



UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
Campus de São José do Rio Preto

Carlos Eduardo Rossi Cubas da Silva

O desenvolvimento de um sistema de animação facial baseado em
performance e no uso de câmera RGB-D

São José do Rio Preto
2017

Carlos Eduardo Rossi Cubas da Silva

O desenvolvimento de um sistema de animação facial baseado em
performance e no uso de câmera RGB-D

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Campus de São José do Rio Preto.

Orientador: Dr. Antonio Carlos Sementille

São José do Rio Preto
2017

Silva, Carlos Eduardo Rossi Cubas da

O desenvolvimento de um sistema de animação facial baseado em performance e no uso de câmera RGB-D / Carlos Eduardo Rossi Cubas da Silva - São José do Rio Preto, 2017. 103f. : il., tabs.

Orientador: Dr. Antonio Carlos Sementille.

Dissertação (Mestrado) – Universidade Estadual Paulista “Júlio de Mesquita Filho”, Instituto de Biociências, Letras e Ciências Exatas..

1. Computação gráfica. 2. Processamento de imagens - Técnicas digitais. 3. Animação por computador. 4. Reconhecimento facial (Computação) 5. Movimento. 6. Algoritmos de computador. I. Dr. Antonio Carlos Sementille. III. Universidade Estadual Paulista “Júlio de Mesquita Filho”, Instituto de Biociências, Letras e Ciências Exatas.. II. Título.

CDU – 518.72:76

Carlos Eduardo Rossi Cubas da Silva

O desenvolvimento de um sistema de animação facial baseado em
performance e no uso de câmera RGB-D

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Campus de São José do Rio Preto.

Comissão Examinadora

Dr. Antonio Carlos Sementille

UNESP - Universidade Estadual Paulista "Júlio de Mesquita Filho", Bauru
Orientador

Dr. Alexandre Cardoso

UFU - Universidade Federal de Uberlândia

Dr. João Fernando Marar

UNESP - Universidade Estadual Paulista "Júlio de Mesquita Filho", Bauru

São José do Rio Preto

2 de fevereiro de 2017

Dedicatória

Dedico este trabalho à minha família, em especial a minha esposa Mirian e meu enteado Kaio, que estiveram ao meu lado nesta jornada, acompanhando tudo de perto. À minha mãe, Carmen, que me incentiva muito a continuar com meus estudos. Aos meus irmãos Luciana e Rodolfo e meu pai Joel (in memoriam).

Agradecimentos

Agradeço a todos que, direta e indiretamente, me ajudaram a concretizar este sonho. Em especial, a minha família que participou diretamente do processo e me apoiou em todos os momentos. Aos amigos do laboratório SACI, um agradecimento especial, por todo apoio e troca de experiências que tivemos em todo esse tempo, quase dois anos e meio, e, especialmente, às pessoas com quem eu tive mais contato, como o Ivan e Everton, sempre disponíveis para ajudar. Gostaria de agradecer, também, ao Tiago de Gaspari, a primeira pessoa que me ajudou nesta jornada, sendo um verdadeiro tutor nos primeiros dias de mestrado.

A todos os professores, o meu muito obrigado, pois compartilharam conhecimentos preciosos para o meu aprendizado e crescimento, principalmente o meu orientador, professor Dr. Antonio Carlos Sementille que, com paciência e sabedoria, soube, como poucos, me guiar neste caminho incrível, trazendo-me para este dia que foi a finalização deste trabalho. A ele, minha eterna gratidão.

*“O sucesso nasce do querer,
da determinação e persistência em se chegar
a um objetivo. Mesmo não atingindo o alvo, quem
busca e vence obstáculos, no mínimo fará coisas admiráveis.”
(José de Alencar)*

Resumo

Nas últimas décadas, o interesse quanto à captura de movimentos da face humana e à identificação de suas expressões com a finalidade de geração de animações faciais realistas, tem aumentado, tanto na comunidade científica quanto na indústria do entretenimento. A alta acurácia nesse processo é necessária, pois os humanos são treinados para identificar expressões faciais, detectando facilmente pequenas imperfeições na animação de uma face virtual. A animação facial baseada em performance é uma das técnicas utilizadas para gerar animações realistas principalmente em filmes. Com o surgimento de câmeras RGB-D de baixo custo, muitos sistemas de animação facial baseada em performance foram desenvolvidos, compartilhando muitos princípios fundamentais, mas com detalhes de implementação específicos. Estes sistemas consistem de uma fase de rastreamento de movimentos e identificação de expressões faciais seguido de um procedimento de redirecionamento de expressões. Neste sentido, ambientes modulares para a animação facial baseada em performance são extremamente úteis na incorporação de novos algoritmos de rastreamento com características de entrada e saídas padronizadas. Considerando o contexto exposto, o presente trabalho teve como principal objetivo a criação e validação de um ambiente com arquitetura modular para animação facial baseada em performance que utilizou uma câmera RGB-D para a captura dos movimentos faciais de um ator, bem como permitiu a incorporação dos principais algoritmos de rastreamento encontrados na literatura, visando o redirecionamento destes movimentos para uma face virtual humana diferente. As entradas e saídas deste ambiente foram padronizadas pelo uso de *blendshapes* (mistura de formas).

Palavras-chaves: Captura de movimentos faciais, *Blendshapes*, *retargeting*.

Abstract

In recent decades, interest in capturing human face movements and identifying expressions for the purpose of generating realistic facial animations has increased in both the scientific community and the entertainment industry. The high accuracy in this process is necessary because humans are trained to identify facial expressions, easily detecting small imperfections in the animation of a virtual face. Performance-based facial animation is one of the techniques used to generate realistic animations especially in movies. With the emergence of low-cost RGB-D cameras, many performance-based facial animation systems have been developed, sharing many fundamental principles but with specific implementation details. These systems consist of a phase of tracking movements and identifying facial expressions followed by an expression redirection procedure. In this sense, modular environments for performance-based facial animation are extremely useful in incorporating new tracking algorithms with standardized input and output characteristics. Considering the above context, the main objective of this work was the creation and validation of an environment with modular architecture for performance-based facial animation that used an RGB-D camera to capture the facial movements of an actor as well as allowed the incorporation of the main algorithms found in the literary, aiming the redirection of these movements to a different human virtual face. The inputs and outputs of this environment were standardized by the use of blendshapes.

Keywords: Capture facial movements, Blendshapes, retargeting.

Lista de ilustrações

Figura 2.1 – Desenvolvido pelos psicólogos Paul Ekman e Wallace Friesen, os <i>FACS</i> possuem trinta e três unidades de ação (<i>AUs</i>) (por exemplo, as rugas da testa é uma unidade de ação).	21
Figura 2.2 – Regiões de ocorrência dos <i>AUs</i>	22
Figura 2.3 – Localização dos marcadores faciais.	23
Figura 2.4 – Conjunto de pontos sugeridos pelo padrão MPEG-4.	23
Figura 2.5 – Medidas de distâncias utilizadas como unidades de animação no padrão MPEG-4.	24
Figura 2.6 – As diferenças de avatares. (a) Fotorrealista (© 2004 The Polar Express). (b) Cartoon (© 2006 Monsters House). Criaturas fantásticas (© 2007 Star Wars: The Clone Wars Cartoon)	25
Figura 2.7 – Evolução do <i>Rigging</i> facial desde 1970.	26
Figura 3.1 – Conjunto de expressões para o refinamento dos <i>blendshapes</i>	27
Figura 3.2 – Formas de <i>blendshape</i>	28
Figura 3.3 – <i>Sliders</i> para alteração das malhas.	28
Figura 3.4 – Matriz-vetor que corresponde às expressões do modelos de <i>blendshapes</i>	29
Figura 3.5 – Captura de uma face usando um dispositivo de captura 3D.	30
Figura 3.6 – Exemplo de modelo morfológico. (a) Foto de uma face 2D; (b) Modelo 3D gerado em posição frontal; (c) Modelo 3D gerado em posição lateral.	30
Figura 3.7 – Transferência de expressões entre modelos 3D.	31
Figura 3.8 – Expressão criada com muitos modelos, inutilizando o resultado final	33
Figura 4.1 – Alinhamento de nuvens de pontos a fim de aumentar a qualidade da captura.	35
Figura 4.2 – ICP baseado numa abordagem ponto a ponto.	36
Figura 4.3 – ICP baseado numa abordagem ponto a superfície.	37
Figura 4.4 – Exemplo de Distância de Hausdorff Modificada. (a) Imagem a ser encontrada; (b) Cena completa; (c) Imagem encontrada destacada em vermelho encontrada na cena.	38
Figura 4.5 – Projeção de dados de entrada x_n no componente principal u_1	39
Figura 5.1 – Os usuários podem modificar as expressões dos seus avatares através da expressão facial capturadas em tempo real.	42
Figura 5.2 – Exemplo de exposições interativas.	42
Figura 5.3 – Visão geral do método. À esquerda, é feita a captura das informações em RGB-D e 2D; Ao centro, as informações são processadas gerando suas expressões correspondentes; À direita, o redirecionamento.	43
Figura 5.4 – Construção dos <i>blendshapes</i> iniciais baseados em 23 <i>FACS</i>	44

Figura 5.5 – Visão geral do processo.	45
Figura 5.6 – Visão geral do processo.	45
Figura 5.7 – Visão geral do processo de rastreamento da face.	46
Figura 5.8 – Visão geral do método.	48
Figura 5.9 – Fase <i>offline</i> do método.	49
Figura 5.10 – Fase <i>online</i> do método.	50
Figura 6.1 – Arquitetura e <i>pipeline</i> do sistema desenvolvido.	52
Figura 6.2 – Os subsistemas da arquitetura desenvolvida.	53
Figura 7.1 – Etapas da geração de <i>blendshapes</i> com associações de marcadores faciais.	55
Figura 7.2 – <i>Software FaceGen Modeller 3.3</i>	56
Figura 7.3 – Interface de manipulação do <i>software FaceGen Modeller 3.3</i>	56
Figura 7.4 – Interface de manipulação do <i>software FaceGen Modeller 3.3</i>	57
Figura 7.5 – Identificação da posição dos marcadores faciais, em uma imagem 2D, com a câmera <i>RealSense</i>	58
Figura 7.6 – Processo de normalização e escala dos modelos 2D e 3D. (a) modelo normalizado; (b) aplicado um ajuste nos pontos 2D para que a escalas entre os modelos fique igual; (c) Pontos 2D ajustados para o modelo 3D usando como referência o ponto mais próximo da câmera, que é a ponta do nariz.	58
Figura 7.7 – Correspondência dos pontos na imagem 2D com o modelo 3D. (a) Modelo 3D com os pontos 2D. Não existe um posicionamento quanto à profundidade dos pontos; (b) Pontos 3D identificados no modelo.	59
Figura 7.8 – Erro no processo automatizado de ajuste de marcadores. (a) posição correta dos marcadores faciais; (b) captura automática na região da boca, aparentemente correta; (c) marcadores faciais ajustados de maneira errada.	60
Figura 7.9 – Marcadores faciais da região da boca ajustados manualmente.	60
Figura 7.10 – Subsistema de captura, processamento e redirecionamento.	61
Figura 7.11 – Capacete utilizado para as capturas dos vídeos na fase inicial do processo. (a) e (b), visão do capacete e o dispositivo para suporte da câmera <i>RealSense</i> ; (c) visão do ajuste da haste, possibilitando uma regulagem de distância da câmera e sua altura.	62
Figura 7.12 – Suporte da câmera <i>RealSense</i> na haste do capacete.	62
Figura 7.13 – Visão da interface do <i>software</i> que faz a captura das imagens, utilizando as bibliotecas de desenvolvimento da câmera <i>RealSense</i> . Do lado direito, a imagem com os marcadores faciais; do lado esquerdo, a imagem de profundidade.	63

Figura 7.14—Exemplos de arquivos exportados pelo <i>software</i> de captura. (a) arquivo <i>json</i> com as informações dos marcadores faciais; (b) imagem do quadro correspondente às informações do arquivo <i>json</i>	64
Figura 7.15—Ajuste dos pontos da face neutra do <i>dataset</i> e da captura.	65
Figura 7.16—Pontos ajustados entre a face neutra da captura e do <i>dataset</i>	66
Figura 7.17—Pontos centralizados, usando o ponto mais próximo da câmera que é o ponto localizado na frente do nariz.	67
Figura 7.18—Exemplo do ajuste dos pontos em função da face neutra. (a) pontos da captura; (b) pontos ajustados à face neutra do <i>dataset</i>	67
Figura 7.19—Exemplo do cálculo das distâncias entre os pontos da face capturada do ator com os <i>blendshapes</i>	68
Figura 7.20—Exemplo de aplicação de pesos nos <i>blendshapes</i>	69
Figura 8.1 – Dispositivos com captura de imagens RGB-D. À esquerda o <i>Kinect</i> da Microsoft, e à direita o <i>RealSense</i> da Intel.	71
Figura 8.2 – Marcadores faciais capturados pela câmera <i>RealSense</i> da Intel.	72
Figura 8.3 – Detecção da face por meio da câmera <i>RealSense</i> da Intel.	72
Figura 8.4 – Pontos selecionados para os testes, totalizando 52 marcadores faciais.	74
Figura 8.5 – Variação do erro entre o posicionamento dos marcadores faciais gerados no módulo de processamento em função dos capturados no módulo inicial para a quantidade de <i>blendshapes</i> selecionados.	75
Figura 8.6 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 25% do <i>dataset</i> . (a) Distância Euclidiana; (b) ICP; (c) DHM; (d) PCA.	76
Figura 8.7 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 50% do <i>dataset</i> . (a) Distância Euclidiana; (b) ICP; (c) DHM; (d) PCA.	78
Figura 8.8 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 75% do <i>dataset</i> . (a) Distância Euclidiana; (b) ICP; (c) DHM; PCA.	80
Figura 8.9 – Visualização, para cada algoritmo, do quadro com maior erro utilizando 100% do <i>dataset</i> . (a) Distância Euclidiana; (b) ICP; (c) DHM; PCA.	82
Figura 8.10—Médias dos erros por algoritmo de rastreamento.	84
Figura 8.11—Desvio padrão do erro por algoritmo, considerando-se a quantidade de <i>blendshapes</i>	85
Figura 8.12—Tempo gasto por algoritmo, quadro a quadro, para a quantidade de <i>blendshapes</i> selecionados.	86
Figura 8.13—Resultado obtido pelos algoritmos em função do tempo de processamento.	87

