTAMIANOS et al., 2003).

Em geral os métodos podem ser divididos em três categorias: baixo nível (aparência), alto nível (forma ou modelo) (POTAMIANOS; NETI, 2001; WERDA; MAHDI; HAMADOU, 2007) e uma combinação dos dois (POTAMIANOS; NETI, 2001).

A abordagem de baixo nível usa diretamente a região dos lábios. Supõe-se que os *pixels* dos lábios tenham diferença na cor comparada com os *pixels* da pele. Apenas informações de cor e gradiente são utilizadas. Teoricamente a segmentação funciona classificando os *pixels* em lábio ou pele. Geralmente são métodos rápidos e pouco precisos (GACON; COULON; BAILLY, 2005; WERDA; MAHDI; HAMADOU, 2007).

Já na abordagem de alto nível é utilizado algum tipo de modelo. Também se utiliza a informação dos *pixels* da imagem, mas busca-se regularidade. Apresenta bom resultado no

rastreamento de lábios, face, mãos, mas não consegue segmentar objetos não definidos previamente, como vasos sanguíneos ou nuvens (WERDA; MAHDI; HAMADOU, 2007).

Há autores que dividem em métodos determinísticos e estatísticos (BOUVIER; COULON; MALDAGUE, 2007). Os determinísticos não dependem de treinamento prévio, pois utilizam somente informações específicas extraídas da própria imagem para tomar uma decisão. Por outro lado, métodos estatísticos, ou holísticos, comparam a imagem atual com imagens utilizadas para treinamento antes de tomar uma decisão.

2.1.1 Active Contour Model (Snakes)

O método de *Snakes*, ou Active Contour Model, foi descrito primeiramente por Kass, Witkin e Terzopoulos (1988). Consiste de uma *spline* deformável influenciada por constantes e forças geradas pela imagem. Um processo iterativo aproxima a *spline* para a região de menor energia. A Figura 7 exemplifica a convergência da *Snake* após a aplicação de uma força afastando do ponto de menor energia.

Figura 7: Aplicação de força em uma *Snake* (KASS; WITKIN; TERZOPOULOS, 1988)

Kass, Witkin e Terzopoulos (1988) representam os parâmetros de posição da *Snake* s por $v(s) = (x(s), y(s))$ e sua energia por:

$$\begin{aligned} E_{snake} &= \int_0^1 E_{snake}(v(s)) ds = \\ &= \int_0^1 [E_{int}(v(s)) + E_{imagem}(v(s)) + E_{const}(v(s))] ds \end{aligned} \quad (2.1.1.1)$$

Onde E_{int} é a energia interna da *spline* devido à curvatura, E_{imagem} é dada pelas forças da imagem e E_{const} vem das forças constantes externas. Tais forças externas vem de aplicação de forças especificadas pelo usuário em um processo semiautomático, o que deixa de ser interessante para a aplicação deste trabalho.

A energia interna da *spline* por sua vez pode ser representada por:

$$E_{int} = \left(\alpha(s) |v_s(s)|^2 + \beta(s) |v_{ss}(s)|^2 \right) / 2 \quad (2.1.1.2)$$

Onde $\alpha(s)$ é o termo de energia da *spline* de primeira ordem e $\beta(s)$ o termo de segunda ordem. O primeiro termo faz a *spline* atuar como uma membrana e o segundo termo como uma fina placa. Com o valor de $\beta(s)$ igual a zero em um ponto, tem-se descontinuidade de segunda ordem, ou seja, um canto. As *Snakes* resultantes são flácidas demais; para obter *Snakes* mais rígidas, deve-se utilizar um método mais estável de acomodar as forças (KASS; WITKIN; TERZOPOULOS, 1988).

Já a energia dada pela imagem E_{imagem} pode ser dada pela soma ponderada das energias:

$$E_{imagem} = w_{linha} E_{linha} + w_{borda} E_{borda} + w_{fronteira} E_{fronteira} \quad (2.1.1.3)$$

Onde w_{linha} , w_{borda} e $w_{fronteira}$ são pesos.

A energia da linha pode ser representada pela própria intensidade em nível de cinza de um ponto:

$$E_{linha} = I(x, y) \quad (2.1.1.4)$$

Dependendo do sinal de w_{linha} a *Snake* será atraída por linhas claras ou linhas escuras.

No entanto, Kass, Witkin e Terzopoulos (1988) sugerem utilizar:

$$E_{linha} = -\left(G_{\sigma} * \nabla^2 I\right)^2 \quad (2.1.1.5)$$

Onde G_{σ} é uma curva gaussiana de desvio padrão σ . O mínimo desta função está nos zeros de $G_{\sigma} * \nabla^2 I$, o que define as bordas segundo a teoria de Marr-Hildreth (KASS; WITKIN; TERZOPOULOS, 1988).

Já a energia da borda pode ser dada pelo gradiente da imagem:

$$E_{borda} = |\nabla I(x, y)|^2 \quad (2.1.1.6)$$

Por fim, o termo de energia da fronteira, considera-se $C(x, y) = G_{\sigma}(x, y) * I(x, y)$ como sendo uma versão um pouco mais continua da imagem. Considera-se também $\theta = \text{atan}(C_x/C_y)$ como ângulo do gradiente, $n = (\cos \theta, \sin \theta)$ e $n_{\perp} = (-\sin \theta, \cos \theta)$ como vetores unitários, paralelo e perpendicular respectivamente, ao vetor gradiente. Então:

$$E_{fronteira} = \frac{\partial \theta}{\partial n_{\perp}} = \frac{\partial^2 C / \partial^2 n_{\perp}}{\partial C / \partial n} = \frac{C_{yy}C_x^2 - 2C_{yy}C_xC_y + C_xC_y^2}{(C_x^2 + C_y^2)^{\frac{3}{2}}} \quad (2.1.1.7)$$

O processo de minimização de energia é dependente de uma configuração inicial para ser iniciado, o qual poder feito pelo usuário ou dado por outras condições detectadas previamente na imagem. Num processo contínuo de rastreamento em um vídeo, a configuração inicial em um quadro pode ser dada pela configuração obtida do quadro anterior.

A precisão do sistema depende do critério de convergência utilizado para minimização da energia; quanto maior a precisão, mais apertado será o critério e maior o esforço computacional (LUCEY; SRIDHARAN; CHANDRAN, 2000).

Sem informações contextuais calculadas previamente nas etapas anteriores, as *Snakes* não têm bom desempenho para rastrear objetos complexos (DE LA TORRE; YACOOB; DAVIS, 2000). O gradiente pode não ser apropriado para buscar a fronteira dos lábios (LUETTIN; THACKER; BEET, 1996).

Na Figura 8 é possível ver a utilização de uma *Snake* para segmentação de lábios em uma imagem.

Figura 8: Segmentação de lábios utilizando *Snakes* (KASS; WITKIN; TERZOPOULOS, 1988)

Existem inúmeras variações dessa técnica. Shin, Bongjin Jun e Daijin Kim (2009) propõem que as forças sejam sempre perpendiculares, com isso se aumenta a velocidade de convergência e se diminui a possibilidade de a *Snake* ficar presa em mínimos locais.

2.1.2 Templates

Um *template* pode ser entendido como o modelo de um objeto. Ao ajustar alguns parâmetros, o *template* pode ser deformado até encaixar com um objeto de forma ideal, definido por uma função de custo. Minimizando essa função de custo, pode-se considerar que os parâmetros obtidos descrevem o objeto. A Figura 9 exibe um esquema adotado por Henecke, Prasad e Stork (1994) para descrever lábios.

Figura 9: Esquema básico do *Template* (HENECKE; PRASAD; STORK, 1994)

O modelo consiste basicamente de dois tipos de curva: parábolas e *quartics*, com centro nas coordenadas x_c , y_c e rotação θ . Estas coordenadas e o ângulo definem o sistema de coordenadas dos lábios (HENECKE; PRASAD; STORK, 1994).

As parábolas e os *quartics* são definidos como simétricos neste sistema de coordenadas e definidos para interceptar os eixos nos pontos $(-w, 0)$, $(0, h)$ e $(w, 0)$.

Uma parábola de altura h e largura $2w$ é descrita pela equação:

$$y = h \left(1 - \frac{x^2}{w^2} \right) \quad (2.1.2.1)$$

E a equação do *quartic* correspondente é:

$$y = h \left(1 - \frac{x^2}{w^2} \right) + 4q \left(\frac{x^4}{w^4} - \frac{x^2}{w^2} \right) \quad (2.1.2.2)$$

Onde q determina o quanto o *quartic* deriva da parábola.

Duas parábolas são do contorno interno dos lábios. Elas têm alturas diferentes h_2 e h_3 , mas largura comum $2w_i$. Para o contorno inferior externo, utiliza-se um *quartic* com altura h_4 e largura $2w_o$. Já para o contorno superior externo, são dois *quartic* com altura h_3 , largura w_o e um *offset* comum a_{off} .

Não existem restrições para definição da função custo. No entanto, usualmente se define da seguinte maneira (HENECKE; PRASAD; STORK, 1994):

- um ou mais campos de potencial são computados da imagem, normalmente representando bordas, vales e picos – e podem ser ajustados a fim de garantir um gradiente;
- a função custo seria a soma das integrais nesses campos potenciais ao longo ou entre as curvas do *template*;
- adicionalmente, parâmetros podem ser colocados via termos de penalidade a fim de melhorar o modelo.

A função custo deve ser simples computacionalmente, porém eficiente (HENECKE; PRASAD; STORK, 1994).

Já que os contornos dos lábios são basicamente horizontais, é interessante utilizar o gradiente vertical como campo potencial. Utiliza-se o filtro de Prewitt para detecção das bordas:

$$\frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad (2.1.2.3)$$

A função custo consiste de quatro integrais, uma para cada contorno dos lábios.

Considerando os contornos positivos e negativos, ela tem a seguinte forma:

$$E = -\frac{c_1}{|\Gamma_1|} \int_{\Gamma_1} \phi_e(\vec{x}) ds - \frac{c_2}{|\Gamma_2|} \int_{\Gamma_2} \phi_e(\vec{x}) ds + \frac{c_3}{|\Gamma_3|} \int_{\Gamma_3} \phi_e(\vec{x}) ds + \frac{c_4}{|\Gamma_4|} \int_{\Gamma_4} \phi_e(\vec{x}) ds \quad (2.1.2.4)$$

Onde os parâmetros c_i são coeficientes que permitem pesos diferentes às curvas Γ_i , que são as quatro curvas do *template*, os módulos $|\Gamma_i|$ são os comprimentos das curvas e $\phi_e(\vec{x})$ é o campo potencial.

Adicionado à função custo, os termos de penalização introduzem constantes espaciais e constantes temporais. As constantes espaciais garantem que os parâmetros do *template* fiquem dentro de limites aceitáveis. Esses parâmetros são definidos estatisticamente ou podem depender de outros parâmetros, por exemplo, uma constante que garante que $h_1 > h_2$. Já as constantes temporais permitem controlar as variações na espessura dos lábios. Forças do tipo mola são adicionadas para garantir a espessura dos lábios próximas da média. Tais constantes têm equação na forma:

$$E_{temp} = k_{12} \left((h_1 - h_2) - \overline{(h_1 - h_2)} \right)^2 + k_{34} \left((h_3 - h_4) - \overline{(h_3 - h_4)} \right)^2 \quad (2.1.2.5)$$

Onde k_{12} e k_{34} são as constantes das molas e $\overline{(h_1 - h_2)}$ e $\overline{(h_3 - h_4)}$ são as médias das alturas em vários quadros.

É feita média da informação de muitos *pixels* ao longo do tempo e espaço para o melhor arranjo, especialmente nos contornos do *template*, o que o torna bastante robusto. Parâmetros como taxas máximas de deformação e constantes físicas (como simetria) podem ser facilmente incorporadas à função de custo para aumentar a chance de um encaixe correto (HENECKE; PRASAD; STORK, 1994).

O *template* é sensível aos contornos relevantes, como também diferencia contornos positivos de negativos. Num contorno positivo, a intensidade da imagem é maior acima do contorno do que abaixo; num contorno negativo, a intensidade da imagem é maior abaixo do que acima do contorno. Utilizando isso como *template*, não há possibilidade de confundir o contorno superior e inferior.