Lista de tabelas

Tabela 1	– Médias (em milímetros) dos erros por algoritmo de rastreamento dos conjuntos de <i>blendshapes</i>	84
Tabela 2	– Desvio padrão do erro por algoritmo, considerando-se a quantidade de <i>blendshapes</i>	84
Tabela 3	– Médias dos tempos de processamento dos quadros para cada algoritmo.	85

Lista de abreviaturas e siglas

2D	<i>Bi-Dimensiona</i>
3D	<i>Tri-Dimensional</i>
AAM	<i>Active Appearance Model</i>
AUs	<i>Action Units</i>
DE	<i>Distância Euclidiana</i>
DH	<i>Distância de Hausdorff</i>
DHM	<i>Distância de Hausdorff Modificada</i>
FACS	<i>Facial Action Coding System</i>
FFD	<i>Free Form Deformer</i>
FAPs	<i>Facial Animation Parameters</i>
FDPs	<i>Facial Definition Parameters</i>
FPS	<i>Frame per second</i>
FPs	<i>Feature Points</i>
FAPU	<i>Facial Animation Parameter Units</i>
FCP	<i>Facial Characteristic Point</i>
ICP	<i>Iterative closest point</i>
JSON	<i>JavaScript Object Notation</i>
KLT	<i>Karnuhen-Loève transform</i>
MPPCA	<i>Probabilistic Principal Component Analyzers</i>
MIDI	<i>Musical instrument device interface</i>
MPEG-4	<i>Moving Picture Experts Group</i>
NURBS	<i>Now-Uniform Rational B-splines</i>
OPF	<i>Optimum Path Forest</i>

PCA	<i>Principal Componentes Analysis</i>
RGB	<i>Red, Green, Blue</i>
RGB-D	<i>Red, Green, Blue and Depth</i>
RBF	<i>Radial Basis Function</i>
WPCA	<i>Weighted Principal Component Analysis</i>

Sumário

1	INTRODUÇÃO	17
1.1	O Problema	18
1.2	Objetivos	19
1.2.1	Objetivo geral	19
1.2.2	Objetivos específicos	19
1.2.3	Organização da dissertação	19
2	EXPRESSÕES FACIAIS E <i>RIGGING</i> FACIAL	21
2.1	Expressões universais ou unidades de ação	21
2.2	Marcadores faciais	22
2.3	Conceito de <i>Rigging</i>	23
2.4	Domínios da aplicação	24
3	BLENDSHAPES	27
3.1	Conceito	27
3.2	Criação de expressões com <i>blendshapes</i>	28
3.3	Construção de <i>blendshapes</i>	30
3.4	Técnicas de animação e interatividade	31
3.5	Problemas no uso de <i>blendshapes</i> para a animação facial	32
4	ALGORITMOS DE RASTREAMENTO DE FACES	34
4.1	Distância Euclidiana	34
4.2	<i>Iterative Closest Point</i>	35
4.2.1	ICP ponto a ponto	35
4.2.2	ICP ponto a superfície	36
4.3	Distância de Hausdorff Modificada	37
4.4	<i>Principal Component Analysis</i>	38
5	ANIMAÇÃO FACIAL BASEADA EM PERFORMANCE	41
5.1	Conceito	41
5.2	Domínios da animação facial baseada em performance	41
5.3	Trabalhos relacionados	42
5.3.1	Método de Li et al. (2013)	43
5.3.1.1	Visão geral do método	44
5.3.2	Método de Li et al. (2013)	44
5.3.2.1	Visão geral do método	45

5.3.3	Método de Behrens et al. (2013)	46
5.3.3.1	Visão geral do método	47
5.3.4	Método de Weise et al. (2011)	47
5.3.4.1	Visão geral do método	48
5.4	Desafios da animação facial baseada em performance	50
6	SISTEMA DE ANIMAÇÃO FACIAL BASEADO EM PERFORMANCE	51
7	IMPLEMENTAÇÃO DO PROTÓTIPO DO SISTEMA	54
7.1	Subsistema de geração de <i>dataset</i> de <i>blendshapes</i>	54
7.1.1	Geração dos <i>blendshapes</i> das faces	54
7.1.2	Inclusão de marcadores faciais	54
7.1.3	Ajuste dos marcadores faciais	57
7.1.4	Ajuste dos marcadores faciais da região da boca	59
7.2	Subsistema de captura, processamento e redirecionamento	60
7.2.1	Captura	60
7.2.2	Processamento e a geração dos pesos dos <i>blendshapes</i>	63
7.2.3	Redirecionamento	66
8	TESTES E ANÁLISE DE RESULTADOS	70
8.1	Ambiente experimental	70
8.1.1	Componentes de <i>Hardware</i>	70
8.1.1.1	Computador	70
8.1.1.2	Câmera <i>RealSense</i>	71
8.1.2	Componentes de <i>Software</i>	72
8.2	Experimentos realizados	73
8.2.1	Resultados obtidos na comparação dos algoritmos	74
8.2.1.1	Resultados obtidos com relação ao tempo de processamento de cada algoritmo	85
9	CONCLUSÕES	88
	REFERÊNCIAS	90
A	APÊNDICE, O <i>DATASET</i> COMPLETO	95

1 Introdução

Um campo de pesquisa muito ativo na área de Computação Gráfica é o da geração de modelos da face humana visando à criação de animações faciais realistas. Diversas são as aplicações que podem se beneficiar dos avanços neste campo, tais como: filmes para cinema e televisão, *videogames*, videoconferência com utilização de avatares, planejamento de cirurgia facial entre outras.

Na animação de personagens virtuais, a reprodução acurada dos movimentos faciais tem uma importância crítica, visto que é uma das principais fontes de informação emocional. Segundo Fratarcangeli (2013), a grande complexidade e sofisticação da estrutura da cabeça humana aumentam a dificuldade da reprodução de uma animação facial convincente. Uma alta acurácia é necessária porque os humanos são treinados para observar e decodificar expressões faciais desde o nascimento, tornando-os especialistas na detecção de pequenos erros na animação de uma face virtual.

As técnicas de animações faciais podem ser dirigidas à fala (HONG; WEN; HUANG, 2002) ou dirigidas à performance (ZHANG et al., 2006; ZENG et al., 2012).

Neste projeto, o foco encontra-se na animação facial baseada em performance. As técnicas baseadas em performance fornecem um meio essencial para a geração de animações faciais realistas, cujos resultados impressionantes podem ser observados em filmes recentes.

Normalmente, os sistemas de animação baseados em performance consistem em um módulo de captura de performance facial e um módulo de transferência de movimento. Para a captura da performance facial, vários sistemas utilizam diversas câmeras e um grande número de marcadores faciais nos atores. Apesar de alcançarem bons resultados, o uso desses marcadores pode não ser prático, além de intrusivo para os atores. Em adição, esses sistemas normalmente requerem muita intervenção manual (LUO et al., 2014).

Uma compensação fundamental em todos os sistemas é a relação entre a qualidade dos dados adquiridos e a complexidade da configuração da aquisição. Em uma extremidade do espectro estão os sistemas concebidos para uma maior acurácia possível que levam a avatares virtuais impressionantes, adequados para a produção do filme. Por causa da sua robustez, técnicas baseadas em marcadores são amplamente utilizadas para a animação facial em tempo real e, geralmente, fornecem parâmetros de movimento suficientes para um convincente redirecionamento para modelos de criaturas não humanas ou personagens de *videogames*.

Para a digitalização realista de rostos humanos, abordagens que não utilizam marcadores como *scanners* 3D em tempo real são, geralmente, mais vantajosas devido à

sua capacidade para capturar a dinâmica de escala fina (por exemplo, rugas e dobras). Todos esses métodos envolvem sensores altamente especializados e/ou um ambiente de estúdio controlado. No entanto, a disponibilidade recente de equipamentos com câmeras RGB-D de baixo custo, mas de boa resolução de profundidade, tem mudado este panorama, viabilizando a criação de ambientes voltados ao usuário comum.

Portanto, a principal motivação para este projeto de pesquisa é o fato da criação de ambientes de animação facial baseado em performance que ofereçam a usabilidade, desempenho e robustez, a partir de dispositivos RGB-D de baixo custo ainda ser um problema científico e tecnológico em aberto.

1.1 O Problema

Visto que a captura dos movimentos faciais e a animação baseada em performance são áreas de pesquisa ativa nos últimos anos, existe uma grande quantidade de diferentes sistemas de aquisição e *pipelines* de processamento que compartilham muitos princípios fundamentais, bem como detalhes de implementação específicos.

Frequentemente, a face digital que necessita ser animada não é uma réplica digital do ator. Nesse caso, é necessário realizar um processo que adapte a performance gravada à face do personagem alvo. Esse processo é chamado de redirecionamento (*retargeting*) ou mapeamento cruzado (*cross-mapping*).

Uma questão importante para o redirecionamento é a escolha da parametrização do modelo facial (ou *rig*). Um *rig* fornece, portanto, uma parametrização das expressões faciais de uma face digital. Ele descreve as expressões faciais como um pequeno número de parâmetros e limita o escopo de expressões ao escopo desses parâmetros. Existem muitas abordagens diferentes para a parametrização de uma face digital, sendo que as principais são: *blendshape* (ou mescla de formas), PCA (*Principal Component Analysis*) e malha bruta. Dentre essas abordagens, a mais usada pela indústria e pelos *softwares* modeladores é o *blendshape*, motivo pelo qual foi adotado no presente trabalho.

Os *blendshapes* possibilitam uma parametrização linear das deformações da face. O espaço das faces em potencial é o espaço linear abrangido pelos *blendshapes* (ou uma porção deste espaço, se os pesos forem limitados). O redirecionamento é reduzido à tarefa de estimar um conjunto de pesos de mesclagem (*blending weights*) para a face destino em cada quadro (*frame*) da animação de origem. As expressões faciais são representadas como uma soma de pesos de malhas de *blendshapes*. Um modelo de *blendshape* fornece uma representação compacta do espaço das expressões faciais, reduzindo significativamente a dimensionalidade do problema de otimização.

A animação facial baseada em performance tipicamente consiste de uma fase de

rastreamento não rígida (muitas vezes com um modelo de template paramétrico) seguido de um procedimento de redirecionamento de expressão.

Por sua vez, o desenvolvimento de novos algoritmos de rastreamento dependem, normalmente, da implementação de todas as etapas de um *pipeline* de processamento. A criação de um sistema de arquitetura modular para a animação facial baseada em performance que permitisse a incorporação de novos algoritmos de rastreamento, com características de entrada e saídas padronizadas, traria uma contribuição a essa área de pesquisa.

1.2 Objetivos

1.2.1 Objetivo geral

O objetivo geral deste trabalho é criar e validar um ambiente com arquitetura modular para a animação facial baseada em performance que utilize, como forma de captura dos movimentos faciais de um ator, uma câmera RGB-D, bem como permitir a incorporação de diferentes algoritmos de rastreamento. O ambiente deve realizar a transferência de expressões faciais de um usuário para um modelo de face humana diferente. O *pipeline* genérico deste ambiente deve basear-se no uso de *blendshapes*, tanto para a obtenção dos modelos de expressão genéricos, quanto para a animação do modelo da face virtual de saída (alvo do redirecionamento).

1.2.2 Objetivos específicos

São objetivos específicos deste trabalho:

- a) a geração de um *dataset* de *blendshapes* de expressão genéricos; e
- b) a validação do rastreamento por meio da implementação de algoritmos selecionados na literatura, especificamente os propostos por Li et al. (2013), Xu et al. (2014), Weise et al. (2011) e Braun (2014).

1.2.3 Organização da dissertação

O restante desta dissertação está organizada nos seguintes capítulos:

São apresentadas no capítulo 2 as expressões faciais, o padrão MPEG-4 para animação facial e o conceito de *rigging* facial.

No capítulo 3 há uma discussão sobre o uso de *Blendshapes*.

Alguns algoritmos utilizados para a animação facial baseada em performance encontrados na literatura e suas aplicações são apresentados no capítulo 4.

Os conceitos de animação facial dirigida a performance, o redirecionamento facial, seus domínios, seus desafios e alguns trabalhos na área são descritos no capítulo 5.

No capítulo 6 são levantados os detalhes do sistema de animação facial desenvolvido.

A implementação do método de animação dirigida a performance é apresentada no capítulo 7. Cada etapa é detalhada em função de suas características, os algoritmos utilizados e as soluções adotadas para os respectivos problemas encontrados durante a sua implementação.

Os resultados obtidos são detalhados no capítulo 8, a partir de testes realizados utilizando-se a implementação do método proposto.

Por fim, no capítulo 9, são apresentadas as conclusões sobre o trabalho e seus desdobramentos futuros.

2 Expressões faciais e *Rigging* facial

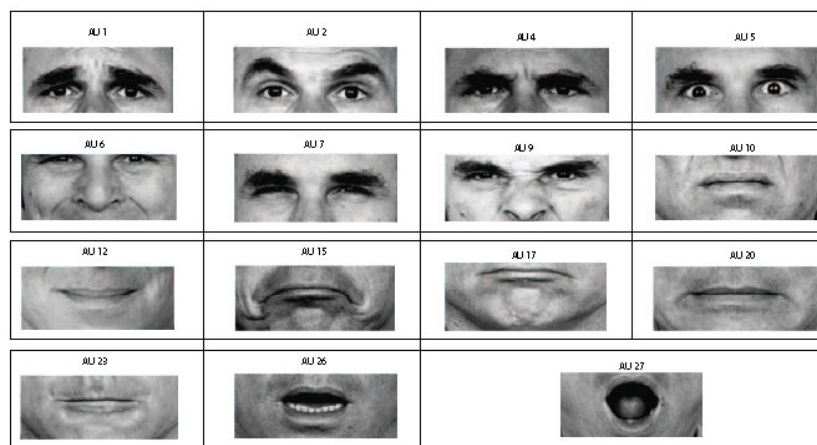
Neste capítulo, são abordados aspectos sobre as expressões faciais e suas representações.

2.1 Expressões universais ou unidades de ação

As expressões faciais têm despertado um grande interesse nos pesquisadores em várias áreas do conhecimento. Para Fang et al. (2011), existem dois modelos principais de expressões faciais: os baseados em mensagens e os baseados em sinais. Os baseados em mensagens têm o foco na interpretação de padrões e classifica as expressões em um número predefinido de categorias chamadas de expressões universais. São elas: neutro, raiva, nojo, medo, felicidade, tristeza e surpresa. Já o modelo baseado em sinais estuda as deformações faciais.

As deformações faciais foram categorizadas e propostas por Ekman e Rosenberg (1997), por meio de um sistema de codificação de ações faciais (*FACS – Facial Action Coding System*). Ekman e Rosenberg (1997) dividiram a face em diferentes grupos musculares, chamados Unidades de Ação (*AU – Action Unit*). Um exemplo pode ser visto na Figura 2.1.

Figura 2.1 – Desenvolvido pelos psicólogos Paul Ekman e Wallace Friesen, os *FACS* possuem trinta e três unidades de ação (*AUs*) (por exemplo, as rugas da testa é uma unidade de ação).

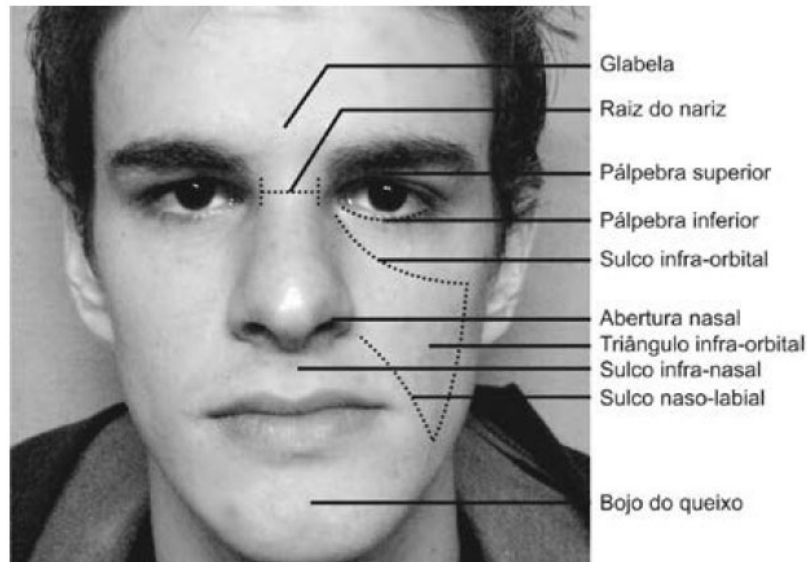


Fonte: Velusamy et al. (2011).

Segundo Ekman (2002 apud MARAR et al., 2011), existem basicamente dez regiões da face onde os *AUs* ocorrem: glabella, raiz do nariz, pálpebra superior e inferior, sulcos

infraorbital, infranasal e nasolabial, triângulo infraorbital (bochechas), e bojo do queixo, como pode ser visto na Figura 2.2.

Figura 2.2 – Regiões de ocorrência dos *AUs*.



Fonte: Marar et al. (2011).

Em seu trabalho Mohammadi, Fatemizadeh e Mahoor (2015) mostraram que as expressões faciais, incluindo as expressões universais, podem ser modeladas por um único *AU* ou uma combinação de *AUs*. Por exemplo, a expressão de alegria é a combinação do *AU6* e *AU12*.

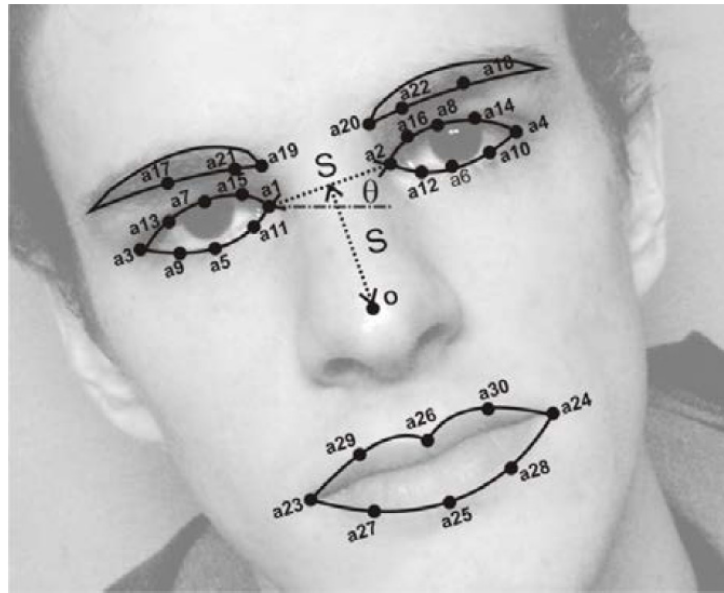
2.2 Marcadores faciais

Marcadores faciais denominados *FCP* (*Facial Characteristic Point*) são pontos que marcam áreas de destaque na face, como olhos, sobrancelhas, boca e nariz. Eles são usados também para delimitar o contorno da face. Um exemplo pode ser visto na Figura 2.3 (MARAR et al., 2011).

O MPEG-4 (*Moving Picture Experts Group*) é um padrão utilizado para animação facial e sugere um conjunto de 84 pontos localizados na face. Ele foi desenvolvido com base no sistema de codificações de ações de Ekman. A Figura 2.4 mostra estes pontos. Um subconjunto desses pontos são usados como pontos de controle para os 68 parâmetros de animação (Facial Animation Parameters – FAPs) também definidos pelo padrão MPEG-4 (BRAUN, 2014).

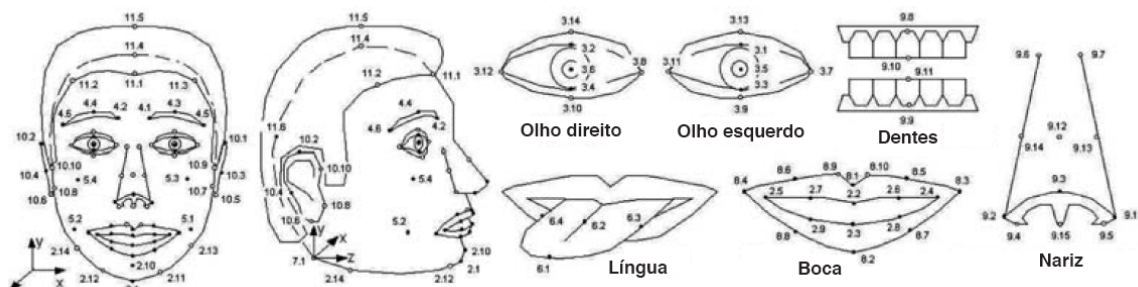
De acordo com Braun (2014), os FAPs são codificados através de valores numéricos normalizados por um conjunto de unidades baseadas nas distâncias entre alguns pontos característicos principais da face, chamadas FAPU (*Facial Animation Parameter Units*).

Figura 2.3 – Localização dos marcadores faciais.



Fonte: Marar et al. (2011).

Figura 2.4 – Conjunto de pontos sugeridos pelo padrão MPEG-4.



Fonte: Braun (2014 apud PANDZIC; FORCHHEIMER, 2002).

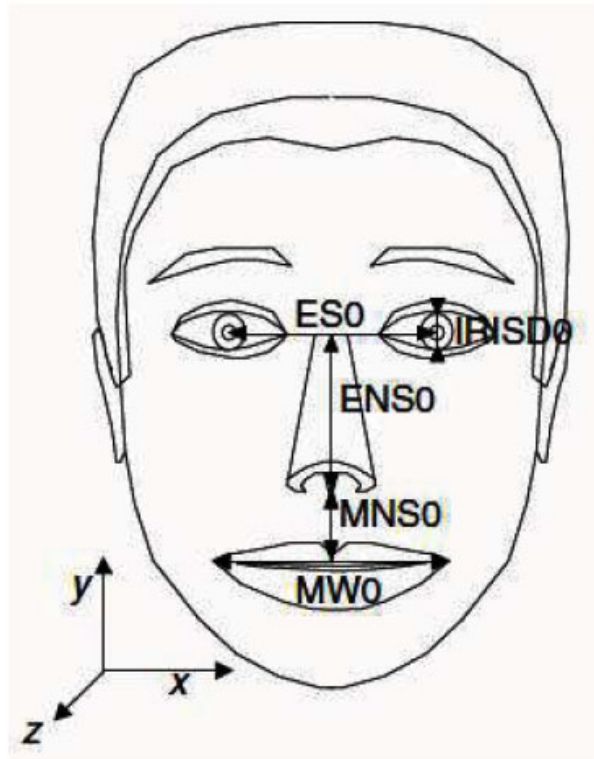
Com essa normalização, é possível animar faces com diferentes tamanhos, proporções e número de polígonos. A Figura 2.5 apresenta essas distâncias e unidades.

2.3 Conceito de *Rigging*

Rigging é o processo de tornar um personagem de computador inanimado e estático em um personagem que um animador pode editar criando os movimentos (FALK et al., 2004), .

Orvalho et al. (2012) definiram que *Rigging* é o processo de configuração de um grupo de controles para movimentar um personagem 3D como se fosse cordas de uma marionete. Citaram esta etapa como sendo um trabalho pesado e muito demorado pois, para cada personagem 3D, existe a necessidade de novas configurações.

Figura 2.5 – Medidas de distâncias utilizadas como unidades de animação no padrão MPEG-4.



Fonte: Pandzic e Forchheimer (2003).

A configuração desses controles na face é um trabalho difícil, pois a musculatura dessa área é complexa e gerar controles realísticos é um trabalho quase artesanal. O responsável pelo processo de *rigging* é o animador ou diretor técnico de animação. Ele é responsável em definir os parâmetros, o controle e as regras que o personagem virtual terá, garantindo todos os movimentos exigidos pelo processo.

2.4 Domínios da aplicação

Segundo Stoiber, Segulier e Breton (2009), a face é capaz de reproduzir, consciente ou inconscientemente, uma infinidade de variações expressivas de forma sutil, gerando uma vasta gama de expressões. A mesma ideia se aplica para o rosto de um personagem virtual, em que as características faciais e desempenho são fundamentais para a credibilidade do personagem, embora isso seja um trabalho difícil, pois os humanos são capazes de identificar comportamentos não naturais devido à familiaridade e sensibilidade da aparência facial.

A indústria do cinema e *games* são consumidores dessa tecnologia e desenvolvem sistemas de animação facial, tentando produzir expressões faciais que sejam convincentes

e realistas. Essas configurações nos personagens 3D, geralmente começam nos estágios iniciais da produção, sendo comum que esse processo seja refeito durante o filme ou um *game*, pois novas necessidades de movimentos surgem a cada momento.

Miranda (2008) cita que, este tipo de aplicação pode ser usada no entretenimento, psicoterapia, criminologia e outros. No seu trabalho, ele aborda o problema do autismo, que é um distúrbio onde os pacientes não reconhecem emoções. Ele propôs um sistema para ensinar as pessoas com esse distúrbio a reconhecerem emoções através de avatares que respondem às emoções humanas em tempo real. Exemplos destas interações podem ser vistos na Figura 2.6.












Figura 2.6 – As diferenças de avatares. (a) Fotorrealista (© 2004 The Polar Express). (b) Cartoon (© 2006 Monsters House). Criaturas fantásticas (© 2007 Star Wars: The Clone Wars Cartoon)



Fonte: Miranda (2008).

Na indústria do entretenimento, as aplicações podem ser divididas em sistemas *offline* e sistemas interativos em tempo real. Os sistemas *offline* são usados principalmente para longas-metragens, efeitos visuais ou transmissão de televisão. Exigem alto realismo e grande nível de detalhes no personagem. Já os sistemas interativos em tempo real como *videogames*, realidade virtual e bonecos virtuais exigem um balanço entre credibilidade e computação rápida (ORVALHO et al., 2012). Exemplos para os sistema podem ser vistos na Figura 2.7.

Figura 2.7 – Evolução do *Rigging* facial desde 1970.

	<i>Offline</i>	Tempo Real
1970	 <p>(a)</p> <p>Parke's: modelo facial parametrizado</p>	<p>Neste período não são conhecidos trabalhos de <i>rigging</i> facial em tempo real</p>
1980	 <p>(b)</p> <p>Tony de Peltrie: o primeiro filme de curta animação a usar um modelo facial parametrizado</p>	 <p>(c)</p> <p>Mike the Talking Head: primeiro teatro de fantoches virtuais em tempo real</p>
1990	 <p>(d)</p> <p>Toy Story: o primeiro filme a introduzir os doze princípios da animação</p>	 <p>(e)</p> <p>Half-Life (1998): usa animação facial em 3D</p>
2000	 <p>(f)</p> <p>Gollum: personagem realista em um papel de liderança em um filme de ação ao vivo</p>	 <p>(g)</p> <p>Doom 3 (2004): <i>rigging</i> facial à base de estruturas ósseas para videogames</p>
2011	 <p>(h)</p> <p>As aventuras de Tin Tin (2011): considerado o estado atual da arte como captura e performance em um filme de animação</p>	 <p>(i)</p> <p>The Samaritan Demo: considerado o estado atual na época para animação facial em tempo real</p>
2015	 <p>(j)</p> <p>Como Treinar seu Dragão II: considerado o melhor filme animado de 2015</p>	 <p>(k)</p> <p>Jogo Game Of Thrones: baseado no livro <i>Iron from ice</i>.</p>

Fonte: Adaptado de Orvalho et al. (2012).