É necessário um bom ponto de partida para a convergência do *template*. Um algoritmo anterior deve ser executado para determinar uma ROI com grande probabilidade de haver uma boca. As alturas e larguras iniciais são definidas de forma estatística com formato natural. Por meio do gradiente, minimizando a função custo (ou energia), o *template* converge para os lábios.

O *template*, assim como as *Snakes*, é um processo iterativo, que depende de várias iterações para atingir resultado satisfatório; como é possível ver na Figura 10.

Figura 10: *Template* após várias iterações (HENECKE; PRASAD; STORK, 1994)

É possível utilizar múltiplos *templates* para um mesmo objeto, cada qual correspondendo a uma diferente pose do objeto, o que melhora o desempenho do rastreamento. Por outro lado, dessa forma o método fica computacionalmente mais pesado (YANG; WAIBEL, 1996).

A função custo geralmente é muito sensível ao interlocutor e as condições de luz (LUETTIN; THACKER; BEET, 1996; HENECKE; PRASAD; STORK, 1994), o que o torna pouco robusto. O *template* não é invariante a mudanças e não leva em conta representações mais complexas (DE LA TORRE et al., 2000).

2.1.3 Active Shape Model (ASM)

O Active Shape Model (ASM) é um modelo estatístico de forma de objetos que se deformam a fim de fazer encaixar um objeto conhecido em uma imagem desconhecida.

Cootes et al (1995) afirmam que o modelo deve variar somente de forma característica do objeto. A ideia consiste em construir um modelo que represente tanto a forma típica como a variação típica.

O método foi proposto após *Snakes* e *templates* justamente para suprir a deficiência dos anteriores, sendo mais robusto mesmo em situações onde o objeto a ser rastreado esteja parcialmente ocluído ou a imagem tenha muito ruído. Chega a ser chamado por alguns autores como *Smart Snakes*. A expressão Active Shape Model foi dada como uma variação de Active Contour Model, outro nome de *Snakes*. Isso foi proposital para resumir a diferença entre os dois métodos (COOTES et al., 1995).

A técnica se baseia em descrever o objeto a ser rastreado como um conjunto de pontos, como mostrado na Figura 11, que mostra um exemplo de detecção de resistores em um circuito.

Figura 11: Modelo para ASM
(COOTES et al., 1995)

Os pontos são colocados da mesma maneira em vários exemplos de objetos, isso de maneira manual (LUETTIN; THACKER; BEET, 1996), com isso o modelo é treinado, como é possível ver na Figura 12. Com base nesse modelo treinado e uma nova imagem, o sistema tenta ajustar iterativamente os parâmetros do modelo para encaixá-lo na nova imagem.

Figura 12: Variações treinadas para o modelo ASM (COOTES et al., 1995)

É menos sensível ao movimento angular da face (YAU; WEGHORN, 2007). Por ser um método estatístico, é mais difícil de ser implantado, pois envolve treinamento.

Cada ponto representa uma parte particular do objeto. Os pontos podem ser definidos como três tipos diferentes:

Tipo 1: pontos marcando partes particulares do objeto dependentes da aplicação, como o centro do olho ou os cantos de uma fronteira;

Tipo 2: pontos independentes da aplicação, como o ponto mais alto de um objeto em uma orientação particular ou de maior curvatura;

Tipo 3: pontos que podem ser interpolações dos dois outros tipos.

Os pontos do tipo 1 são preferíveis aos demais, pois geralmente são mais fáceis de identificar, no entanto, os outros tipos são necessários para definir a forma do objeto com detalhe suficiente.

Primeiramente é necessário alinhar os n pontos marcados manualmente de cada imagem em um vetor x_i da seguinte forma (COOTES et al., 1995):

$$x_i = (x_{i1}, y_{i1}, x_{i2}, y_{i2}, \dots, x_{in}, y_{in}) \quad (2.1.3.1)$$

Pode-se considerar uma rotação e escala $M(s_j, \theta_j)$ e uma translação t_j entre duas formas similares x_i e x_j de forma a reduzir a soma ponderada:

$$E_j = \left(x_i - M(s_j, \theta_j) [x_j] - t_j \right)^T W \left(x_i - M(s_j, \theta_j) [x_j] - t_j \right) \quad (2.1.3.2)$$

Onde:

$$M(s, \theta) \begin{bmatrix} x_{jk} \\ y_{jk} \end{bmatrix} = \begin{bmatrix} s \cos \theta * x_{jk} - s \sin \theta * y_{jk} \\ s \sin \theta * x_{jk} + s \cos \theta * y_{jk} \end{bmatrix} \quad (2.1.3.3)$$

$$t_j = (t_{x1}, t_{y1}, \dots, t_{xj}, t_{yj}) \text{ com } 2n \text{ elementos} \quad (2.1.3.4)$$

Os pesos são escolhidos para dar mais significância aos pontos que tendem a ser mais estáveis nas imagens, os pontos que se movem menos que outros pontos do objeto.

Sendo R_{kl} a distância entre os pontos k e l em uma forma e sendo $V_{R_{kl}}$ a variância dessa distância nas formas da base de dados. Tendo isso, é possível obter o peso w_k para o ponto k como sendo:

$$w_k = \left(\sum_{l=1}^n V_{R_{kl}} \right)^{-1} \quad (2.1.3.5)$$

Desse modo, se um ponto tende a mover-se muito em relação aos outros, a soma das variâncias vai ser alta e um peso baixo será obtido. Se, ao contrário, um ponto tem tendência a manter-se fixo em relação aos outros, a soma das variâncias tende a ser menor e haverá peso maior; estes pontos serão prioritários.

Para alinhar duas formas similares x_1 e x_2 , é necessário obter uma rotação θ , uma escala s e uma translação $t = (t_x, t_y)$. Aplicando a equação (2.1.3.2) com x_1 e x_2 obtém-se:

$$E = \left(x_1 - M(s, \theta) [x_2] - t \right)^T W \left(x_1 - M(s, \theta) [x_2] - t \right) \quad (2.1.3.6)$$

Aplicando uma minimização por mínimos quadrados nas equações (2.1.3.6), (2.1.3.3) e (2.1.3.4) e adotando $a_x = s \cos \theta$ e $a_y = s \sin \theta$ para facilitar a notação, tem-se:

$$\begin{pmatrix} X_2 & -Y_2 & W & 0 \\ Y_2 & X_2 & 0 & W \\ Z & 0 & X_2 & Y_2 \\ 0 & Z & -Y_2 & X_2 \end{pmatrix} \begin{pmatrix} a_x \\ a_y \\ t_x \\ t_y \end{pmatrix} = \begin{pmatrix} X_1 \\ Y_1 \\ C_1 \\ C_2 \end{pmatrix} \quad (2.1.3.7)$$

Onde:

$$X_i = \sum_{k=1}^n w_k x_{ik} \text{ e } Y_i = \sum_{k=1}^n w_k y_{ik} \quad (2.1.3.8)$$

$$Z = \sum_{k=1}^n w_k (x_{2k}^2 + y_{2k}^2) \quad (2.1.3.9)$$

$$W = \sum_{k=1}^n w_k \quad (2.1.3.10)$$

$$C_1 = \sum_{k=1}^n w_k (x_{1k} x_{2k} + y_{1k} y_{2k}) \quad (2.1.3.11)$$

$$C_2 = \sum_{k=1}^n w_k (y_{1k} x_{2k} - x_{1k} y_{2k}) \quad (2.1.3.12)$$

Com isso se resolve o problema de rotação, escala e translação obtendo a solução do conjunto de equações lineares (2.1.3.7) (COOTES et al., 1995).

Tendo todos os pontos x_i da base de dados, chega-se à média $\bar{x}$ e ao desvio da média dx_i :

$$dx_i = x_i - \bar{x} \quad (2.1.3.13)$$

E pode-se calcular a matriz $2n \times 2n$ de covariância S :

$$S = \sum_{i=1}^N dx_i \cdot dx_i^T / N \quad (2.1.3.14)$$

Sendo p_k e λ_k os autovetores e autovalores de S :

$$S \cdot p_k = \lambda_k \cdot p_k \text{ e } p_k^T \cdot p_k = 1, \text{ sendo } \lambda_k \geq \lambda_{k+1} \quad (2.1.3.15)$$

Pode-se aproximar uma imagem x_i utilizando uma matriz com os autovetores mais importantes, ou seja, com maiores valores de λ_k . Desta forma:

$$x_i \cong \bar{x} + P b_i \quad (2.1.3.16)$$

Onde x_i é um conjunto de pontos de uma imagem, $\bar{x}$ a média dos conjuntos de pontos, P uma matriz com os primeiros autovetores e b_i um vetor de pesos na forma $b_i = (b_{i1}, b_{i2}, \dots, b_{it}, \dots, b_{it})$, sendo t os primeiros autovetores/autovalores.

Com a equação (2.1.3.16) podem-se gerar novos exemplos de formas variando b_i . Os parâmetros são linearmente independentes, porém pode existir dependência não-linear. Cootes et al. (1995) demonstram que tipicamente as imagens variam dentro da faixa:

$$-3\sqrt{\lambda_k} \leq b_{ik} \leq 3\sqrt{\lambda_k} \quad (2.1.3.17)$$

Tendo esse modelo, é necessário descrever como aproximar x_i de uma nova imagem, uma imagem desconhecida da base de treinamento. A ideia consiste em examinar a colocação de cada ponto da estimativa inicial e determinar como movê-lo para uma localização melhor.

As coordenadas de determinado conjunto de pontos X gerados pela equação (2.1.3.16) podem ser dadas por:

$$X = M(s, \theta)[x] + X_c \quad (2.1.3.18)$$

Onde $M(s, \theta)[\]$ é uma aplicação da equação (2.1.3.3) em todo o conjunto de pontos, e $X_c = (x_c, y_c, x_c, y_c, \dots, x_c, y_c)^T$ a posição central do modelo na imagem.

Iterativamente deve-se aproximar X para a imagem desconhecida. Para tanto se deseja obter o vetor de correção dX :

$$dX = (dX_1, dY_1, \dots, dX_j, dY_j)^T \text{ com } 2n \text{ elementos} \quad (2.1.3.19)$$

Supondo que a atual estimativa está centrada em (X_c, Y_c) com orientação θ e escala s , deseja-se obter uma translação (dX_c, dY_c) , rotação $d\theta$ e um fator de escala $(1+ds)$ que melhor aproximem o conjunto x ao conjunto de pontos dado por $(X + dX)$ utilizando a equação (2.1.3.7) e o método descrito acima.

Sendo a posição dos pontos de determinada iteração dada pela equação (2.1.3.18) acrescido da correção dX , tem-se:

$$M(s(1+ds), (\theta + d\theta))[x + dx] + (X_c + dX_c) = (X + dX) \quad (2.1.3.20)$$

Desenvolvendo:

$$M(s(1+ds), (\theta + d\theta))[x + dx] = (M(s, \theta)[x] + dX) - (X_c + dX_c) \quad (2.1.3.21)$$

Como $M^{-1}(s, \theta)[\] = M(s^{-1}, -\theta)[\]$, chega-se a:

$$dx = M((s(1+ds))^{-1}, -(\theta + d\theta))[M(s, \theta)[x] + dX - dX_c] - x \quad (2.1.3.22)$$

A equação (2.1.3.20) fornece um meio de obter a movimentação dos pontos para aproximar as coordenadas da forma à imagem desconhecida. Esses movimentos podem não ser consistentes com o modelo, portanto, transforma-se dx em db com a equação (2.1.3.16),

obtendo a variação dos parâmetros do modelo. Desenvolvendo a equação (2.1.3.16) e sabendo que $P^{-1} = P^T$ tem-se que:

$$db = P^T dx \quad (2.1.3.23)$$

Utilizando as equações acima é possível ajustar iterativamente os parâmetros para melhorar o modelo aproximando-o da imagem:

$$\begin{cases} X_c \rightarrow X_c + w_t dX_c \\ Y_c \rightarrow Y_c + w_t dY_c \\ \theta \rightarrow \theta + w_\theta d\theta \\ s \rightarrow s(1 + w_s ds) \\ b \rightarrow bW_b db \end{cases} \quad (2.1.3.24)$$

Onde w_t , w_θ e w_s são pesos escalares e W_b é uma matriz diagonal de pesos.

A Figura 13 mostra a convergência da ASM, com a posição inicial (a), após 100 (b), 200 (c) e 350 iterações (d).