3 Blendshapes

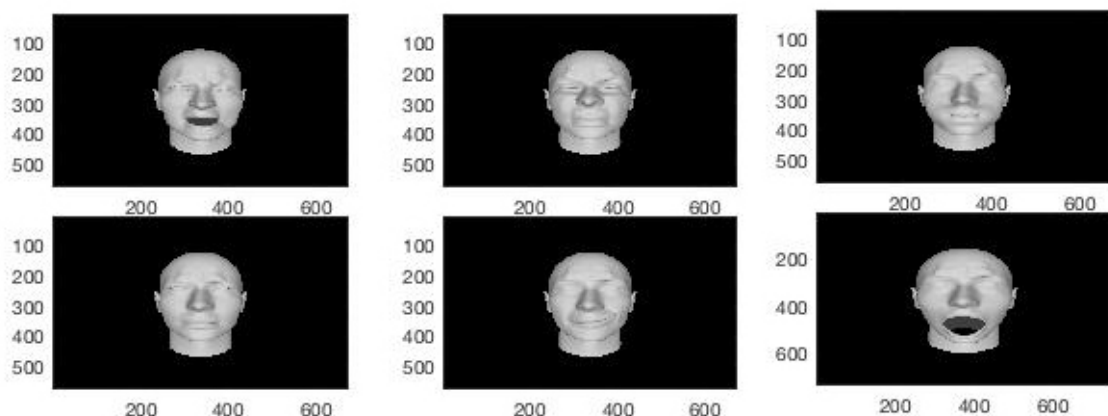
Neste capítulo são abordados os aspectos mais importantes ligados aos *blendshapes*, utilizados neste trabalho para a parametrização de faces digitais.

3.1 Conceito

O termo *blendshape* foi introduzido pela indústria gráfica na década de 80 quando se tornou popular em *softwares* comerciais. Lewis et al. (2014) definiu *blendshapes* como sendo modelos faciais em que os vetores representam expressões faciais individuais. Isto consiste em criar poses da face em várias malhas. Cada malha é designada a uma forma. Uma das malhas é a forma base enquanto que as outras são chamadas de formas alvo. A diferença entre a forma base e a forma alvo é representada por vetores de configuração. Cada vetor corresponde à diferença entre a base e o alvo (NENDYA; YUNIARNO; GANDANG, 2014). Para Braun (2014), *blendshapes* são matrizes escalares que representam o grau de combinação entre duas formas do mesmo modelo 3D.

Um modelo de *blendshape* gera uma pose facial como uma combinação linear de um número de expressões faciais, os *blendshapes* "alvos". Ao variar os pesos da combinação linear, uma gama de expressões faciais podem ser expressas com pequeno esforço computacional. O conjunto de formas pode ser estendido conforme desejado para refinar a gama de expressões que o personagem pode produzir (Figura 3.1) (LEWIS et al., 2014).

Figura 3.1 – Conjunto de expressões para o refinamento dos *blendshapes*.



Fonte: elaborada pelo autor.

As expressões alvo podem ser criadas por artistas ou adquiridas por meio de dispositivos de captura de alta resolução como scanners 3D. Alguns trabalhos, como o de

Sumner e Popović (2004), apresentam métodos para a transferência de expressões faciais entre uma expressão esculpida na forma neutra em um *blendshape* para um conjunto de expressões pré-existentes.

Um novo *blendshape* pode ser completamente criado usando esta técnica (Figura 3.2) entretanto, é necessária a criação de um grande número de *blendshapes* para prover os controles em todas as regiões da face. *Stuart Little*, *Star Wars* e *Senhor dos Anéis* são exemplos de filmes em que foram utilizados os *blendshapes* (DENG; NEUMANN, 2008).

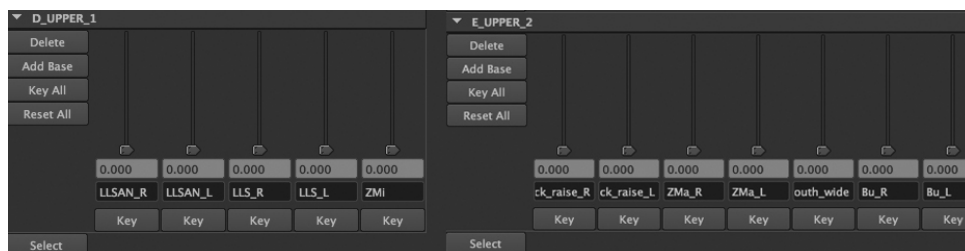
Figura 3.2 – Formas de *blendshape*.



Fonte: Deng e Neumann (2008).

Muitos aplicativos usados pela indústria de animação possuem a técnica de mudar as formas dos *blendshapes* implementada e seu controle é feito por meio do uso de painéis conhecidos como *Sliders*, que são usados para alterar as malhas (Figura 3.3).

Figura 3.3 – *Sliders* para alteração das malhas.



Fonte: Lewis et al. (2014).

3.2 Criação de expressões com *blendshapes*

Lewis et al. (2014) mostram que a criação de expressões utilizando *blendshapes* pode ser descrita como uma simples soma de vetores. Como exemplo, é considerado um conjunto de $n = 100$ expressões, cada uma com $p = 1000$ vértices, cada vértice possui 3 componentes x, y, z . Cada *blendshape* é transformado em um vetor b_k com 30.000 posições que correspondem a 1.000×3 tendo seus elementos dispostos na ordem de $xxxyyyzzz$ ou $xyzxyzxyz$ (Figura 3.4).

Figura 3.4 – Matriz-vetor que corresponde às expressões do modelos de *blendshapes*.

$$\begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{1z} \\ f_{2x} \\ f_{2y} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_{nz} \end{bmatrix} = \begin{bmatrix} x & x & \cdots & x \\ y & y & \cdots & y \\ z & z & \cdots & z \\ x & x & \cdots & x \\ y & y & \cdots & y \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

Fonte: Lewis et al. (2014).

O modelo de *blendshape* é expressado pela seguinte equação:

$$f = \sum_{k=0}^n W_k B_k \quad (3.1)$$

ou, em notação de matriz

$$f = B_w \quad (3.2)$$

onde f é a face resultante, B é uma matriz $m = 30.000 \times 100$ onde cada coluna do vetor b_k , corresponde a um *blendshape* individual e w é o peso aplicado em cada forma.

Uma face b_0 , tipicamente em expressão de repouso, é designada como face neutra e as faces restantes, b_k , $K = 1 \dots N$ são substituídas pela diferença entre $b_k - b_0$ entre a enésima K faces alvos e a face neutra:

$$f = b_0 + \sum_{k=1}^n W_k (b_k - b_0) \quad (3.3)$$

onde b_0 sendo a forma neutra. Denota-se isto como:

$$f = b_0 + B_w \quad (3.4)$$

Nesta formulação, os pesos w são limitados entre $[0, 1]$

3.3 Construção de *blendshapes*

Existem várias abordagens para a construção de *blendshapes*. Uma forma tradicional é a deformação de uma malha por um artista digital habilidoso. Outra alternativa é o escaneamento da face de um ator real, utilizando equipamentos como *kinect* da *Microsoft* ou a *RealSense*, que é uma câmera desenvolvida pela *Intel* que captura imagens em RGB e RGB-D. (Figure 3.5).

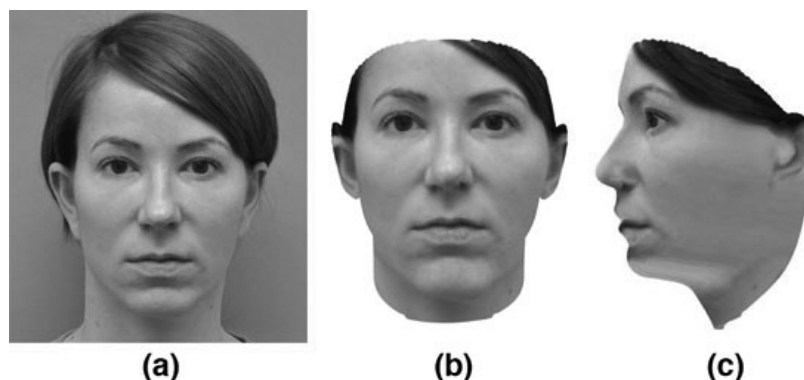
Figura 3.5 – Captura de uma face usando um dispositivo de captura 3D.



Fonte: elaborada pelo autor.

Blanz e Vetter (1999) desenvolveram uma técnica para modelar faces 3D automaticamente a partir de uma ou mais fotografias. Através de modelos previamente capturados, é realizado um processamento para gerar a face final adaptada para a geometria da face das fotos. Isto é feito usando uma base de dados de mais de 200 faces capturadas em alta resolução, como pode ser visto na figura 3.6.

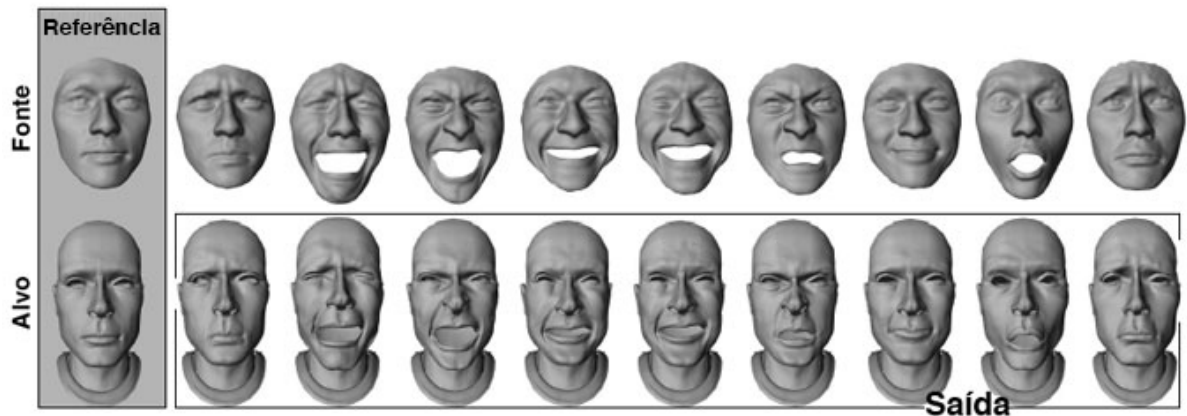
Figura 3.6 – Exemplo de modelo morfológico. (a) Foto de uma face 2D; (b) Modelo 3D gerado em posição frontal; (c) Modelo 3D gerado em posição lateral.



Fonte: elaborada pelo autor.

Sumner e Popović (2004) criaram uma técnica de modelagem de *blendshapes* que usa transferências de expressões entre um modelo padrão e um conjunto de *blendshapes* como mostrado na figura 3.7.

Figura 3.7 – Transferência de expressões entre modelos 3D.



Fonte: Sumner e Popović (2004).

3.4 Técnicas de animação e interatividade

Lewis et al. (2014) dividem as técnicas de animações utilizando *blendshapes* em:

- Animação por quadros chaves onde os modelos de *blendshapes* são animados utilizando-se controles deslizantes para alterar as formas. Este tipo de abordagem é encontrado em *softwares* comerciais como o *Maya* da *Wavefront* (RITCHIE; CALLERY; BIRI, 2005);
- Animação baseada em performance utilizada para que o movimento facial de um ator conduza a face de um ator virtual e é muito usado em efeitos visuais de filmes através de computação gráfica;
- Clonagem de expressões possibilitando que um modelo facial fonte seja redirecionado para um rosto alvo com proporções significativamente diferentes. A clonagem de expressão é muito usada em animações. Por exemplo, um ator adulto pode produzir o movimento para uma pessoa mais nova ou mais velha, como no filme o Curioso Caso de Benjamin Button ou uma criatura não humana como no filme Avatar e o Gollum, personagem do filme Senhor dos Anéis; e
- Animação parcialmente automatizada que permite um ajuste manual. Isto ocorre porque muitas das etapas do processo de animação baseada em performance não conseguem mostrar com fidelidade alguns detalhes da face. Outro motivo é que, em alguns casos, o movimento humano deverá ser transferido para uma face não humana, exigindo um ajuste para que isto aconteça.

3.5 Problemas no uso de *blendshapes* para a animação facial

Para Lewis et al. (2014), o uso de *blendshapes* em animações faciais é a maneira mais simples para alterar a forma de uma face, e limitações com este modelo linear são evidentes.

A abordagem de criar novas faces, usando *blendshapes* pode, dependendo do foco, trazer alguns inconvenientes. Quando se usa toda a face, aplicando o peso em todas as áreas ao mesmo tempo, é mais fácil reproduzir as expressões e pode ser eficiente para modelar a fala se a base inclui um conjunto de visemes¹. Por outro lado, essa abordagem torna mais difícil o controle dos ajustes locais como o ajuste mais acurado da boca ou dos lábios separadamente (LEWIS; ANJYO, 2010).

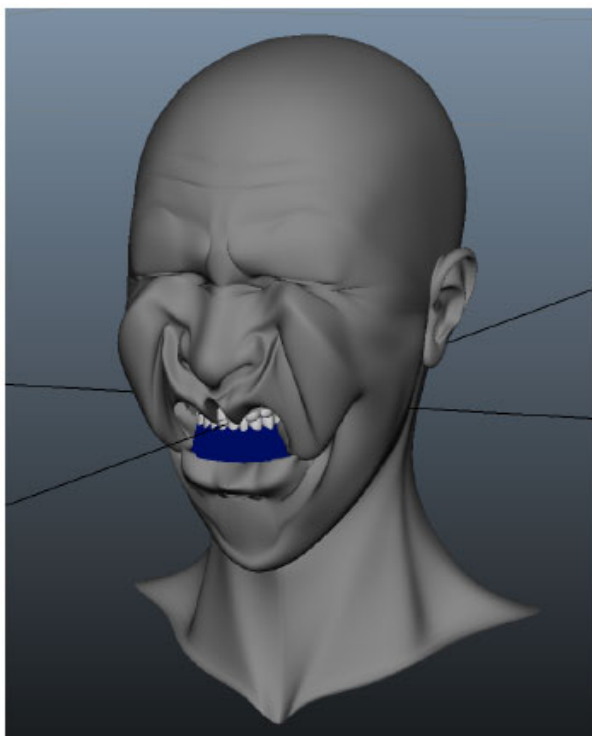
A linearidade dos modelos é importante, pois, para que sejam criadas faces de maneira correta, os modelos devem ter similaridade entre os vértices, evitando distorções. Caso isto não ocorra, os pesos não poderão ser aplicados.

A construção de uma base para os *blendshapes* deve acompanhar um significado semântico bem definido. Cada expressão segue um padrão identificando um tipo de área da face ou comportamento. Com isto, podemos separar áreas específicas dentro de um *dataset* de *blendshapes*. Isto permite direcionar o uso específico deste *dataset* para ações específicas reduzindo, assim, o uso de muitos *blendshapes* para a construção da animação.

Para uma base bem construída, a produção de expressões faciais utiliza poucas animações alvo. Quando muitos modelos alvo são utilizados de forma desnecessária, o resultado final falha, criando uma expressão totalmente errada como pode ser vista na Figura 3.8.

¹ Visemes são expressões que contém o movimento labial e simulam as letras do alfabeto

Figura 3.8 – Expressão criada com muitos modelos, inutilizando o resultado final



Fonte: Lewis et al. (2014).

Outro problema que pode ocorrer é a questão da geometria entre o modelo e a captura. Quando a captura e o modelo possuem a mesma geometria entre as partes da face, como a distância dos olhos, o tamanho da boca entre outras, a acurácia para o cálculo dos pesos é maior. Caso isto não ocorra, ajustes adicionais são necessários.

4 Algoritmos de rastreamento de faces

Para o processo de animação facial baseada em performance, são utilizados algoritmos para calcular os pesos que serão aplicados aos *blendshapes* para reproduzir a face do ator em um modelo digital. Este capítulo apresenta alguns algoritmos utilizados para esta finalidade e foram utilizados nesta pesquisa.

4.1 Distância Euclidiana

A geometria euclidiana ocupa-se do estudo das formas e das ligações algébricas conectadas a elas. A geometria euclidiana (plana) fundamenta-se na ideia intuitiva de ponto, sendo que, a partir dele, formam-se as ideias de retas e planos. As retas e os planos nada mais são que um conjunto de pontos, sem se limitar a um fim, ou seja, são infinitos em ambas as direções (ROBISON, 2014).

A distância euclidiana (DE) é a medida que calcula a distância entre dois pontos em um espaço vetorial o qual se pode deduzir a partir do teorema de Pitágoras. Aplicando esta distância, este espaço se torna um espaço métrico.

Se $u = (x_1, y_1)$ e $v = (x_2, y_2)$ são dois pontos em um plano, a distância euclidiana é dada por:

$$d_2(x, y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad (4.1)$$

Geometricamente, é a distância do segmento entre u e v .

A distância euclidiana entre os pontos $P = (p_1, p_2, \dots, p_n)$ e $Q = (q_1, q_2, \dots, q_n)$ de um espaço euclidiano n-dimensional é definido como:

$$(P, Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^N (p_i - q_i)^2}. \quad (4.2)$$

Para calcular a distância euclidiana entre dois vetores de n-dimensões x e y é definida como o escalar:

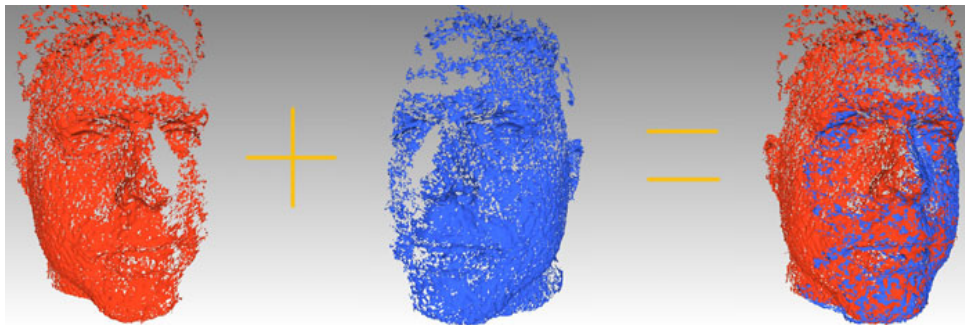
$$d_e(x, y) = \|x - y\| = \|y - x\| = [(x_1 - y_1)^2 + \dots + (x_n - y_n)^2]^{1/2} \quad (4.3)$$

A Equação 4.3 é a Norma da diferença entre os vetores.

4.2 Iterative Closest Point

Quando se faz uma captura de um objeto tridimensional através de um dispositivo de alta resolução, o resultado final possui uma ótima qualidade. Quando se utiliza equipamentos de baixo custo, como o *Kinect* ou *RealSense*, que não possuem alta resolução, é necessário implementar algumas técnicas de alinhamento de nuvens de pontos para melhorar o resultado final, pois na maioria dos casos, a captura apresenta algumas falhas como pode ser visto na Figura 4.1.

Figura 4.1 – Alinhamento de nuvens de pontos a fim de aumentar a qualidade da captura.



Fonte: Raposo e Batista (2012).

Para este alinhamento, é necessária uma transformação rígida para definir a relação entre as nuvens de pontos, estimando a pose da câmera para cada mapa de profundidade gerado pela captura. Este processo é chamado de registro e a transformação é rígida porque, possivelmente, o modelo não sofrerá deformações durante o processo de aquisição (MACEDO; SOUZA, 2012).

Um dos algoritmos mais utilizados para o alinhamento rígido é o ICP (*Iterative Closest Point*). A sua entrada consiste em nuvens de pontos onde a correspondência é definida com base na vizinhança dos pontos próximos utilizando posições geométricas entre os pontos ou as cores. O processo é iterativo e pode ser repetido várias vezes para refinar o alinhamento, removendo as discrepâncias entre os pontos e redefinindo as correspondências entre eles (BELLEKENS et al., 2014).

Existem dois algoritmos de ICP: o ICP ponto a ponto e o ICP ponto a superfície. Estas duas abordagens diferem na correspondência dos pontos e são descritas a seguir.

4.2.1 ICP ponto a ponto

Descrito inicialmente por Rusu (2010), este tipo de ICP obtém os pontos correspondentes através do vizinho mais próximo do ponto q_i de um ponto P_j na nuvem de pontos.

O vizinho mais próximo é definido através dos termos da métrica da distância euclidiana:

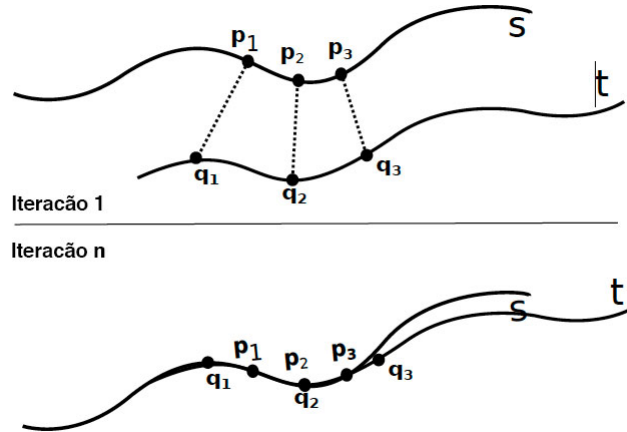
$$\hat{i} = \arg \min_i \|p_i - q_j\|^2, \quad (4.4)$$

onde $i \in [0, 1, \dots, N]$ e N representa o número de pontos na nuvem de pontos alvo. Os parâmetros da rotação R e a translação t são estimados, minimizando o quadrado das distâncias entre os pares correspondentes:

$$R, t = \arg \min_{R, t} \sum_{i=1}^N \|(Rp_i + t) - q_i\|^2 \quad (4.5)$$

É feita uma iteração entre as equações 4.4 e 4.5 para melhorar as estimativas das interações anteriores, como pode ser visto na Figura 4.2.

Figura 4.2 – ICP baseado numa abordagem ponto a ponto.



Fonte: Bellekens et al. (2014).

4.2.2 ICP ponto a superfície

Este algoritmo foi proposto por Low (2004). Ao invés de encontrar o vizinho mais próximo de um ponto p_j , leva em conta a vizinhança g_j para reduzir a sensibilidade do algoritmo a ruídos.

Ele assume que as nuvens de pontos são lineares e coplanares¹ sendo definido pelo seu vetor normal n que é obtido com o menor autovetor da matriz de covariância dos pontos que circulam o ponto correspondente q_i .

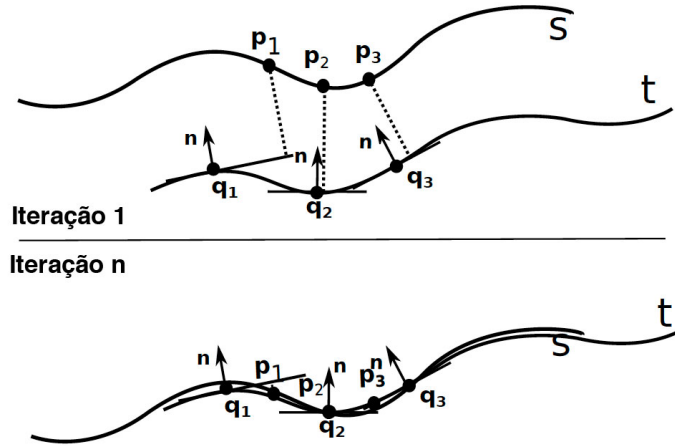
¹ Todos os pontos estão no mesmo plano geométrico.

Ao invés de minimizar diretamente a distância euclidiana entre os pontos correspondentes, pode-se minimizar a projeção escalar dessa distância sobre a superfície planar definida pelo vetor normal n onde:

$$\hat{R}, \hat{t} = \arg \min_{R, t} \left(\sum_{i=1}^N \|((Rp_i + t) - q_i)n_i\| \right) \quad (4.6)$$

O resultado pode ser visto na Figura 4.3

Figura 4.3 – ICP baseado numa abordagem ponto a superfície.



Fonte: Bellekens et al. (2014).

4.3 Distância de Hausdorff Modificada

Hossain et al. (2012) demonstra que a distância de Hausdorff é a máxima distância de um conjunto ao ponto 2D mais próximo de outro conjunto. Ela é uma métrica muito usada em vários tipos de aplicações de visão computacional como a correspondência de imagens, a detecção de objetos móveis, o rastreamento e o reconhecimento, a recuperação de formas e a análise de imagens baseada em conteúdo.

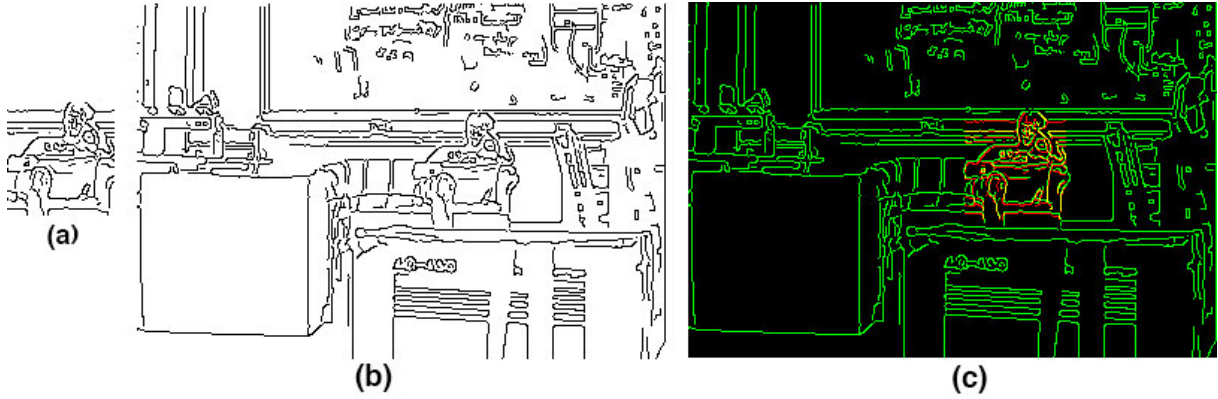
Segundo Braun (2014) a distância de Hausdorff, h , do conjunto A ao conjunto B é uma função de máximo/mínimo definida por:

$$h(A, B) = \max_{a \in A} \left\{ \min_{b \in B} \{d(a, b)\} \right\} \quad (4.7)$$

onde $d(a, b)$ é a distância Euclidiana entre $a \in A$ e $b \in B$. A Equação 4.8 encontra a mínima distância de cada ponto de A a todos os pontos de B , selecionando o maior entre esses valores.

Dubuisson e Jain (1994) introduziram uma mudança no cálculo e criaram a Distância de Hausdorff Modificada (DHM) que, através dos resultados apresentados, se mostrou

Figura 4.4 – Exemplo de Distância de Hausdorff Modificada. (a) Imagem a ser encontrada; (b) Cena completa; (c) Imagem encontrada destacada em vermelho encontrada na cena.



Fonte: Cornell University (1994).

melhor para a comparação de objetos em imagens do que a original. Ela é a média entre os mínimos valores de distância dos pontos do conjunto A aos pontos do conjunto B e é dada pela seguinte Equação:

$$H(A, B) = \frac{1}{N_A} \sum (\min_{b \in B} \{d(a, b)\}) \quad (4.8)$$

onde N_A é o número de elementos de A .

4.4 Principal Component Analysis

Shlens (2005) cita que a PCA nasceu no campo da estatística e foi popularizada na década de 60 sendo uma técnica de análise estatística usada para compressão, visualização e classificação de dados. Ela reduz a dimensionalidade de um conjunto de dados com o mínimo de perda de informação. Os conjuntos de dados têm a propriedade de que os pontos correspondentes num espaço de n dimensões ficam concentrados em um espaço de dimensionalidade muito menor. Encontrar esse espaço pode reduzir, nesses casos, a complexidade dos dados e tornar mais fácil a extração de conhecimento a partir deles (BISHOP, 2006).

Inicialmente foi proposta em 1901 por Karl Pearson que explorou a idéia de adaptar um espaço a uma série de pontos, onde o ajuste era medido pela soma das distâncias ortogonais ao quadrado de cada ponto. Em 1933, Harold Hotelling usando os mesmos princípios de Pearson começou a desenvolver a idéia que hoje é chamada de redução, tomando um conjunto de variáveis e encontrando um conjunto menor independente dos valores da variáveis originais. Por este motivo, ela também é conhecida como transformação

de *Hotelling*. Em outros campos de pesquisa ela é também conhecida como a transformada de *Karhunen-Loève*(KLT)(RERIS; BROOKS, 2015).