Figura 13: Convergência da ASM
(COOTES et al., 1995)

2.1.4 Active Appearance Model (AAM)

O Active Appearance Model (AAM) tem o funcionamento muito similar ao ASM, porém, além de o modelo conter a forma do objeto, também incorpora a aparência em nível de cinza da imagem.

Possui forma e textura em escala de cinza do objeto em questão, não somente o contorno (EDWARDS; TAYLOR; COOTES, 1998). Para encaixar na imagem nova, o modelo busca parâmetros para minimizar a diferença entre a imagem original e o que seria uma face sintetizada (EDWARDS; TAYLOR; COOTES, 1998), como é possível ver na Figura 14.

Figura 14: Face original e face coberta pela textura do AAM (EDWARDS; TAYLOR; COOTES, 1998)

O AAM representa a imagem gerada como uma função dos parâmetros internos e subespaços, que fornecem os parâmetros do lábio buscado.

Na fase de treinamento, o AAM aprende a correlação entre os parâmetros escolhidos e o resíduo da aproximação. Na fase de encaixe, faz o contrário, mede os resíduos e corrige os parâmetros atuais, direcionando a melhor aproximação (EDWARDS; TAYLOR; COOTES, 1998).

O modelo deve ser treinado com uma quantidade suficiente de imagens. Cada imagem precisa ser marcada com pontos representando as características buscadas. O modelo é

representado colocando o conjunto de pontos de cada imagem em um vetor x_i , contendo n_0 pontos de treinamento, da seguinte forma (COOTES; EDWARDS; TAYLOR, 1998):

$$x_i = (x_{i1}, y_{i1}, x_{i2}, y_{i2}, \dots, x_{in_0}, y_{in_0}) \quad (2.1.4.1)$$

Pode-se considerar uma rotação e escala $M(s, \theta_j)$ e uma translação t_j entre duas formas similares x_i e x_j de forma a reduzir a soma ponderada:

$$E_j = (x_i - M(s_j, \theta_j)[x_j] - t_j)^T W (x_i - M(s_j, \theta_j)[x_j] - t_j) \quad (2.1.4.2)$$

Onde:

$$M(s, \theta) \begin{bmatrix} x_{jk} \\ y_{jk} \end{bmatrix} = \begin{bmatrix} s \cos \theta * x_{jk} - s \sin \theta * y_{jk} \\ s \sin \theta * x_{jk} + s \cos \theta * y_{jk} \end{bmatrix} \quad (2.1.4.3)$$

$$t_j = (t_{xj}, t_{yj}, \dots, t_{xj}, t_{yj}) \text{ com } 2n_0 \text{ elementos} \quad (2.1.4.4)$$

E W é uma matriz diagonal de pesos para cada coordenada. Como a quantidade de pontos marcados manualmente é pequena para representar uma imagem, executa-se uma triangulação dos pontos descrita por Cootes e Taylor (1994) e incluem-se pontos no ponto médio de cada arco, gerando mais pontos até quanto se queira. Como exemplo, uma figura como um olho, com n_0 pontos demarcados: a partir desses pontos é feita a triangulação, determinando segmentos.

Figura 15: Olho demarcado e triangulação dos pontos (COOTES; TAYLOR, 1994)

Então os segmentos são divididos ao meio, obtendo-se novos pontos. Depois é possível triangular novamente, conforme a Figura 16, abaixo. Repetidamente é possível obter quantos pontos se desejar.

Figura 16: Divisão dos segmentos e nova triangulação
(COOTES; TAYLOR, 1994)

Agora há n pontos. Não é necessário alinhá-los novamente, minimizando a equação (2.1.4.2), pois a triangulação foi executada com os pontos já alinhados.

Preenchendo o vetor x_i com os novos pontos na forma da equação (2.1.4.1), só que agora com n pontos, e aplicando uma análise de componentes principais aos dados, tem-se que qualquer conjunto pode ser aproximado por:

$$x_i \cong \bar{x} + P_s b_{si} \quad (2.1.4.5)$$

Onde x_i é um conjunto de pontos de uma imagem, $\bar{x}$ é a média dos conjuntos de pontos, P_s é uma matriz com as variações na forma ortogonal e b_{si} é um conjunto de parâmetros.

Destes n pontos, cada qual tem um valor de intensidade em nível de cinza I . Como não necessariamente os pontos intermediários têm coordenadas inteiras, deve-se fazer uma interpolação bilinear para obter a intensidade em tal ponto. Alinhando as coordenadas de cada ponto é obtido para cada imagem:

$$g_i = (I_{i1}, I_{i2}, \dots, I_{in}) \quad (2.1.4.6)$$

A fim de minimizar o efeito da luminosidade, é possível normalizar os dados aplicando uma escala α e um *offset* β à informação de escala de cinza de g_i .

$$g_i = (g'_i - \beta_i \cdot 1) / \alpha_i \quad (2.1.4.7)$$

Onde g'_i é o valor original do vetor g_i . E α_i e β_i são escolhidos da seguinte forma:

$$\alpha_i = g_i \cdot \bar{g} \text{ e } \beta_i = (g_i \cdot 1) / n \quad (2.1.4.8)$$

Note-se que é um processo iterativo. Aplicando análise de componentes principais à equação (2.1.4.6), de maneira análoga a equação (2.1.4.5) é obtida a aproximação:

$$g_i \cong \bar{g} + P_g b_{gi} \quad (2.1.4.9)$$

Onde g_i é um vetor normalizado de nível de cinza de uma imagem, $\bar{g}$ é a média desses vetores de todas as imagens de treinamento, P_g é uma matriz com as variações na forma ortogonal e b_{gi} é um conjunto de parâmetros de nível de cinza.

Desse modo, a forma e a aparência de qualquer imagem podem ser descritas pelos vetores b_{si} e b_{gi} . É possível então definir um vetor b'_i concatenando ambos os vetores, de forma que:

$$b'_i = \begin{bmatrix} b_{si} \\ b_{gi} \end{bmatrix} = \begin{bmatrix} P_s^T (x_i - \bar{x}) \\ P_g^T (g_i - \bar{g}) \end{bmatrix} \quad (2.1.4.10)$$

Pode-se ainda adotar uma matriz diagonal W_s com pesos para cada parâmetro de forma, de modo a permitir diferenças entre as unidades da forma com o nível de cinza:

$$b_i = \begin{bmatrix} W_s b_{si} \\ b_{gi} \end{bmatrix} = \begin{bmatrix} W_s P_s^T (x_i - \bar{x}) \\ P_g^T (g_i - \bar{g}) \end{bmatrix} \quad (2.1.4.11)$$

Aplicando novamente o modelo da análise de componentes principais:

$$b_i = Qc_i \quad (2.1.4.12)$$

Onde Q são os autovalores e c_i são os parâmetros de aparência, que incorporam tanto a forma como os níveis de cinza. Os parâmetros de forma e os de nível de cinza têm média zero, e b_i também tem.

Pela linearidade do modelo, pode-se expressar a forma e o nível de cinza como funções diretas de c_i :

$$x_i = \bar{x} + P_s W_s Q_s c_i \text{ e } g_i = \bar{g} + P_g Q_g c_i, \text{ onde } Q = \begin{bmatrix} Q_s \\ Q_g \end{bmatrix} \quad (2.1.4.13)$$

Dessa forma, uma imagem pode ser sintetizada para um dado c_i por meio da variação da forma x_i e do nível de cinza sem-forma g_i . A seguir é mostrado como a variação dos parâmetros dos autovalores e autovetores mais importantes modificam a imagem sintetizada, todos com variações de ± 3 vezes o desvio padrão.

Na Figura 17, são mostrados os dois primeiros modos de variação da forma.

Figura 17: Dois primeiros modos de variação da forma (COOTES; EDWARDS; TAYLOR, 1998)

Na Figura 18 são mostrados os dois primeiros modos de variação de nível de cinza.

Figura 18: Dois primeiros modos de variação de nível de cinza (COOTES; EDWARDS; TAYLOR, 1998)

E na Figura 19 é possível ver os quatro primeiros modos de variação da aparência (conjunto de forma e nível de cinza).

Figura 19: Quatro primeiros modos de variação de aparência (COOTES; EDWARDS; TAYLOR, 1998)

Descrito o modelo, é necessário agora é minimizar a diferença entre uma imagem sintetizada e uma imagem desconhecida da base de treinamento. Um vetor de diferença δI pode ser definido como:

$$\delta I = I_i - I_m \quad (2.1.4.14)$$

Onde I_i é o vetor de nível de cinza na nova imagem e I_m é o vetor de nível de cinza da imagem sintetizada pelo modelo. É preciso minimizar a diferença entre a imagem sintetizada e a nova imagem, minimizando o vetor diferença $\Delta = |\delta I|^2$ variando os parâmetros c_i . Cootes, Edwards e Taylor (1998) descrevem que, apesar de parecer ser um problema de otimização

complexo, notou-se que cada tentativa de minimizar essa diferença entre cada nova imagem é um problema similar.

É possível aprender como resolver esse problema em avanço por meio do conhecimento de informações *a priori* de como ajustar os parâmetros do modelo durante o encaixe da face sintetizada.

O que é feito aprendendo a relação entre δI e os erros nos parâmetros do modelo δc utilizando esse conhecimento em um processo iterativo para minimizar Δ .

O modelo mais simples de obter a relação entre a diferença δI e o erro entre os parâmetros do modelo (e as correções a serem feitas) é linear, da forma:

$$\delta c = A\delta I \quad (2.1.4.15)$$

Segundo Cootes et al. (1998), essa aproximação é boa o bastante para obter bons resultados dentro de uma pequena faixa de valores. Para obter A , é feita uma múltipla regressão linear multivariável utilizando δc a partir de imagens treinadas, e o correspondente δI . Pode-se gerar essa variação aleatória a partir da perturbação do modelo conhecido de imagens treinadas.

Além de perturbações nos parâmetros do modelo, é possível introduzir variações na translação, escala e orientação na regressão.

São incluídos $s \cos \theta$, $s \sin \theta$, t_x e t_y como elementos extras de δc para facilitar a notação.

A partir dos parâmetros conhecidos da imagem c_0 , são feitos movimentos com um valor conhecido δc para obter $c = \delta c + c_0$. Com tais parâmetros é sintetizada a nova a forma x_i e o nível de cinza g_i com a equação (2.1.3.13).

Novamente é amostrada a imagem com os pontos x_i e obtido g_{si} . O erro dessa amostra i é, portanto, $\delta g_i = g_{si} - g_i$.

Analisando cada imagem da base de dados, obtém-se para cada uma δc_i e δg_i .

Tendo isso, é feita a regressão linear multivariável para obter $\delta c = A\delta g$.

Assim é possível definir um método iterativo:

- Obter o vetor erro $\delta g_{ij} = g_{si} - g_i$ para a j-ésima interação
- Obter $E_j = |\delta g_{ij}|^2$
- Calcular a nova variação $\delta c = A\delta g$
- Calcular novo $c_{j+1} = c_j - k\delta c$ com $k=1$
- Amostrar a imagem com a nova estimativa e calcular o novo vetor erro δg_{ij+1}
- Se $E_{j+1} < E_j$ então aceitar a nova estimativa c_{j+1} , senão repetir a estimativa com $k=1,5$; $k=0,5$; $k=0,25$...
- Repetir este procedimento até que não tenha diferença no erro.

Na Figura 20, é exibida a sugestão de Saitoh, Morishita e Konishi (2008) de rastrear o contorno interno e externo dos lábios e a narina:

Figura 20: AAM aplicado ao rastreamento labial (SAITOH; MORISHITA; KONISHI, 2008)

Alguns autores deram continuidade à pesquisa do AAM e propuseram melhorias. Luo, Huang e Hsiao (2011) sugerem aplicar o AAM somente em alguns quadros da sequência, nos demais utilizam outro algoritmo chamado Lucas & Kanade somente para acompanhar o movimento dos pontos. Essa alteração visa diminuir o custo computacional da aplicação do AAM, ou seja, o tempo gasto na convergência em cada quadro de uma sequência de imagens.