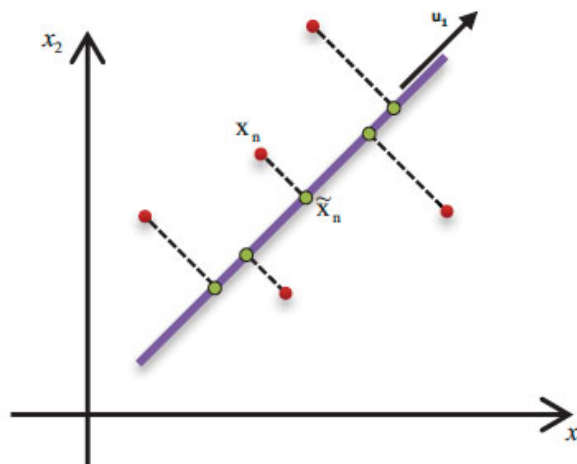
Braun (2014) cita que a PCA pode ser definida como a projeção ortogonal dos dados em um espaço linear de dimensionalidade menor, conhecido como subespaço principal, tal que a variância dos dados projetados é maximizada. Cada i -ésimo vetor de base desse novo subespaço é chamado Componente Principal (PC) e denotado por u_i . Os PCs são vetores unitários e ortogonais entre si, constituindo assim uma base do subespaço principal. Dessa forma, cada vetor de dados de entrada x_n tem uma representação no subespaço principal \tilde{x}_n por uma combinação linear da seguinte maneira:

$$\tilde{x}_n = \sum_{k=1}^d a_k u_k \quad (4.9)$$

onde d é a dimensionalidade do subespaço principal. Se d for igual ao número de variáveis de entrada D , \tilde{x}_n é aproximadamente equivalente a x_n , porém num novo sistema de coordenadas.

Considere os pontos vermelhos do gráfico da Figura 4.5 como dados x_n referentes a duas variáveis x_1 e x_2 . O subespaço principal de dimensão $d = 1$ é, nesse caso, representado pela linha roxa e o seu PC é denotado por u_1 . Esse subespaço é definido de forma a maximizar a variância dos pontos nele projetados (em verde).

Figura 4.5 – Projeção de dados de entrada x_n no componente principal u_1 .



Fonte: Braun (2014).

Seja um conjunto de dados x_n , onde $n = 1, 2, \dots, N$ e x_n é uma variável euclidiana com dimensão D projetada em um espaço com dimensionalidade $d < D$ com a variância maximizada. Em um primeiro momento, seja $d = 1$. Define-se a direção deste espaço

usando um vetor unitário u_1 . Cada ponto x_n é então projetado em um valor escalar $u_1^T x_n$. A variância dos dados projetados é dada por:

$$\frac{1}{N} \sum_{n=1}^N (u_1^T x_n - u_1^T \tilde{x})^2 = u_1^T S u_1 \quad (4.10)$$

onde S é a matriz de covariância dos dados definida por:

$$S = \frac{1}{N} \sum_{n=1}^N (x_n - \tilde{x})(x_n - \tilde{x})^T \quad (4.11)$$

com \tilde{x} sendo a média amostral dos dados.

Maximizando a variância projetada por $u_1^T S u_1$ (equação 4.10) com respeito a u_1 otimizando para prevenir que $\|u_1\| \rightarrow \infty$. A restrição apropriada consiste na condição de normalização $u_1^T u_1 = 1$. Para reforçar essa restrição, pode-se introduzir um multiplicador de Lagrange (Oliveira O, 2012) (denotado por λ_1), e realizar a maximização irrestrita de:

$$u_1^T S u_1 + \lambda_1(1 - u_1^T u_1) \quad (4.12)$$

Para encontrar a maximização da variância, pode-se igualar a derivada da função acima com respeito a u_1 a zero. Com isso, pode-se observar que um ponto estacionário é obtido quando:

$$S u_1 = \lambda_1 u_1 \quad (4.13)$$

o que indica que u_1 deve ser o autovetor de S . Ao se multiplicar a equação 4.13 por u_1^T e fazer o uso da restrição $u_1^T u_1 = 1$, pode-se observar que a variância é dada por:

$$u_1^T S u_1 = \lambda_1 \quad (4.14)$$

Assim, a variância será máxima quando u_1 for igual ao autovetor que tiver o maior autovalor. Este será o primeiro componente principal. Os próximos componentes principais serão os que terão o maior autovalor.

Além da implementação padrão da PCA existem outras como a WPCA (*Weighted Principal Component Analysis*) apresentada por Seol et al. (2011) onde é usado primeiro a distância entre a amostra de teste e cada amostra de treinamento para calcular a matriz de covariância ponderada. Em seguida é explorada a matriz de covariância obtida para executar a extração de características obtendo uma precisão maior do que a PCA convencional (FAN; LIU; XU, 2011).

5 Animação facial baseada em performance

Neste capítulo, são apresentados os conceitos de animação facial baseada em performance, o redirecionamento facial, seus domínios, seus desafios e alguns trabalhos na área que possuem no seu *pipeline* a utilização de *blendshapes*.

5.1 Conceito

A animação facial baseada em performance também conhecida como *retargeting* tem sido um assunto muito pesquisado nos últimos anos. Ela introduz a ideia da captura da face de um ator real e seu redirecionamento para um ator virtual. Essa transferência pode ser aplicada a uma animação 3D criada por um artista de forma manual (DUTREVE; MEYER; BOUAKAZ, 2008).

Para Li et al. (2013), animação facial baseada em performance refere-se ao problema de mapeamento de expressões faciais de um ator para um avatar digital de maneira realista e compatível com o desempenho de entrada. É normalmente constituída por uma etapa de rastreamento da face seguida por um procedimento de síntese de expressão.

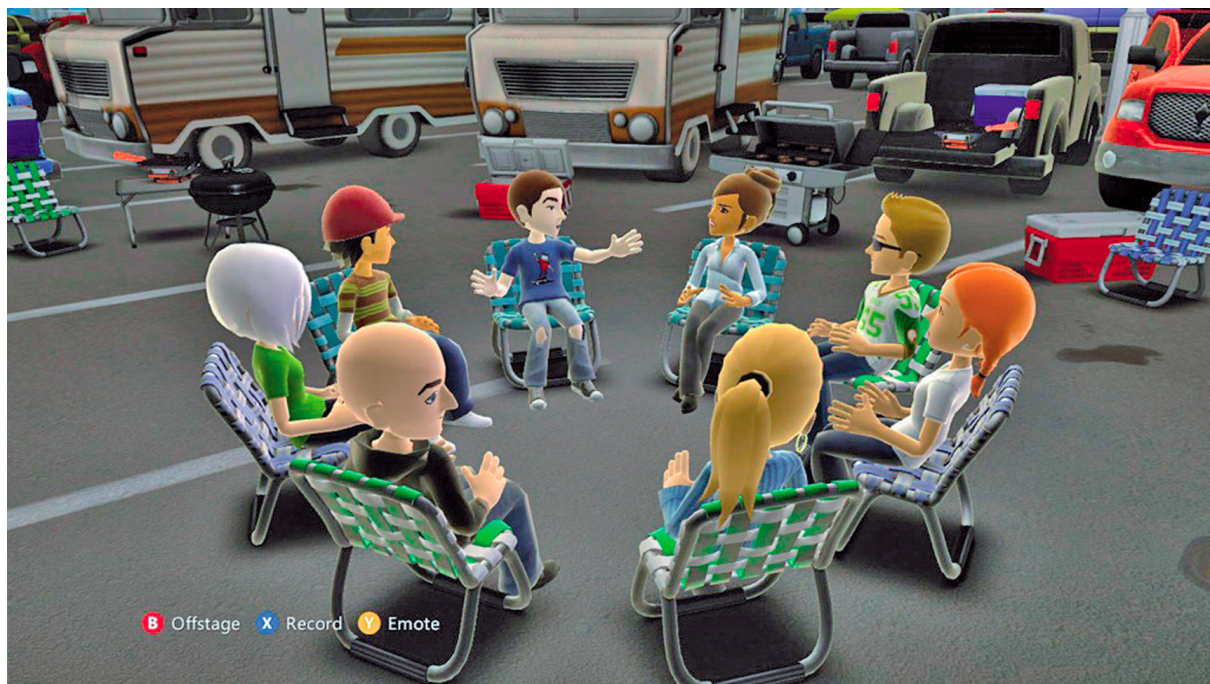
5.2 Domínios da animação facial baseada em performance

De acordo com Dutreve, Meyer e Bouakaz (2008), a animação facial baseada em performance é encontrada em aplicações como jogos 3D, interação homem/máquina e na indústria do cinema. Filmes como *King Kong* e *Avatar* são exemplos do uso de animação facial baseada em performance (SEOL et al., 2012). No mundo dos *games*, esta tecnologia é usada para controlar avatares, representando a face do usuário. Isso permite que os seus amigos reais percebam a expressão do usuário transmitida para o mundo virtual em tempo real como pode ser visto na Figura 5.1.

Um uso sugerido para essa tecnologia é em exposições interativas. Com o redirecionamento em tempo real, pode-se construir instalações artísticas onde quadros estáticos poderão interagir com uma pessoa que esteja posicionada à sua frente, imitando as suas expressões faciais em tempo real. A Figura 5.2 mostra um observador passando na frente do quadro e o personagem retratado na pintura virtual começa a imitar a expressão facial da pessoa em tempo real. Um sensor para captura da face deverá estar incorporado à estrutura.

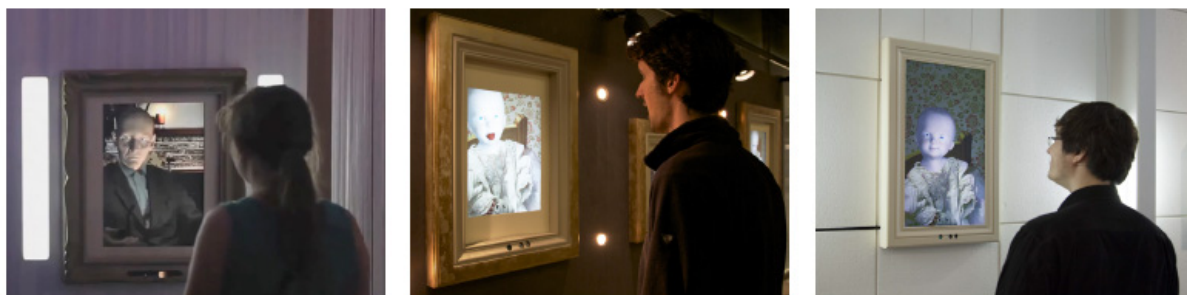
Aplicações médicas também podem se beneficiar dessa tecnologia, na qual novas formas de terapia interativa são possíveis. Por exemplo: sessões de treinamento baseadas

Figura 5.1 – Os usuários podem modificar as expressões dos seus avatares através da expressão facial capturadas em tempo real.



Fonte: Zhang (2012).

Figura 5.2 – Exemplo de exposições interativas.



Fonte: Pauly (2013).

em avatares podem ser criadas para pessoas com autismo ou outros transtornos do desenvolvimento neural (PAULY, 2013).

5.3 Trabalhos relacionados

A seguir, são apresentados alguns trabalhos que desenvolveram métodos que compartilham os princípios fundamentais da animação facial baseada em performance. Na maioria dos casos, utilizam informação RGB-D que é a informação de profundidade da imagem que se caracteriza por uma nuvem de pontos que contém informações em 3D, marcadores faciais 2D e câmeras de baixo custo como o *Kinect* da *Microsoft*. Além disso,

alguns trabalhos utilizam *blendshapes*.

5.3.1 Método de Li et al. (2013)

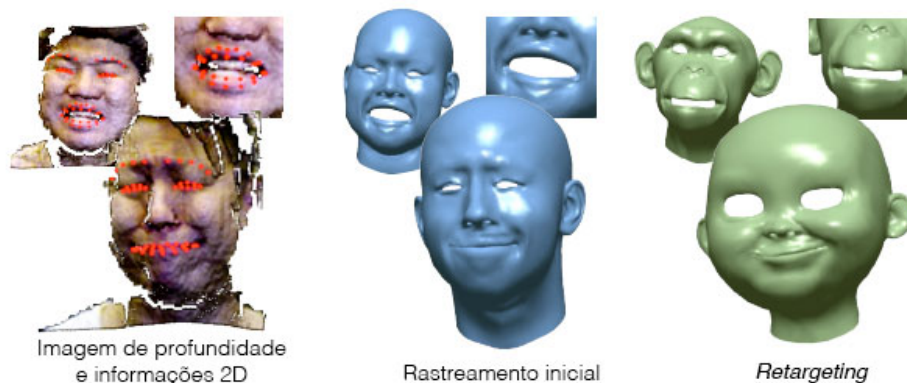
O método apresentado por Li et al. (2013), captura a face através do *Kinect* que é um dispositivo desenvolvido pela *Microsoft* que, além de capturar as imagens em RGB, também captura as imagens de profundidade. São utilizadas imagens RGB-D e vídeos 2D em tempo real. Ele possui, no seu *pipeline*, uma etapa de pré-processamento feito de forma *offline* para capturar dados iniciais e o processo *online* feito em tempo real. Em ambos os processos, alguns algoritmos e métodos para ajuste da malha e redirecionamento são utilizados.

Um deles é o método ICP (*Iterative closest point*). Este método, de acordo com Rusinkiewicz e Levoy (2001), tornou-se dominante no alinhamento tridimensional de modelos baseados em geometria e malhas. Ele é usado nesse trabalho para ajustar a captura da malha nas fases iniciais do método, corrigindo as suas imperfeições.

O PCA (*Principal Component Analysis*), outro método utilizado nesse trabalho é amplamente utilizado para a redução de dimensionalidade das informações, compressão de dados, extração de características relevantes e visualização de informação. Ele é usado no processamento das informações para o redirecionamento (BRAUN, 2014), .

O método é iniciado por uma captura da face na posição neutra. São usadas as informações RGB-D e 40 marcadores faciais, capturados automaticamente, que correspondem aos pontos ao redor da boca e dos olhos que, segundo os autores, são fundamentais para o resultado final do método como pode ser visto na Figura 5.3.

Figura 5.3 – Visão geral do método. À esquerda, é feita a captura das informações em RGB-D e 2D; Ao centro, as informações são processadas gerando suas expressões correspondentes; À direita, o redirecionamento.

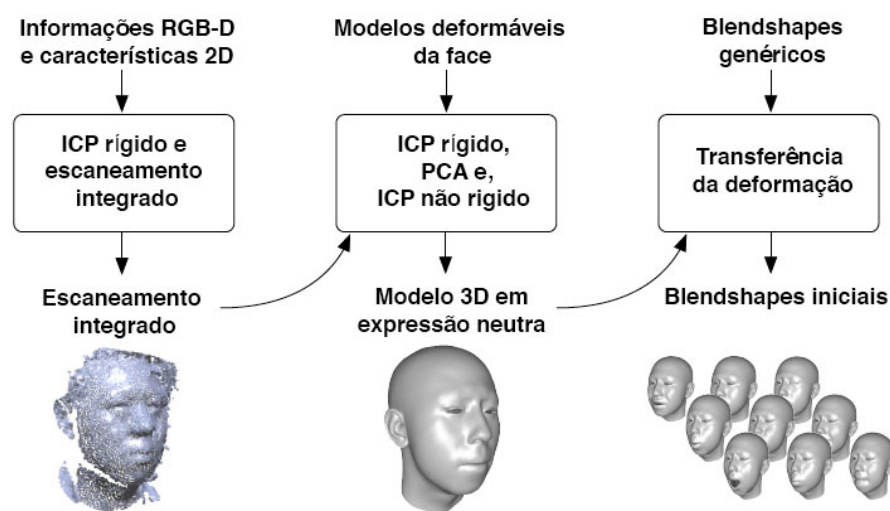


Fonte: Li et al. (2013).

5.3.1.1 Visão geral do método

A etapa inicial proposta no método é *offline* e pode ser vista na Figura 5.4. É construído um modelo 3D através de uma captura da face em expressão neutra usando o *Kinect*. É usado o método ICP para agrupar múltiplos quadros de informações RGB-D e melhorar a captura de entrada. É gerada, automaticamente, uma base de dados com 23 expressões baseadas nos *FACS* que serão os *blendshapes* iniciais usados no redirecionamento através do método de Sumner e Popović (2004).

Figura 5.4 – Construção dos *blendshapes* iniciais baseados em 23 *FACS*.



Fonte: Li et al. (2013).

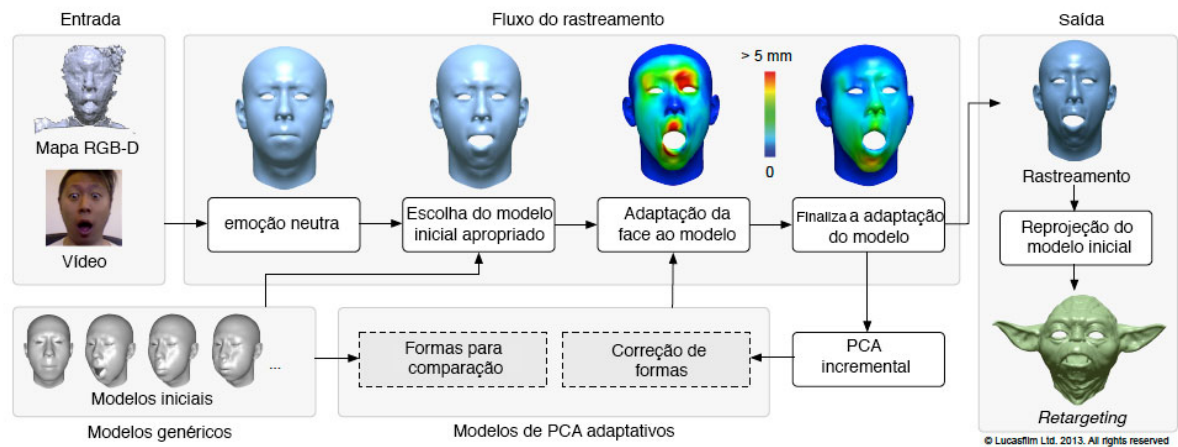
No processo *online*, é iniciada a captura do ator através do *Kinect* que traz a imagem de profundidade e o vídeo 2D. Nesta etapa, é usado o ICP pra ajustar a malha 3D. Após o ajuste, o *pipeline*, através do cálculo dos pesos, indica qual é o modelo inicial de *blendshape* do *dataset* que deve ser utilizado. Na etapa seguinte, é feita uma adaptação ao modelo, usando um ICP não rígido. O ajuste final usa uma técnica de PCA incremental refinando os coeficientes usados para o redirecionamento (Figura 5.5).

5.3.2 Método de Li et al. (2013)

Li et al. (2013) apresentam um método de redirecionamento em tempo real, usando o *Kinect* e o *software* de modelagem 3DS Max.

O redirecionamento se dá entre a captura em tempo real de uma face para um avatar virtual implementado no *software* 3DS Max. A transferência das informações entre a captura de entrada e o avatar se dá pelo protocolo *MIDI* (*Musical instrument device interface*), que é um protocolo padrão da indústria da música (YUAN; LU; HE, 2010).

Figura 5.5 – Visão geral do processo.



Fonte: Li et al. (2013).

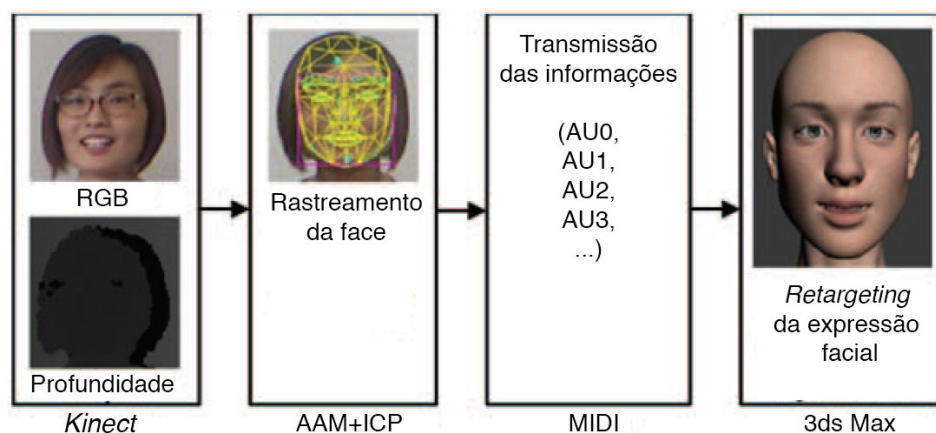
Neste método, é utilizado o AAM (*Active Apperance Model*), responsável pelo rastreamento da face e extração dos pontos faciais. (ZHOU et al., 2010).

5.3.2.1 Visão geral do método

O método é composto por duas etapas. A primeira consiste na construção de um avatar digital, utilizando o aplicativo 3DS Max. Este avatar será usado no processo do redirecionamento.

Na segunda etapa, o processo acontece em tempo real e foi dividido em quatro estágios: captura da face, rastreamento da face, transmissão dos dados e redirecionamento. Uma visão geral do método da segunda etapa pode ser visto na Figura 5.6.

Figura 5.6 – Visão geral do processo.



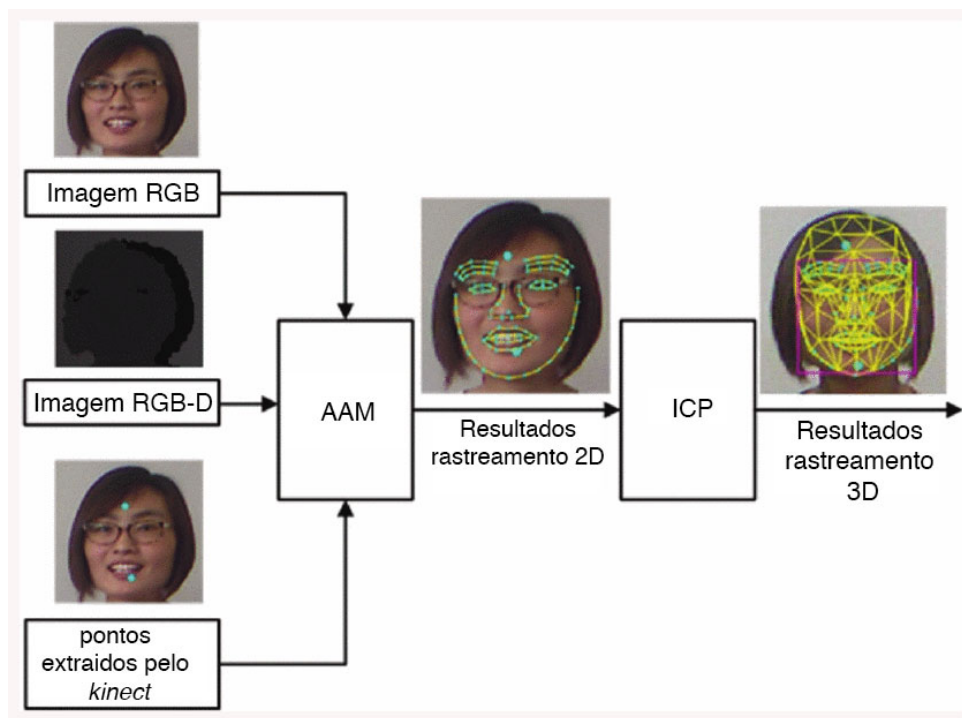
Fonte: Li et al. (2013).

No primeiro estágio da segunda etapa, a captura da face é feita usando o *Kinect*.

No segundo estágio da segunda etapa, o método AAM (*Active Apperance Model*) é usado na face capturada para extrair 100 pontos faciais que serão utilizados no processo para o redirecionamento.

Após o rastreamento dos pontos faciais, é criado sobre a face um modelo chamado Candide-3. Este modelo está disponível publicamente e é compatível com a base de dados MPEG-4. O Candide-3 é usado como base para a extração dos AUs da face capturada, pois já possui os padrões de AUs identificados de acordo com o arranjo de seus vértices (AHLBERG, 2001). A extração dos AUs se dá através do método ICP para minimizar as distâncias dos pontos do Candide-3 com os pontos da imagem de profundidade capturada da face. Um exemplo desse processo pode ser visto na Figura 5.7.

Figura 5.7 – Visão geral do processo de rastreamento da face.



Fonte: Li et al. (2013).

No último estágio da segunda etapa, os parâmetros de controle obtidos na etapa de processamento são enviados para o avatar pré-configurado, usando o protocolo MIDI.

5.3.3 Método de Behrens et al. (2013)

O método apresentado por Behrens et al. (2013) propõe um sistema automático para controle de expressões, em um avatar digital, em tempo real. A captura da face se dá através do *Kinect* e utiliza imagens RGB-D e vídeos 2D. Possui uma etapa de pré-processamento feito de forma *offline* e uma etapa *online*.

O método possui, tanto na etapa *offline* como na *online*, um processo de normalização da face que possui a detecção da geometria da face, remoção das imagens de fundo, alinhamento frontal da face através do uso do método de ICP e correções de iluminação.

Por meio do uso de um *software* de modelagem 3D, é gerado um avatar virtual base. Este avatar usa *blendshapes* que são alterados para formar novos modelos customizados para cada usuário.

Na etapa de pré-processamento, é gerada uma base de dados customizada para cada usuário com as expressões universais.

É usado o algoritmo de Lucas-Kanade (BAKER; MATTHEWS, 2004) para calcular a distância entre os pontos faciais obtidos na captura *online* com os armazenados no processo *offline*.

5.3.3.1 Visão geral do método

Na etapa de pré-processamento, a face de um ator é capturada a partir de um *Kinect* em expressão neutra. Após a captura, ela é normalizada, onde é feita a detecção do rosto, a remoção do fundo, o alinhamento frontal e a correção da iluminação, se necessário.

São detectados, de maneira automática, 11 marcadores faciais ao redor dos olhos, nariz, cantos da boca, bochechas, ossos da mandíbula e queixo. Estes pontos representam os limites exteriores do rosto do usuário, bem como as posições e tamanho das características faciais.

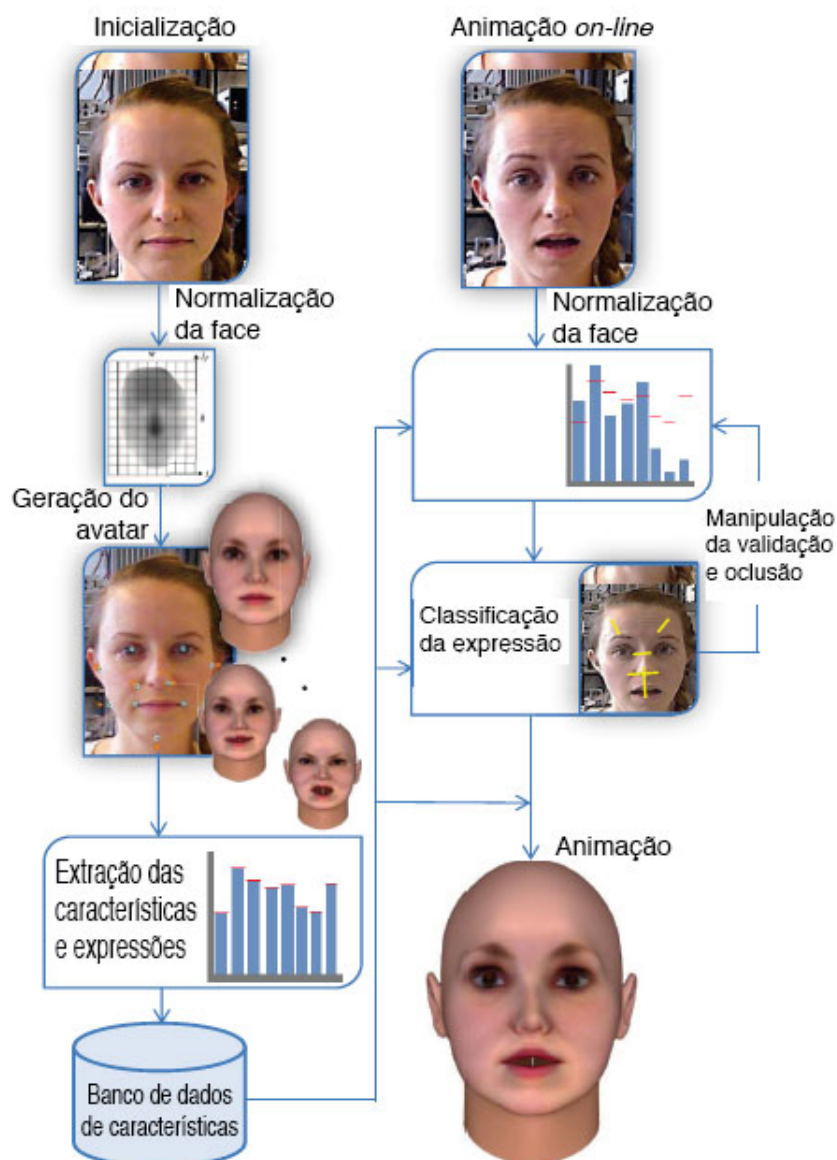
Após a detecção do conjunto de marcadores faciais, é gerada uma base de dados, contendo a posição dos mesmos nas expressões universais, usando o método proposto por Chai, Xiao e Hodgins (2003). A base de dados é usada em tempo real para a detecção das expressões.