Já Umit Batur e Monson (2003) sugerem alterar a forma de cálculo do gradiente referente à matriz A da equação (2.1.4.15), devido ao fato de a linearidade ser observada somente em intervalos muito pequenos. Conforme Cootes, Edwards e Taylor (1998), a matriz A deve ser calculada a partir das imagens de treinamento, obtendo uma matriz constante, enquanto Umit Batur e Monson (2003) a recalculam em cada iteração da imagem de teste, e com isso melhoraram a velocidade do algoritmo.

2.1.5 Thresholding

Um dos métodos mais simples para segmentação de imagens que se pode pensar é o chamado *Thresholding*. Dada uma escala, geralmente em níveis de cinza, acima de determinado limite é considerado “1”, objeto, abaixo desse limite “0”. Variações de tal método são propostas por alguns autores para segmentação de lábios.

Por ser um modelo de baixo nível, apresenta grande velocidade, diferente dos modelos de alto nível, que precisam de bom treinamento para reduzir o tempo de processamento (OOI et al., 2008).

A maior dificuldade no método *thresholding* é determinar um limite ideal em um quadro e aplicá-lo em uma sequência de imagens (LUCEY; SRIDHARAN; CHANDRAN, 1999).

Alguns autores, como, por exemplo, Lucey, Sridharan e Chandran (1999), propõem transformações no espaço de cores da imagem, utilizando a proporção vermelho sobre verde, R/G como escala para essa análise. Zhang e Mersereau (2000) afirmam que a escala em Hue apresenta maior discriminação da região dos lábios do restante do rosto do que o sistema RGB e YCbCr, bem como menor variação com condições de luz e interlocutor. Há autores (EVENO; CAPLIER; COULON, 2001), por outro lado, que afirmam que, para determinadas

pele e os lábios têm valor de Hue muito próximos. Guan (2006) sugere transformações complexas de cores para facilitar a separação e minimizar a influência da luminosidade. Já Dargham e Chekima (2006) propõem transformação simples dos valores RGB (vermelho, verde e azul respectivamente) para outra escala pela equação:

$$O(x, y) = R(x, y) + B(x, y) - 6G(x, y) \quad (2.1.5.1)$$

Rui Ma et al. (2007) sugerem outra transformação simples dos valores RGB a fim de compensar a luminosidade pelas equações:

$$R'(x, y) = \frac{R(x, y)}{R(x, y) + B(x, y) + G(x, y)} \quad (2.1.5.2)$$

$$G'(x, y) = \frac{G(x, y)}{R(x, y) + B(x, y) + G(x, y)} \quad (2.1.5.3)$$

$$B'(x, y) = \frac{B(x, y)}{R(x, y) + B(x, y) + G(x, y)} \quad (2.1.5.4)$$

É possível utilizar medição Fuzzy para determinar um limite de maneira mais confiável. Isso com base na informação temporal de uma sequência de imagens para obter um limite de *threshold* estável e separar os lábios do restante da pele do rosto do interlocutor (LUCEY; SRIDHARAN; CHANDRAN, 1999).

O maior problema de determinar um valor-limite para segmentação de lábios é que as distribuições de cores nos *pixels* dos lábios e nos *pixels* da face se sobrepõem em situações normais de luminosidade; fora o efeito *blur* que ocorre devido ao movimento (PAN; KONG; LIEW, 2008). Um algoritmo baseado em *thresholding* deve encontrar um limite que minimize a probabilidade de segmentar errado as imagens (LUCEY; SRIDHARAN; CHANDRAN, 1999). Especialmente em algumas pessoas que apresentam pouca diferença de cor entre o contorno externo dos lábios e a pele da face (GUAN, 2006).

Tal técnica tem a vantagem de ser mais rápida que as anteriores, baseado apenas nos *pixels*, pode-se obter resultados com apenas um passe pela imagem (YANG; WAIBEL, 1996; POTAMIANOS; NETI, 2001). A cor em determinadas condições de luz é invariante à orientação (YANG; WAIBEL, 1996).

No entanto, basear a representação de cores para fazer a segmentação apresenta problemas: a representação é muito influenciada pela luz ambiente e pela imagem de fundo, e diferentes câmeras produzem diferentes valores de cor, ainda que na mesma pessoa e ambiente (LUCEY; SRIDHARAN; CHANDRAN, 2000). Existe a dificuldade de determinar o contorno interno dos lábios devido à semelhança da cor dos lábios com a língua; o algoritmo proposto por Pan, Kong e Liew (2008) consegue contornar esse problema.

Por fim, por ter menor precisão, mas ser rápida, esta técnica pode ser utilizada como aproximação inicial de outros métodos citados anteriormente; bem como pode servir de verificação de outros métodos.

2.1.6 Outras formas de rastreamento labial

Alguns autores (MENG LI; CHEUNG, 2008) definem duas categorias: métodos estacionários e métodos baseados em movimento. Todos os já analisados são estacionários, apesar de poder usar informações de quadros anteriores.

Já nos métodos baseados em movimento, resumidamente, a detecção funciona comparando o quadro atual com os quadros anteriores, com isso se obtêm as regiões com movimentação. O movimento é um parâmetro que pode ser facilmente extraído, obtendo rapidamente, por exemplo, a região da cabeça. Basicamente é necessário obter o valor

absoluto da diferença entre os quadros, aplicar um filtro passa-baixa e *thresholding* (GRAF; COSATTO; POTAMIANOS; 1997).

Quando uma pessoa fala, o movimento da face relativo à câmera é pequeno, enquanto o movimento dos lábios é grande. A frequência do movimento dos lábios é mais alta que outros movimentos da face. A detecção do movimento por meio de processamento morfológico pode ser usada para detectar a área da boca e os parâmetros dos lábios. Os resultados não são exatos a cada quadro da sequência, porém refletem a informação dinâmica do movimento (GANG LI; WANG; LING LIN, 2006).

2.2 ANÁLISE DA BIBLIOGRAFIA

A informação instantânea da configuração da face deve ser feita a cada quadro da sequência de vídeo. A informação dos quadros anteriores pode ser utilizada para aperfeiçoar o desempenho do rastreamento.

A informação temporal não deve ser deixada de lado, pois se trata de um fluxo de imagens e não de um único quadro. Potamianos e Neti (2001) sugerem que utilizar janelas temporais maiores melhoram o desempenho do rastreamento. Uma janela temporal pode melhorar o rastreamento atuando em quadros em que ocorre oclusão dos lábios (PETAJAN; GRAF, 1996). Alguns autores utilizam maquiagens especiais para facilitar o processo, como batom e marcas pontuais na face (BREGLER; OMOHUNDRO, 1995; PETAJAN; GRAF, 1996; GRAF; COSATTO; POTAMIANOS; 1997).

Outra consideração importante é a tolerância ao movimento, o quanto a face se move na sequência de imagens. Quando o movimento é lento, a localização da face varia pouco, então a busca é necessária somente em uma região pequena, reduzindo o custo computacional. Se o rastreador tiver de buscar em uma janela muito grande, precisará trabalhar com grande

quantidade desnecessária de dados, porém, ao buscar em uma região pequena, pode facilmente perder a posição da face (YANG; WAIBEL, 1996). Durante o processo, o rastreamento em uma pequena região pode se perder, e nesse caso é necessário reiniciar o método. A dificuldade consiste em determinar quando a reinicialização é necessária (STILLITTANO; GIRONDEL; CAPLIER, 2009).

2.2.1 Determinação da Técnica

Apesar do grande avanço na tecnologia de rastreamento labial, não existe um caminho claro mostrando quais algoritmos trazem vantagens sobre outros (REHMAN; LI LIU; HAIBO LI, 2007b).

Dadas às opções, nota-se que métodos de baixo nível (*thresholding*) são bem menos robustos que os de alto nível, especialmente no tocante à variação de condições de iluminação. Dentre os métodos de alto nível, os estatísticos (ASM e AAM) apresentam melhores resultados que os determinísticos (*Snakes* e *templates*).

No caminho inverso, os estatísticos são os métodos com maior carga computacional, no entanto, com o avanço da tecnologia, isso deixa de se tornar limitante.

Zhang e Mersereau (2000) afirmam que os modelos estatísticos são relativamente mais sofisticados o que (na época) proibiam análise em tempo real, ou seja, na mesma velocidade com que a sequência de quadros do vídeo é gerada. Perez et al. (2005) descrevem uma implementação de ASM em dispositivos embarcados.

Talvez a maior dificuldade desses métodos seja a grande quantidade de dados de treinamento para criação do modelo (ZHIYONG WU; JIYING WU; MENG, 2008).

O ASM e o AAM têm vantagem sobre os demais, pois trabalha diretamente com o nível de cinza da imagem, sendo menos influenciados por problemas com cores, como fundo ou iluminação (LUCEY; SRIDHARAN; CHANDRAN, 2000).

O AAM incorpora a textura de nível de cinza ao método ASM. Isso o torna mais robusto, especialmente em relação ao problema de mínimo local (NGUYEN; MILGRAM; NGUYEN, 2008).

Assim, optou-se por desenvolver uma técnica baseada no AAM, com algumas possíveis melhorias.

2.2.2 Treinamento

Para implantação do AAM, faz-se necessário treinamento do modelo do algoritmo por meio da aprendizagem de imagens marcadas manualmente. A constelação de pontos marcados é diferente para cada imagem. Para uma mesma pessoa, diferentes imagens terão diferentes configurações. Também ocorrerão alterações devido ao ângulo de câmera e pose (DIBEKLIOGLU; SALAH; GEVERS, 2011).

Rehman, Li Liu e Haibo Li (2007b) demonstram que a marcação humana introduz ruído, o que pode ser significativo a ponto de prejudicar o desempenho. O problema é devido à ambiguidade da marcação; para uma mesma imagem, duas pessoas podem marcar a imagem de modo diferente, até uma mesma pessoa pode fazer duas marcações diferentes na mesma imagem. Ignorar esse fato gera confiança exagerada no algoritmo, o que pode prejudicar o desempenho.

3 DESENVOLVIMENTO

O algoritmo desenvolvido foi dividido em três partes: Treinamento, Principal e Rastreamento.

O Treinamento consiste basicamente de uma interface para marcação manual dos pontos em uma grande quantidade de imagens. Na seção Principal, ocorre a classificação dos dados marcados manualmente gerando um modelo. Finalmente na seção do Rastreamento é aplicado o modelo obtido na etapa anterior a uma imagem ou a uma sequência de imagens.

O software foi desenvolvido em ambiente Visual Studio 2010, a parte de treinamento em linguagem Visual Basic, enquanto as partes Principal e Rastreamento em linguagem C++. Foi utilizada a biblioteca Eigen para representação e cálculos de matriz e vetores com tamanho variável, bem como foi utilizada a biblioteca OpenCV para leitura e escrita de imagens.

3.1 TREINAMENTO

Nesta etapa existe uma interface para marcação manual dos pontos em cada uma das imagens do banco de dados como pode ser observado na Figura 21:

Figura 21: Interface de marcação de pontos

É possível criar ou mover os pontos marcados. É importante que todas as imagens tenham a mesma quantidade de pontos e os pontos sejam demarcados na mesma sequência em cada uma das imagens do banco de dados.

Os dados são salvos em um arquivo XML para utilização na etapa posterior.

3.2 PRINCIPAL

Tendo um conjunto de m imagens demarcadas e as coordenadas dos respectivos pontos em um vetor x_i , contendo n_0 pontos de treinamento, de forma que:

$$x_i = (x_{i1}, y_{i1}, x_{i2}, y_{i2}, \dots, x_{in_0}, y_{in_0}) \quad (3.2.1)$$

É obtido $\bar{x}$ como vetor médio do vetor x_i de todas as imagens. Considera-se uma rotação e escala $M(s_i, \theta_i)$ e uma translação t_i entre cada x_i em relação ao $\bar{x}$, calculado

resolvendo o sistema de equações descrito na equação (3.2.2). De modo a facilitar a notação, é considerado $s_x = s \cos \theta$ e $s_y = s \sin \theta$; sendo w_k o peso de cada ponto, e x_{ik} e y_{ik} os valores das coordenadas do ponto k da imagem i .

$$\begin{pmatrix} X_i & -Y_i & W & 0 \\ Y_i & X_i & 0 & W \\ Z & 0 & X_i & Y_i \\ 0 & Z & -Y_i & X_i \end{pmatrix} \begin{pmatrix} s_x \\ s_y \\ t_x \\ t_y \end{pmatrix} = \begin{pmatrix} \bar{X} \\ \bar{Y} \\ C_1 \\ C_2 \end{pmatrix} \quad (3.2.2)$$

Onde:

$$X_i = \sum_{k=1}^{n_0} w_k x_{ik} \text{ e } Y_i = \sum_{k=1}^{n_0} w_k y_{ik} \quad (3.2.3)$$

$$\bar{X} = \sum_{k=1}^{n_0} w_k \bar{x}_k \text{ e } \bar{Y} = \sum_{k=1}^{n_0} w_k \bar{y}_k \quad (3.2.4)$$

$$Z = \sum_{k=1}^{n_0} w_k (x_{ik}^2 + y_{ik}^2) \quad (3.2.5)$$

$$W = \sum_{k=1}^{n_0} w_k \quad (3.2.6)$$

$$C_1 = \sum_{k=1}^{n_0} w_k (\bar{x}_k x_{ik} + \bar{y}_k y_{ik}) \quad (3.2.7)$$

$$C_2 = \sum_{k=1}^{n_0} w_k (\bar{y}_k x_{ik} - \bar{x}_k y_{ik}) \quad (3.2.8)$$

Os pesos dão mais importância aos pontos que se movem menos em relação aos outros do objeto. Sendo R_{kl} a distância entre os pontos k e l em uma forma e $V_{R_{kl}}$ a variância dessa distância nas formas da base de dados. Pode-se obter o peso w_k para o ponto k como sendo:

$$w_k = \left(\sum_{l=1}^{n_0} V_{R_{kl}} \right)^{-1} \quad (3.2.9)$$

Adota-se inicialmente o peso para cada ponto como unitário e $s_x = 1$, $s_y = 0$, $t_x = 0$ e $t_y = 0$. Iterativamente é calculada a rotação, escala e translações, o peso e finalmente o novo $\bar{x}$ de todos os pontos, até que a variação da norma de $\bar{x}$ seja menor que 1%. A cada iteração é necessário combinar as novas e anteriores rotações, escalas e translações, pois essas informações são necessárias no momento de extrair os valores de nível de cinza da imagem original.

Tendo os pontos devidamente alinhados, é feita sua triangulação a fim de obter maior quantidade de pontos, sendo agora n pontos, conforme descrito no tópico 2.1.4. Quanto maior a quantidade de pontos, melhor a resolução da imagem de modelo, porém o custo computacional aumenta em ordem quadrática. Devido a isso, trabalhou-se com a quantidade na ordem de 2.000 pontos.

É importante executar o algoritmo de triangulação no vetor $\bar{x}$ e não individualmente em cada imagem, pois isso pode gerar pontos que não são equivalentes entre as imagens. Cada ponto gerado na imagem média (do vetor $\bar{x}$) deve ser registrado e depois criado na mesma ordem em cada uma das imagens do banco de dados.

O vetor x_i , agora com n pontos e sendo aplicada análise de componentes principais aos dados, pode ser aproximado por:

$$x_i \cong \bar{x} + P_s b_{si} \quad (3.2.10)$$

Onde x_i é um conjunto de pontos de uma imagem, $\bar{x}$ é a média dos conjuntos de pontos, P_s é uma matriz com as variações na forma ortogonal e b_{si} um conjunto de parâmetros. Os conjuntos autovetores/autovalores são organizados do mais importante para o menos importante, sendo então adotados como representativos conforme os autovalores que correspondam a, pelo menos, 90% da soma dos autovalores.

É extraído da imagem o valor de intensidade em nível de cinza I de cada um dos n pontos. É feita uma interpolação bilinear em cada ponto, pois devido aos cálculos anteriores os pontos não necessariamente têm coordenadas inteiras. É necessário desfazer a rotação, translação e escala aplicadas às coordenadas dos pontos no momento de extrair as intensidades. Os valores de nível de cinza são então alinhados de forma que:

$$g_i = (I_{i1}, I_{i2}, \dots, I_{in}) \quad (3.2.11)$$

A fim de minimizar o efeito da luminosidade, são normalizados iterativamente os dados aplicando uma escala α e um *offset* β à informação de escala de cinza de g_i em cada imagem da base de dados:

$$g_i = (g'_i - \beta_i \cdot 1) / \alpha_i \quad (3.2.12)$$

Onde g'_i é o valor original do vetor g_i . São escolhidos α_i e β_i da seguinte forma:

$$\alpha_i = g_i \cdot \bar{g} \text{ e } \beta_i = (g_i \cdot 1) / n \quad (3.2.13)$$

Essa normalização é executada até que a variação da norma do vetor $\bar{g}$ seja menor que 0,1%. Aplicando uma análise de componentes principais aos vetores g_i de maneira análoga à equação (3.2.10), é obtida a aproximação:

$$g_i \cong \bar{g} + P_g b_{gi} \quad (3.2.14)$$

Onde g_i é um vetor normalizado de nível de cinza de uma imagem, $\bar{g}$ é a média desses vetores de todas as imagens de treinamento, P_g é uma matriz com as variações na forma ortogonal e b_{gi} um conjunto de parâmetros de nível de cinza. De modo equivalente os conjuntos autovetores/autovalores são organizados do mais importante para o menos

importante e adotados como representativos os que correspondam a, pelo menos, 90% da soma dos autovalores.

Desse modo, a forma e o nível de cinza de qualquer imagem pode ser descrita pelos vetores b_{si} e b_{gi} . É definido um vetor b_i concatenando os dois, de modo que:

$$b_i = \begin{bmatrix} b_{si} \\ b_{gi} \end{bmatrix} = \begin{bmatrix} P_s^T (x_i - \bar{x}) \\ P_g^T (g_i - \bar{g}) \end{bmatrix} \quad (3.2.15)$$

Diferente do proposto por Cootes, Edwards e Taylor (1998), não foi considerada uma matriz W_s com pesos, de fato foi adotado a matriz identidade.

É aplicado novamente o modelo da análise de componentes principais:

$$b_i \cong Qc_i \quad (3.2.16)$$

Onde Q são os autovalores e c_i os parâmetros de aparência, que incorporam tanto a forma como os níveis de cinza. Os parâmetros de forma e os de nível de cinza têm média zero, b_i também tem. Igualmente foram organizados os autovetores/autovalores do mais importante para o menos importante e adotados como representativos os que correspondam a, pelo menos, 90% da soma dos autovalores.

São incluídos na notação do vetor c_i mais quatro parâmetros na última linha: s_x , s_y , t_x e t_y , nessa ordem, obtidos com a rotação, translação e escala da equação (3.2.2). Afinal os quatro parâmetros também influenciam no resultado final.

Nas imagens marcadas manualmente do banco de dados, obtêm-se os valores exatos de x_i e g_i , com que é possível obter a aproximação c_i apenas invertendo a equação (3.2.16) e a equação (3.2.15). A partir dos vetores c_i de cada imagem, calcula-se o desvio padrão σ_c para ser utilizado na etapa seguinte.

Finalmente são gravados os parâmetros: quantidade de pontos marcados manualmente n_0 e total n , os vetores $\bar{x}$ e $\bar{g}$, e as matrizes $P_s Q_s$ e $P_g Q_g$, bem como o vetor σ_c em um arquivo XML para utilização na etapa de Rastreamento.

3.3 RASTREAMENTO

A etapa de Rastreamento é a mais crítica em questão de tempo de execução do projeto, pois é a única que é executada *online* com a entrada da sequência de imagens ou de uma imagem estática. A abordagem utilizando a matriz proposta por Cootes, Edwards e Taylor (1998) e descrita na equação (2.1.4.15) apresenta uma linearidade que só pode ser observada em uma faixa muito pequena, como se pode notar na Figura 22, onde é mostrado o valor real e o valor previsto dos *pixels*.

Figura 22: Linearidade do AAM
(COOTES; EDWARDS; TAYLOR, 1998)

Alguns autores, como Umit Batur e Monson (2003), sugerem alternativas a esse método, recalculando a matriz a matriz A a cada iteração. Cootes e Taylor (1994) já sugerem utilizar um algoritmo genético para buscar a convergência do AAM, porém sem entrar muito em detalhes.

Dessa forma foi decidido adotar um algoritmo genético para convergência do modelo do AAM às imagens de teste.

Nesta etapa, tendo o modelo parametrizado e uma imagem ou sequência de teste, deseja-se obter o vetor aproximado de x que parametriza a imagem de teste. Para tanto, deseja-se aproximar o vetor c da imagem de teste. É adotado um valor inicial c_k , onde $k = 0$, e são aproximados os vetores x_k e g_k a partir da aplicação inversa da equação (3.2.16) e da equação (3.2.15). Assim, extrai-se da imagem o vetor g'_k nos pontos descritos no vetor x_k . A convergência é obtida minimizando a diferença entre os vetores g_k e g'_k .

O algoritmo genético (GONZALEZ, 2007) cria combinações a partir de duas estimativas iniciais (chamadas de pais). Dessa nova geração de indivíduos (chamados de filhos), dois deles são escolhidos por um processo de seleção (*survival of the fittest*), em que os indivíduos com melhor aproximação são escolhidos; no caso, com menor diferença entre os vetores g_k e g'_k . Os dois filhos se tornam pais de uma nova geração; e essa sequência é repetida até a convergência.

O processo chamado de *crossover* faz a recombinação dos dois pais gerando filhos, de maneira análoga ao processo biológico. Também de maneira análoga à biologia, aleatoriamente na geração dos filhos ocorrem mutações, em que parâmetros saem do intervalo dos parâmetros dos pais. A mutação introduz ruído ao processo, com intenção de que a convergência não fique presa em mínimos locais.

As estimativas iniciais (pais), se geradas de forma controlada ou totalmente aleatória, geralmente descrevem uma constelação de pontos com tamanho diferente da imagem de teste. Os conjuntos gerados que tiverem muitos pontos fora da imagem de teste não têm muita utilidade, pois estes pontos de fora não podem ser comparados com a imagem de teste. Diante disso, os pais passam por uma etapa de ajuste dos quatro últimos parâmetros do vetor c , que são os parâmetros de escala horizontal, escala vertical, translação horizontal e translação vertical respectivamente. O ajuste é feito de forma que os pais estejam centralizados e que o maior tamanho possível, sem pontos fora da imagem de teste.

Originalmente o algoritmo genético foi enunciado para os indivíduos representados como vetores de números binários, bem como os processos de *crossover* e mutação aplicados em conjuntos de *bits*. Porém também é possível representar os indivíduos como vetores de números reais (GONZALEZ, 2007).

O *crossover* adotado foi do tipo *blend*, que permite gerar filhos um pouco fora da faixa dos pais (DAMM, 2005) da forma:

$$filho = pai_1 + a(pai_1 - pai_2) \quad (3.3.1)$$

Em que a foi adotado neste trabalho no intervalo de $[-\alpha, \beta]$.

A mutação ocorre com probabilidade m_1 nos indivíduos filhos. Caso um indivíduo esteja no conjunto dos que sofrem mutação, cada parâmetro independente tem uma probabilidade que determina se este passa por mutação. Dessa forma, um indivíduo que esteja na faixa determinada por m_1 pode não sofrer mutação em nenhum dos parâmetros ou pode sofrer mutação em todos, em função das probabilidades internas.

O conjunto das probabilidades é:

$$\begin{pmatrix} m_2 \\ \vdots \\ m_2 \\ m_3 \\ m_3 \\ m_4 \\ m_4 \end{pmatrix} \quad (3.3.2)$$

Ou seja, os parâmetros do vetor c dados pela análise de componentes principais da equação (3.2.16) têm probabilidade m_2 de sofrer mutação. Os dois parâmetros de escala s_x e s_y adicionados ao vetor c têm probabilidade m_3 de sofrer mutação; e os parâmetros de translação t_x e t_y têm probabilidade m_4 de sofrer mutação.

No caso de ocorrer mutação de determinado parâmetro, é adicionado um valor aleatório ao parâmetro dentro das faixas abaixo:

$$\begin{pmatrix} [-f_1\sigma c^1, +f_1\sigma c^1] \\ \vdots \\ [-f_1\sigma c^t, +f_1\sigma c^t] \\ [-0,1f_2\sigma s_x, +0,1f_2\sigma s_x] \\ [-f_2\sigma s_y, +f_2\sigma s_y] \\ [-f_3l, +f_3l] \\ [-f_3h, +f_3h] \end{pmatrix} \quad (3.3.3)$$

Em que l e h são a largura e altura da imagem de teste respectivamente e f_x são fatores multiplicadores. A sensibilidade dos parâmetros do algoritmo genético será analisada no tópico 4.2.2. Os fatores foram diferenciados para os parâmetros s_x e s_y apesar de as probabilidades serem iguais, pois o parâmetro de escala horizontal s_x demonstrou variação muito maior do que a escala vertical s_y , isto porque a escala vertical só se manifesta quando existe rotação da face, e a escala horizontal existe para qualquer mudança de resolução de imagem. Portanto foi adicionado um fator 0,1 a escala horizontal s_x a fim de equilibrar os desvios padrão.

No cálculo da função custo, primeiramente são encontrados os novos valores x_k e g_k pelas equações:

$$x_k = \bar{x} + P_s W_s Q_s c_k \text{ e } g_k = \bar{g} + P_g Q_g c_k, \text{ onde } Q = \begin{bmatrix} Q_s \\ Q_g \end{bmatrix} \quad (3.3.4)$$

Que são obtidas invertendo-se as equações (3.2.15) e (3.2.16). As coordenadas dos pontos são, então, combinadas com os parâmetros s_x , s_y , t_x e t_y . Tendo isso, são contados os pontos fora da imagem, caso mais de 70% dos pontos estejam fora da imagem, o indivíduo é descartado. E, se a imagem gerada for menor que 30% da altura ou da largura da imagem de

teste, o indivíduo também é descartado. Isso é feito para garantir que a nova geração seja factível (DAMM, 2005).

O vetor de intensidades de nível de cinza gerado, g_k , é normalizado para a faixa $[0,255]$, de forma a facilitar a comparação com a escala da imagem de teste, pois esta é a faixa de valores possíveis de nível de cinza de uma imagem. A função custo é então calculada pela equação:

$$custo = \frac{\sum_j |I'_j - I_j|}{n'} \quad (3.3.5)$$

Onde I'_j é a intensidade de nível de cinza extraída na imagem de teste no ponto j ; I_j é a intensidade de nível de cinza do ponto j da imagem gerada, que equivale a g_k no ponto j e n' é o número de pontos dentro da imagem. Note que quanto menor o valor da função custo, melhor a coincidência das imagens.

Se o algoritmo convergir a um resultado aceitável antes das vinte gerações, o algoritmo é interrompido. Caso o número de gerações seja constante, o tempo de execução praticamente não varia com as execuções, mas, da forma que foi adotado, o tempo de execução traz uma variabilidade de resultados. Tais resultados são mostrados no capítulo 4.2.3.

4 RESULTADOS OBTIDOS

4.1 TESTE DO MODELO

O teste do modelo de AAM foi feito primeiramente com um modelo criado com fotos do presidente americano Barack Obama (NEGREIROS; BARRETTO; OKAMOTO, 2012). Na Figura 23 e na Figura 24 é possível ver o banco de dados treinado. As imagens originalmente coloridas foram reduzidas e convertidas para tons de cinza somente para exibição no documento, mas nota-se que as imagens têm formato e resolução diversas, algumas têm grande espaço de fundo, enquanto outras não. Isso é indiferente para o algoritmo utilizado, pois este busca informações somente no espaço delimitado pelos pontos marcados. As trinta imagens utilizadas no exemplo foram retiradas de sites de internet sem proteção autoral. O ideal para a base de dados é ter a maior variabilidade de faces, poses e expressões; no caso específico deste banco de dados, todas as faces são da mesma pessoa.

Figura 23: Banco de dados Obama (parte 1)

Figura 24: Banco de dados Obama (parte 2)

As imagens foram marcadas com 41 pontos cada uma conforme é possível verificar na Figura 25. Neste caso foi feito o teste com a face toda, não somente na região dos lábios, a fim de visualizar melhor o funcionamento do modelo. Foram marcados dois pontos nas narinas, por ser uma região de fácil rastreamento, afinal são dois pontos pretos no meio da face. Também foram marcados quatorze pontos na região dos olhos de cada imagem, a fim de captar melhor as expressões faciais. Oito pontos estão na região dos lábios, tanto para a leitura labial como para visualizar as expressões faciais. Por fim, dezessete pontos estão no contorno delimitando a região.

Figura 25: Marcação de pontos do modelo Obama

No processo de triangulação, a partir dos originais 41 pontos obtiveram-se 4.987 pontos. Como componentes principais de forma, apenas dois componentes representam 92,69% das variações. Já para os componentes principais de escala de cinza, dezesseis componentes representam 90,94% das variações. Onze componentes de aparência representam 91,95% das variações.

É possível observar a influência dos parâmetros internos do modelo variando-os e gerando imagens por meio da equação (3.3.4). Inicialmente é gerada a imagem média do modelo, quando o vetor c_i é nulo, como é visível na Figura 26.

Como o modelo gera valores de tons de cinza pontuais, não de toda a extensão, as imagens são geradas *pixel a pixel*, considerando o tom de cinza do ponto mais próximo do modelo, exceto pelos *pixels* fora da área delimitada, que são mantidos brancos.

Figura 26: Imagem média do modelo Obama

Na Tabela 1, é mostrada a variação do parâmetro b_{si} nas imagens geradas. Variou-se cada um dos componentes j do vetor em $-3\sigma b_{si}$ e $+3\sigma b_{si}$. Basicamente ocorre somente uma variação de pose, já que as imagens são da mesma pessoa.

j	$-3\sigma b_{si}$	$+3\sigma b_{si}$	j	$-3\sigma b_{si}$	$+3\sigma b_{si}$
1			2

Tabela 1: Variação de forma do modelo Obama

Já na Tabela 2, é visível a variação do parâmetro b_{gi} . Os componentes j do vetor foram variados em $-3\sigma b_{gi}$ e $+3\sigma b_{gi}$. Neste caso ocorrem variações nas expressões, pois a forma é mantida constante.

j	$-3\sigma b_{gi}$	$+3\sigma b_{gi}$	j	$-3\sigma b_{gi}$	$+3\sigma b_{gi}$
1			2
3			4
5			6
7			8
9			10
11			12

13			14
15			16

Tabela 2: Variação de nível de cinza do modelo Obama

Finalmente, na Tabela 3, vê-se a variação do parâmetro c_i em cada componente j do vetor na faixa de $-3\sigma_{c_i}$ e $+3\sigma_{c_i}$. Nota-se a variação do conjunto de aparência, ou seja, forma e nível de cinza.

j	$-3\sigma_{c_i}$	$+3\sigma_{c_i}$	j	$-3\sigma_{c_i}$	$+3\sigma_{c_i}$
1			2
3			4

Tabela 3: Variação de aparência do modelo Obama

4.2 TESTE DE CONVERGÊNCIA

Um segundo modelo foi criado com fotos de jornalistas e um apresentador, com cerca de vinte imagens de cada jornalista, totalizando 190 fotos. As imagens foram retiradas de sites de internet sem proteção autoral. Existem fotos de homens e mulheres, com maquiagem e sem, alguns com barba e/ou óculos. Os jornalistas são: Ana Paula Padrão, Boris Casoy,

Fátima Bernardes, José Luís Datena, Paulo Henrique Amorim, Patrícia Poeta, Roberto Cabrini, William Bonner e Zeca Camargo; e o apresentador e humorista Jô Soares,

As imagens foram marcadas com doze pontos como exibido na Figura 27. Foram marcados quatro pontos na região dos lábios, quatro pontos delimitando a região das bochechas, dois pontos no queixo e dois pontos nas narinas, pela facilidade de rastreamento, somando doze pontos em cada imagem. Uma região aumentada dos lábios foi utilizada a fim de melhorar o desempenho em especial em pessoas com barba, conforme descrito anteriormente.

Figura 27: Marcação de pontos do modelo de lábios

A partir da triangulação, os doze pontos originais geraram, no total, 1.361 pontos. Cinco componentes de forma representam 92,90% das variações, 31 componentes de nível de cinza representam 90,35% das variações e finalmente dezessete componentes de aparência representam 90,73% das variações.

A imagem média do modelo é mostrada na Figura 28.

Figura 28: Imagem média do modelo de lábios

Na Tabela 4, é mostrada a variação do parâmetro c_i em cada componente j do vetor na faixa de $-3\sigma c_i$ e $+3\sigma c_i$.

j	$-3\sigma c_i$	$+3\sigma c_i$	j	$-3\sigma c_i$	$+3\sigma c_i$
1			2		
3			4		
5			6		
7			8		
9			10		
11			12		

13			14
15			16
17

Tabela 4: Variação de aparência do modelo de lábios

4.2.1 Testes Iniciais

Testes iniciais foram executados a fim de observar o comportamento do erro da função custo e determinar um critério de parada do algoritmo genético. Primeiramente, testou-se o valor do erro da função custo com a imagem média do modelo, utilizando o vetor c_i nulo e os parâmetros de translação e escala ajustados à resolução da imagem, sendo verificado o erro obtido. O resultado está na Figura 29.

Figura 29: Solução ideal para a imagem média

Neste resultado o erro é de 0,186055. Nota-se que a função custo pode ter valores na faixa $[0,255]$, devido à escala de intensidade de nível de cinza. Observa-se que o valor

mínimo da função custo é ligeiramente maior que zero, devido a aproximações de cálculos em relação aos *pixels*, como a interpolação bilinear.

Testes iniciais foram executados com o algoritmo genético ainda não ajustado e alguns resultados foram observados. A sequência de convergência do algoritmo genético está na Figura 30:

Figura 30: Sequência de convergência

Na Figura 31 observa-se um resultado obtido pelo algoritmo genético com erro de 0,215855.

Figura 31: Convergência com erro de 0,215855 para a imagem média

Nota-se pela figura que a solução se aproxima muito da ideal.

Observa-se outro resultado na Figura 32, desta vez com erro de 0,386069.

Figura 32: Convergência com erro de 0,386069 para a imagem média

Nota-se que ainda assim visualmente a solução está bem próxima ao ideal.

No entanto, na Figura 33, tem-se um resultado que, visualmente, percebe-se que os pontos encontrados estão muito distantes dos desejados; nesta situação o erro foi 0,343982, menor do que a Figura 32.

Figura 33: Convergência com erro de 0,343982 para imagem média

Diante dos resultados iniciais, foi considerado como critério de parada um erro menor que 0,30. Dessa forma, sendo o erro menor que 0,30, garante-se que ocorreu a convergência correta do algoritmo. Entretanto, observa-se que um erro maior do que 0,30 pode significar também que o resultado é satisfatório.

4.2.2 Análise de Sensibilidade

Foi executada análise de sensibilidade para cada um dos parâmetros do algoritmo genético descrito no tópico 3.3. Cada parâmetro foi variado, e o erro de convergência

comparado. Primeiramente foi analisada a probabilidade de ocorrência de mutação em um indivíduo m_1 como se pode observar no Gráfico 1, a seguir.

Gráfico 1: Erro x probabilidade m_1 de ocorrência de mutação

Nota-se grande sensibilidade do algoritmo, e conforme a probabilidade aumenta melhora a convergência. Foi adotado o valor de 90% como ideal, pois próximo de 100% ocorre ligeiro aumento do erro. Possivelmente maior probabilidade de mutação orienta que a convergência do algoritmo não fique presa a um mínimo local.

Para a probabilidade m_2 de ocorrência de mutação em um parâmetro do vetor c_i relativo à análise de componentes principais dado pela equação (3.2.16), tem-se o Gráfico 2.

Gráfico 2: Erro x probabilidade m_2 de ocorrência de mutação

Pode-se observar pouca sensibilidade do algoritmo e uma faixa ideal entre 15% e 30%. Foi, portanto, adotado o valor de 20% como ideal.

No Gráfico 3 pode-se verificar a atuação da probabilidade m_3 de ocorrência de mutação em um parâmetro de escala do modelo.

Gráfico 3: Erro x probabilidade m_3 de ocorrência de mutação

Pequenas probabilidades claramente geram grande erro de convergência e probabilidades maiores que 40% tendem a estabilidade. Foi adotado como ideal o valor de 90%, pois está na região de mínimo da curva.

A seguir, no Gráfico 4, tem-se a influência da probabilidade m_4 de ocorrência de mutação, desta vez em um parâmetro de translação.

Gráfico 4: Erro x probabilidade m_4 de ocorrência de mutação

Nota-se que probabilidades menores que 30% têm desempenho melhor do algoritmo, porém sem um valor ideal para seguir. Foi adotado o valor de 10% como melhor opção.

O Gráfico 5 traz a variação do erro em função do fator de variação f_1 relativo aos parâmetros do vetor c_i , relativos à análise de componentes principais na equação (3.2.16).

Gráfico 5: Erro x fator de variação f_1 de mutação

Diferente dos parâmetros anteriores, os fatores f_x não têm limite claro, podendo assumir qualquer valor real de zero a infinito. Variações foram testadas até ser possível observar algum ponto de mínimo local. Com isso, este parâmetro apresentou grande sensibilidade ao algoritmo genético. Nota-se melhor opção na faixa de 10 a 20. Foi adotado como ideal o valor de 15.

O fator de variação f_2 referente à escala horizontal e vertical foi testado e a sensibilidade exibida no Gráfico 6.

Gráfico 6: Erro x fator de variação f_2 de mutação

Este fator demonstrou sensibilidade pequena, especialmente se considerado o ajuste de 0,1 no fator de escala horizontal conforme descrito no capítulo 3.2 diante da diferença entre os desvios padrão. Foi adotado o valor 1,0 como ideal.

Em relação à translação horizontal e vertical, tem-se o fator de variação f_3 , cuja análise de sensibilidade está descrita no Gráfico 7.

Gráfico 7: Erro x Fator de variação f_3 de mutação

Nota-se que este parâmetro exibiu a menor de todas as sensibilidades, ou seja, o valor tem pouca influencia no resultado final do algoritmo. No entanto, foi adotado como ideal o valor de 0,1.

Agora se analisam os parâmetros do *crossover blend* α e β no Gráfico 8 e Gráfico 9.

Gráfico 8: Erro x Parâmetro α do crossover blend

Gráfico 9: Erro x Parâmetro β do crossover blend

Estes parâmetros têm uma faixa pequena de valores possíveis, pois valores muito altos deixariam de caracterizar o *crossover*. Notou-se também pouca influência no resultado. Pela equação (3.3.1), supondo os valores $\alpha = 0$ e $\beta = 1$, o *crossover* tipo *blend* se descaracteriza e se torna o *crossover* aritmético, onde os filhos estão sempre em valores intermediários dos pais.

Observando os resultados, nota-se que a escolha do *crossover* tipo *blend* se justifica, pois apresenta resultados melhores do que $\alpha = 0$ e $\beta = 1$. Foram adotados os valores $\alpha = 2$ e $\beta = 2$. Mediante esses ajustes, foi então analisada a influência do número de indivíduos por geração, cujo resultado é exibido no Gráfico 10.

Gráfico 10: Erro x Quantidade de filhos por geração

Claramente, quanto maior a quantidade de filhos por geração, melhor a convergência. Porém a quantidade de indivíduos totais (filhos por geração e quantidade de gerações) tem grande influência sobre o tempo de execução. Então o tempo de execução em relação ao número de indivíduos por geração foi analisado e tem o resultado exibido no Gráfico 11.

Gráfico 11: Tempo de execução (s) x Quantidade de filhos por geração

Existe uma proporção direta entre o tempo de execução e a quantidade de filhos de geração. Isso impede que valores muito altos de filhos por geração sejam adotados. Foi adotado, portanto, um ponto de equilíbrio entre os dois: 200 indivíduos por geração. Valores pequenos, menores que 200, apresentam grande piora no erro de convergência, porém valores maiores não apresentam melhora que justifique maior tempo de execução.

Por fim, é analisada a sensibilidade do algoritmo em relação à quantidade de gerações. A quantidade de gerações também tem relação direta com o tempo de execução do algoritmo, então valores altos devem ser evitados.

No Gráfico 12, tem-se o erro médio obtido a cada geração do algoritmo genético; e no Gráfico 13 pode-se observar o mesmo erro, porém de 100 execuções diferentes.

Gráfico 12: Erro médio x Número de gerações

Gráfico 13: Erro x Número de gerações

Foi adotado como vinte o valor melhor de quantidade de gerações. Após vinte gerações, as execuções com grande erro acabam ficando presas em mínimos locais e não apresentam melhora significativa, mesmo com cinquenta gerações. Antes de vinte gerações, pode-se observar que o valor médio ainda apresenta erro muito grande, no entanto, após vinte gerações o valor de erro médio tende a estabilizar-se.

No Gráfico 13 é possível constatar a aleatoriedade do algoritmo genético, pois mostra que existe grande variabilidade nos resultados obtidos, dadas as mesmas condições.

4.2.3 Análise dos Resultados

Com o algoritmo genético ajustado, novos testes de convergência foram realizados com a imagem média do modelo. Foi possível observar pelos resultados que houve grande dependência da configuração inicial para a convergência.

Como estimativa inicial foram considerados vetores c_o múltiplos do desvio padrão σc , pois Cootes, Edwards e Taylor (1998) afirmam que a variabilidade de soluções do modelo se encontram no intervalo $[-3\sigma c, +3\sigma c]$, como descrito no tópico 2.1.4 e observado na Figura 19. Decidiu-se testar com as estimativas iniciais em uma faixa maior, de $-5\sigma c$ e $+5\sigma c$ e foram observadas as convergências com erro médio descrito no Gráfico 14.

Gráfico 14: Erro médio com pais $-5\sigma c$ e $+5\sigma c$ a cada iteração

Nota-se que não foram atingidos valores ideais, pois o menor erro ainda foi maior que 0,30. Um exemplo do resultado é mostrado na Figura 34.

Figura 34: Exemplo de resultado com pais $-5\sigma c$ e $+5\sigma c$

Para executar os testes, então as estimativas iniciais foram aproximadas da solução, com os pais $-0,7\sigma_c$ e $+0,7\sigma_c$. As estimativas iniciais (pais), bem como a imagem de teste podem ser observadas na figura 35. A imagem de teste é a mesma exibida na Figura 28.

Figura 35: Pais $-0,7\sigma_c$ e $+0,7\sigma_c$ (acima) e a imagem de teste (abaixo)

O erro médio para cada geração é mostrado no Gráfico 15.

Gráfico 15: Erro médio com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ a cada iteração

No Gráfico 16 tem-se os erros finais de cada uma das 100 execuções.

Gráfico 16: Erros finais com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução

Nota-se novamente uma variabilidade nos resultados finais, dado pela aleatoriedade do algoritmo genético. Os dois extremos de resultados obtidos com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ são mostrados na Figura 36 e Figura 37, com menor e maior erro obtido respectivamente.

Figura 36: Convergência com erro de 0,201062

Figura 37: Convergência com erro de 0,299886

Para melhor análise do resultado, define-se uma grandeza Erro de Posição da forma:

$$Erro_{posição} = \frac{\sum_i^{n_0} \sqrt{(x_i - x'_i)^2 + (y_i - y'_i)^2}}{n_0} \quad (4.2.1)$$

Onde n_0 é o número de pontos marcados manualmente, x_i e y_i são as coordenadas dos pontos na imagem gerada pelo modelo e x'_i e y'_i são as coordenadas dos mesmos pontos na imagem com solução ideal. Dessa forma, em vez de utilizar o erro da diferença dos vetores g_k e g'_k , que não têm comportamento muito intuitivo, utiliza-se um erro que de fato é o resultado buscado por este trabalho. Tal erro de posição é dado em *pixels*, ou seja, a distância que determina se os pontos estão próximos ou distantes da solução ideal. Para fins de referência, a imagem de teste tem tamanho de 100 por 66 *pixels*.

No Gráfico 17, tem-se os erros de posição dos testes com pais $-0,7\sigma_c$ e $+0,7\sigma_c$.

Gráfico 17: Erros de posição (*pixel*) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução

Os resultados apresentam erros de posição na faixa entre 1 a 4 pixels.

No Gráfico 19, é mostrado o tempo de execução de cada uma das 100 execuções, com os pais $-0,7\sigma_c$ e $+0,7\sigma_c$.

Gráfico 18: Tempo de execução (s) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$ de cada execução

Finalmente, podemos observar no Gráfico 19 a o comportamento do erro de posição em relação ao tempo de execução do algoritmo.

Gráfico 19: Erro de posição (*pixel*) x Tempo de execução (s) com pais $-0,7\sigma_c$ e $+0,7\sigma_c$

Nenhuma relação é visível, isto se deve a grande aleatoriedade do algoritmo genético, ou seja, um maior tempo de execução não gera um resultado melhor.

4.3 COMPARAÇÃO COM A PROPOSTA ORIGINAL

Para fins de validação da técnica desenvolvida, esta foi comparada com uma implementação da proposta original do AAM conhecida como AAM TOOLS.

O mesmo modelo de lábios descrito no item 4.2 foi criado no software AAM TOOLS, com o mesmo banco de dados de imagens e pontos marcados manualmente. Primeiramente observemos a imagem média do modelo de lábios gerada por este trabalho na Figura 38.

Figura 38: Imagem média do modelo de lábios

Agora na Figura 39 temos a mesma imagem média, porém gerada pelo software AAM Tools

Figura 39: Imagem média do modelo de lábios utilizando AAM TOOLS

Visualmente os dois resultados são bem próximos, excetuando-se pela resolução de pixels. Com esta comparação podemos validar que o modelo deste trabalho está sendo criado de forma análoga a proposta original.

Executando um teste de convergência na mesma imagem média da Figura 38, com os pais $-0,7\sigma_c$ e $+0,7\sigma_c$, respectivamente representados nas Figuras 40 e 41, do lado esquerdo os pontos representados na imagem média e do lado direito suas imagens geradas; temos a Figura 42 como exemplo de resultado típico.

Figura 40: $\text{Pai } -0,7\sigma_c$

Figura 41: $\text{Pai } +0,7\sigma_c$

Figura 42: Convergência típica

Executando o mesmo teste de convergência agora utilizando o AAM TOOLS, considerando como estimativa inicial os pontos representados na Figura 43 e na Figura 44 temos o resultado típico obtido.

Figura 43: Estimativa inicial utilizando AAM TOOLS

Figura 44: Convergência típica utilizando AAM TOOLS

Nota-se visualmente que, para os dados utilizados, o AAM TOOLS diverge da solução ideal, enquanto que a técnica desenvolvida neste trabalho apresenta melhor resultado, convergindo para a solução ideal.

5 CONSIDERAÇÕES FINAIS

Tem havido grande popularização de novas interfaces, como reconhecimento e geração de voz, rastreamento de gestos, telas *touchscreen*, etc.. No entanto, pessoas com deficiência auditiva ou de fala não podem utilizar interfaces de reconhecimento ou geração de voz.

Este projeto visou desenvolver uma técnica de rastreamento labial para aplicação de reconhecimento labial, obtendo parâmetros que descrevam o movimento dos lábios e outros pontos da face ao longo do tempo, com intenção de utilização desses parâmetros para o reconhecimento labial.

Muitas técnicas são descritas na bibliografia, tendo sido dada atenção especial neste estudo às técnicas *Snake*, *template*, Active Shape Model, Active Appearance Model e *thresholding*, que foram descritas e analisadas. A partir de métodos já disponíveis na literatura, este trabalho desenvolveu um processo baseado em Active Appearance Model (AAM): método estatístico que cria um modelo baseado em informações passadas manualmente ao método, para depois encaixar uma imagem sintetizada aplicando o modelo a uma imagem de teste.

Foi criado um modelo com base de dados com imagens de uma única pessoa e geradas imagens baseadas no modelo, alterando-se os parâmetros internos a fim de observar os efeitos.

Um segundo modelo foi criado com imagens de jornalistas brasileiros, a fim de testar a convergência do modelo utilizando um algoritmo genético. Uma análise de sensibilidade do algoritmo genético foi apresentada, assim como os resultados obtidos. Os resultados foram então comparados.

Para trabalhos futuros, a meta seria aperfeiçoar o desempenho do método para melhorar a convergência em imagens fora do modelo; bem como reduzir o custo computacional. Com isso, talvez seja possível que o rastreamento do movimento dos lábios seja executado com velocidade maior do que as imagens são geradas, em um caso de aplicação em tempo real.

REFERÊNCIAS

- AAM TOOLS. Disponível em http://www.isbe.man.ac.uk/~bim/software/am_tools_doc/index.html Acesso em 10 nov 2012
- BEAUMESNIL, B.; LUTHON, F. Real time tracking for 3D realistic lip animation. **Pattern recognition, 2006**. ICPR 2006. 18th International Conference 2006, vol. 1, pp. 219-22.
- BOUVIER, C.; COULON, P.Y.; MALDAGUE, X. Unsupervised lips segmentation based on ROI optimisation and parametric model. **Image Processing, 2007**. ICIP 2007. IEEE International Conference. Oct. 2007, vol. 4, pp. IV-301-IV-304.
- BREGLER, C., OMOHUNDRO, S.M., Nonlinear manifold learning for visual speech recognition. **Computer Vision, 1995**. Proceedings., Fifth International Conference on. Jun. 1995, pp. 494-9.
- CHAN, M.T.; ZHANG, Y.; HUANG, T.S. Real-time lip tracking and bimodal continuous speech recognition. **Multimedia signal processing, 1998**. IEEE Second Workshop on Dec. 1998, pp. 65-70.
- COOTES, T.F. et al. Active shape models – their training and application. **Computer Vision and Image Understanding**. Jan 1995, vol. 61, pp. 38-59.
- COOTES, T.F.; EDWARDS, G.J.; TAYLOR, C.J. Active apperance models. **ECCV '98 Proceedings., 5th European Conference on Computer Vision**. 1998, vol. 2, pp. 484-98.
- COOTES, T.F.; EDWARDS, G.J.; TAYLOR, C.J. Modelling object appearance using the grey-level surface. **Proceedings., 5th British Machine Vision Conference**. Sep. 1994, pp. 479-88.
- DARGHAM, J.A.; CHEKIMA, A. Lips detection in the normalised RGB colour scheme. **Information and Communication Technologies, 2006**. ICTTA '06. 2nd. Apr. 2006, vol. 1, pp. 1546-51.
- DAMM, R.B. **Um estudo dos parâmetros de controle de um algoritmo genético**. 2005. Dissertação (Mestrado em Matemática Aplicada) – Universidade de São Paulo. São Paulo. 2005.
- DE LA TORRE, F. et al. Eigenfiltering for flexible eigentracking (EFE). **Pattern Recognition, 2000. Proceedings...** 15th International Conference on. Sept. 2000, vol. 3, pp. 1106-9.
- DE LA TORRE, F.; YACOOB, Y.; DAVIS, L. A probabilistic framework for rigid and non-rigid appearance based tracking and recognition. **Automatic Face and Gesture Recognition, 2000. Proceedings**. Fourth IEEE International Conference on Mar. 2000, pp. 491-8.
- DIBEKLIOGLU, H.; SALAH, A.A.; GEVERS, T. A statistical method for 2-D facial landmarking. **Image Processing**. IEEE Transactions on 2012, pp. 844-58.

EDWARDS, G.J.; TAYLOR, C.J.; COOTES, T.F. Interpreting face images using active appearance models. **Automatic Face and Gesture Recognition, 1998. Proceedings...** Third IEEE International Conference on Apr 1998, pp. 300-5.

EVENO, N.; CAPLIER, A.; COULON, P.-Y.; A new color transformation for lips segmentation. **Multimedia Signal Processing, 2001.** IEEE Fourth Workshop on Oct. 2001, pp. 3-8.

GACON, P., COULON, P.-Y., BAILLY, G. Statistical active model for mouth components segmentation. **Acoustics, Speech, and Signal Processing, 2005. Proceedings...** ICASSP '05. IEEE International Conference on Mar. 2005, vol. 2, pp. ii/1021-ii/1024.

LI, Gang; WANG, M.; LIN, Ling. Extracting lip parameters in speech synthesis system driven by visual-speech. **Innovative Computing, Information and Control, 2006.** ICICIC '06. First International Conference on Aug. 2006, vol. 2, pp. 346-9.

GONZALEZ, T.F. **Handbook of approximation algorithms and metaheuristics.** 2007. Imprinta Boca Raton: Chapman & Hall/CRC.

GOSWAMI, B.; CHRISTMAS, W.J. ; KITTLER, J. Statistical estimators for use in automatic lip segmentation. **Visual Media Production, 2006.** CVMP 2006. 3rd European Conference on 2006, pp. 79-86.

GRAF, H.P.; COSATTO, E.; POTAMIANOS, M. Robust recognition of faces and facial features with a multi-modal system. **Systems, Man, and Cybernetics, 1997.** Computational Cybernetics and Simulation. 1997. IEEE International Conference on Oct. 1997, vol. 3, pp. 2034-9,

GUAN, Y.-P. Automatic extraction of lip based on wavelet edge detection. **Symbolic and Numeric Algorithms for Scientific Computing, 2006.** SYNASC '06. Eighth International Symposium on Sept. 2006, pp. 125-32.

HENECKE, M.E.; PRASAD, K.V.; STORK, D.G. Using deformable templates to infer visual speech dynamics. **Signals, Systems and Computers, 1994.** Conference Record of the Twenty-Eighth Asilomar Conference on Nov 1994, vol. 1, pp. 578-82.

KASS, M.; WITKIN, A.; TERZOPOULOS D. Snakes: active contour models. **International Journal of Computer Vision.** 1988, pp. 321-31.

LIEVIN, M.; LUTHON, F. Lip features automatic extraction. **Image Processing, 1998. ICIP 98. Proceedings...** International Conference on Oct. 1998, vol. 3, pp. 168-72.

LIEVIN, M.; LUTHON, F. Unsupervised lip segmentation under natural conditions. **Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings...** IEEE International Conference on. Mar. 1999, vol. 6, pp. 3065-8.

LUCEY, S.; SRIDHARAN, S.; CHANDRAN, V. Chromatic lip tracking using a connectivity based fuzzy thresholding technique. **Signal Processing and Its Applications, 1999. ISSPA '99. Proceedings...** Fifth International Symposium on Aug. 1999, vol. 2, pp. 669-72.

LUCEY, S.; SRIDHARAN, S.; CHANDRAN, V. Initialized eigenlip estimator for fast lip tracking using linear regression. **Pattern Recognition, 2000. Proceedings...** 15th International Conference on Sept. 2000, vol. 3, pp. 178-181

LUETTIN, J.; THACKER, N.A.; BEET, S.W. Visual speech recognition using active shape models and hidden Markov models. **Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Proceedings...** IEEE International Conference on May 1996, vol. 2, pp. 817-20.

LUO, REN C.; HUANG, CHUN Y.; HSIAO, CHIN C. Recognition of facial expressions using component-based Active Appearance Models for human-robot interactions. **IECON 2011. Proceedings...** 37th Annual Conference on IEEE Industrial Electronics Society. Nov. 2011, pp. 4244-9,

MENG, LI; CHEUNG, Y.-M. A novel motion based lip feature extraction for lip-reading. **Computational Intelligence and Security, 2008. CIS '08. International Conference on 2008, vol. 1, pp. 361-5.**

NEGREIROS, T.; BARRETTO, M.R.P.; OKAMOTO, J. Geração de poses de faces utilizando Active Appearance. **Computer on the Beach. 2012, pp. 365-74.**

NGUYEN, Q.D.; MILGRAM, M., Multi features active shape models for lip contours. **Wavelet Analysis and Pattern Recognition, 2008. ICWAPR '08. International Conference on 2008, vol. 1, pp. 172-6.**

NGUYEN, Q. D.; MILGRAM, M.; NGUYEN, T.-H.-L. Multi features models for robust lip tracking. **Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on. 2008, pp. 1333-7.**

OOI, W.C. et al. Effective lip localization and tracking for achieving multimodal speech recognition. **Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on 2008, pp. 90-3.**

PAN, X.; KONG, J.; LIEW, A.W.-C. Lip contour extraction based on support vector machine add support. **Image and Signal Processing, 2008. CISP '08. Congress on May 2008, vol. 3, pp. 697-700.**

PEREZ, J.F.G. et al. Lip reading for robust speech recognition on embedded devices. **Acoustics, Speech, and Signal Processing, 2005. Proceedings...** ICASSP '05). IEEE International Conference on Mar. 2005, vol. 1, pp. 473-6.

PETAJAN, E.; GRAF, H.P. Robust face feature analysis for automatic speechreading and character animation. **Automatic Face and Gesture Recognition, 1996. Proceedings...** Second International Conference on Oct. 1996, pp. 357-62.

POTAMIANOS, G. et al. Recent advances in the automatic recognition of audiovisual speech. **Proceedings...** IEEE. Sep. 2003, vol. 91, Issue 9, pp. 1306-26.

POTAMIANOS, G.; NETI, C. Improved ROI and within frame discriminant features for lipreading. **Image Processing, 2001. Proceedings...** 2001 International Conference on Oct. 2001, vol. 3, pp. 250-3.

RAMAGE, M.; LINDSAY, E. Wrapping snakes for improved lip segmentation. **Acoustics, Speech and Signal Processing, 2009. Proceedings...** ICASSP 2009. IEEE International Conference on 2009, pp. 1205-8.

REHMAN, S.; LI LIU; HAIBO LI. Lipless tracking and emotion estimation. **Signal-Image Technologies and Internet-Based System, 2007. SITIS '07. Third International IEEE Conference on. Dec. 2007, pp. 768-74. 2007a.**

REHMAN, S.; LI LIU; HAIBO LI. Lip localization and performance evaluation. **Machine Vision, 2007. ICMV 2007. International Conference on Dec. 2007, pp. 29-34. 2007b.**

RUI MA et al. Lip localization on multi-view faces in video. **Multimedia and Expo, 2007 Proceedings...** IEEE International Conference on Jul. 2007, pp. 1395-8.

SAITOH, T; MORISHITA, K.; KONISHI, R. Analysis of efficient lip reading method for various languages. **Pattern Recognition, 2008. ICPR 2008. 19th International Conference on. 2008, pp. 1-4.**

SHIN, J.; JUN, BONGJIN; KIM, DAIJIN. Robust two-stage lip tracker. **Signal Processing and Information Technology (ISSPIT), 2009. IEEE International Symposium on. 2009, pp. 265-70.**

STILLITTANO, S.; GIRONDEL, V.; CAPLIER, A. Inner and outer lip contour tracking using cubic curve parametric models. **Image Processing (ICIP), 2009. 16th IEEE International Conference on 2009, pp. 2469-72.**

SUGAHARA, K.; KISHINO, M.; KONISHI, R. Personal computer based real time lip reading system. **Signal Processing Proceedings...** WCCC-ICSP 2000. 5th International Conference on. Aug. 2000, vol. 2, pp. 1341-6.

VIOLA, P.; JONES, M. Robust real-time object detection. **Statistical And Computational Theories of Vision – Modeling, Learning, Computing, And Sampling, 2001. Second International Workshop on Jul 2001.**

UNIT BATUR, A.; MONSON, H.H. A novel convergence scheme for active appearance models. **Computer Vision and Pattern Recognition, 2003. Proceedings...** IEEE Computer Society Conference on. Jun. 2003, pp. I-359-I-366 vol.1.

WERDA, S.; MAHDI, W.; HAMADOU, B. Colour and geometric based model for lip localization: application for lip-reading system. **Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on Sep. 2007, pp. 9-14.**

YANG, J.; WAIBEL, A. A real-time face tracker. **Applications of Computer Vision, 1996. WACV '96., Proceedings...** 3rd IEEE Workshop on Dec. 1996, pp. 142-7.

YAU, W.C.; WEGHORN, H. Visual speech recognition and utterance segmentation based on mouth movement. **Digital Image Computing Techniques and Applications**. 9th Biennial Conference of the Australian Pattern Recognition Society on Dec. 2007, pp. 7-14.

YI WU, Rui Ma et al. Robust lip localization on multi-view faces in video. **Image Processing, 2007**. ICIP 2007. IEEE International Conference on. Oct. 2007, vol. 4, pp. IV-481-IV-484.

ZHANG, X.; MERSEREAU, R.M. Lip feature extraction towards an automatic speechreading system. **Image Processing, 2000. Proceedings...** 2000 International Conference on. Sept. 2000, vol. 3, pp. 226-9.

ZHIYONG WU; JIYING WU; MENG, H. M., The use of dynamic deformable templates for lip tracking in an audio-visual corpus with large variations in head pose, face illumination and lip shapes. **Chinese Spoken Language Processing, 2008**. ISCSLP '08. 6th International Symposium on. 2008, pp. 1-4