Na captura em tempo real, a face do usuário é detectada, é realizado um processo de normalização e os marcadores faciais são extraídos. A posição dos marcadores faciais extraídos em tempo real é comparada com a posição dos marcadores faciais armazenados na base de dados, usando o algoritmo de Lucas-Kanade (BAKER; MATTHEWS, 2004). A posição mais próxima indica qual é a expressão universal que deve ser enviada para o avatar digital. A visão geral do método pode ser vista na Figura 5.8.

5.3.4 Método de Weise et al. (2011)

Weise et al. (2011) criaram um método de rastreamento de faces em tempo real que utiliza o *Kinect*. Ele combina o mapeamento 3D da nuvem de pontos com os marcadores 2D para gerar, a partir de uma sequência de animações, uma nova face através do uso de *blendshapes*.

Figura 5.8 – Visão geral do método.



Fonte: Behrens et al. (2013).

O objetivo deste método, segundo os autores, é ser flexível e utilizar a facilidade de captura de câmeras de baixo custo com animações existentes criadas por capturas mais complexas, mantendo a qualidade final do processo. Ele possui um processo *offline* onde é criado um *dataset* de *blendshapes* com a face do ator que será capturado e uma fase *online* para o processamento das expressões.

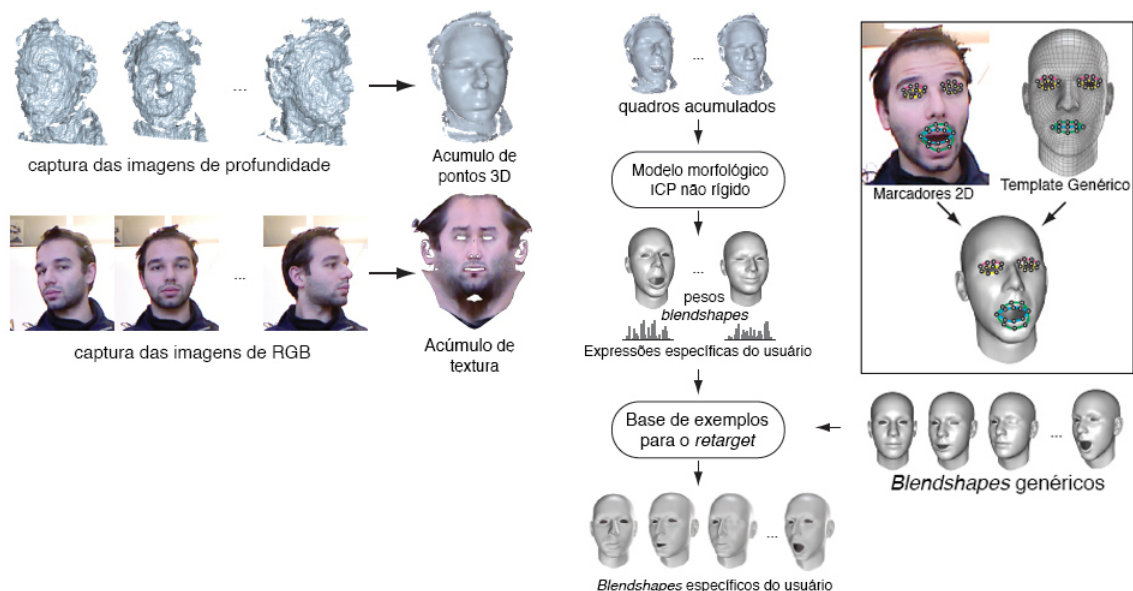
5.3.4.1 Visão geral do método

Na fase *offline*, é feita uma adaptação de um modelo de *blendshape* genérico com algumas expressões do usuário. Estas expressões são capturadas pelo *Kinect* e reconstruídas, usando um modelo morfológico combinado a um método de alinhamento não rígido. Para

customizar o *blendshape* genérico, é gravada uma sequência pré-definida de expressões do ator. Nesta gravação, é capturado mais de um quadro, pois o *Kinect* gera muito ruído na captura da nuvem de pontos e estes quadros são agrupados para criar uma nuvem de pontos mais densa. O autor usa o algoritmo de Viola e Jones (2001) para detectar a face no primeiro quadro e para o ajuste das cores capturadas usa a reconstrução de Poisson (PÉREZ; GANGNET; BLAKE, 2003). Para a reconstrução das expressões, o autor utiliza modelo morfológico de Blanz e Vetter (1999) que é um modelo de PCA linear para obter uma alta qualidade para a face do usuário. Este modelo utiliza cerca de 200 faces para a criação de um modelo customizado. A expressão capturada do usuário é aplicada neste *template* de alta resolução, usando uma abordagem de registro não rígido. Para melhorar a acurácia são incorporados marcadores manualmente na região da boca e dos olhos permitindo um alinhamento melhor.

Para a geração do *dataset* completo do ator, é usado o método proposto por Li, Weise e Pauly (2010). Este método usa os *blendshapes* criados na fase inicial e aplica pesos correspondentes para a geração de outras expressões. São gerados, ao todo, 39 *blendshapes* no processo *offline* inicial. Uma visão do processo é apresentada pela Figura 5.9.

Figura 5.9 – Fase *offline* do método.



Fonte: Weise et al. (2011).

O *pipeline* da fase *online* do método pode ser visto na figura 5.10. Ele inicia com uma captura da face do ator em tempo real para a leitura das informações 2D e 3D, seguindo por uma etapa para determinar os pesos aplicados aos *blendshapes* e, no final, a transferência dos pesos obtidos para a animação. Para a determinação dos pesos de forma mais realista, o autor explora uma coerência temporal entre quadros capturados

consecutivos, produzindo um resultado mais elaborado tanto para a geometria quanto para o movimento do usuário rastreado. Para determinação dos pesos é utilizado a MPPCA (*Probabilistic Principal Component Analyzers*) (TIPPING; BISHOP, 1999).

Figura 5.10 – Fase *online* do método.



Fonte: Weise et al. (2011).

Na fase de captura *online*, é feito um alinhamento rígido entre o quadro anterior e o quadro atual, utilizando um ICP rígido. Depois da aquisição da face de forma rígida, inicia-se o processo para estimar os pesos que devem ser aplicados aos *blendshapes*.

5.4 Desafios da animação facial baseada em performance

Uma animação facial requer muito tempo e habilidade do artista, pois a complexidade da face humana provê expressões sutis tornando a tarefa do redirecionamento automático complexa (DUTREVE; MEYER; BOUAKAZ, 2008).

Para Pauly (2013), estabelecer um mapeamento entre a captura do ator e a face virtual é difícil, porque as expressões podem não ser compatíveis entre elas quando a geometria ou o espaço de movimento são diferentes. Ele cita ainda que a criação de faces virtuais atraentes é atualmente um processo altamente complexo, que requer mão de obra especializada e significativo trabalho manual.

Outro problema apontado por Pauly (2013) é que expressões faciais sutis e micro-expressões são difíceis de capturar de maneira confiável. Adicionar mais marcadores faciais não é uma solução viável, devido a um aumento da sensibilidade à captura.

Efeitos secundários, na maioria dos casos, não são considerados, como rastreamento de cabelo, dentes e língua. Penteados são ainda muito desafiadores para modelar e simular. Não há soluções robustas atualmente para acompanhar, de forma realista, cabelos longos em tempo real (PAULY, 2013).

6 Sistema de animação facial baseado em performance

A criação de expressões faciais animadas requer a contínua geração de transições realistas entre diferentes expressões faciais. Em geral, a transformação entre malhas poligonais diferentes é uma tarefa não trivial, pois exige que um conjunto de correspondências entre malhas com topologias normalmente diferentes possa produzir um conjunto razoável de formas intermediárias.

Neste sentido, uma grande parte dos trabalhos encontrados na literatura se baseia em técnicas poderosas de animação, as quais deformam a malha de um objeto de maneira a transformá-lo em outro. Em ambientes profissionais, a animação facial é realizada por meio destas técnicas de deformações geométricas, *blendshapes* ou uma combinação dessas. As deformações geométricas são dirigidas pelas ações simuladas dos músculos, as quais são fracamente baseadas na dinâmica do tecido facial. Os *blendshapes*, por sua vez, interpolam um grande número de formas esculpidas.

Os *blendshapes*, como visto no Capítulo 3, são posturas faciais chaves definidas em um *dataset*, as quais podem ser usadas para definir um espaço linear para as expressões faciais. Como o nome sugere, uma nova postura pode ser gerada como uma mistura (*blend*) de duas ou mais posturas existentes. O controle do espaço linear pode ser especificado por meio do uso de funções generalizadas ou determinado automaticamente a partir de sessões de captura de movimentos. O *blendshaping* é uma das técnicas mais amplamente utilizadas na animação facial e é controlada por pesos – valores em um vetor base – que afetam a geometria da malha poligonal da face.

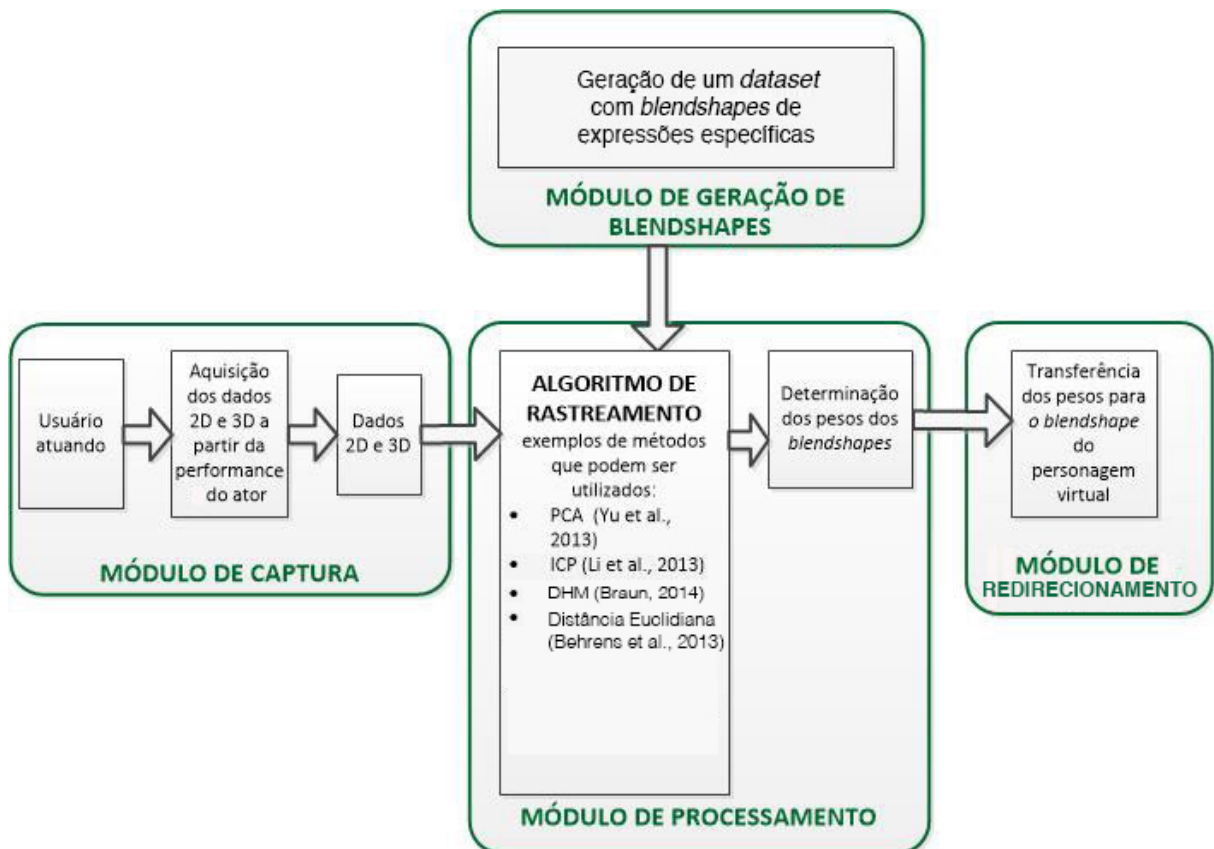
Muitos sistemas de animação, baseados na performance de um ator, possuem, ainda, a facilidade do redirecionamento (*retargeting*), algumas vezes referenciada como mapeamento cruzado (*cross-mapping*). O redirecionamento permite que a performance (em termos de suas expressões faciais) de um ator possa ser mapeada em um modelo de face diferente.

De acordo com Weise et al. (2011), a animação facial baseada em performance requer resolver dois desafios técnicos: o primeiro consiste em rastrear, com acurácia, os movimentos rígidos e não rígidos da face do usuário; o segundo é mapear os parâmetros de rastreamento para os controles adequados que dirigirão a animação da face do personagem virtual. Uma abordagem, encontrada em diversos métodos, é a combinação desses dois problemas em uma única otimização que encontre os valores mais adequados dos parâmetros mais prováveis de um modelo de expressão específico com base nos dados 2D e 3D observados.

Para definir o espaço de expressões faciais realistas, é comum derivar uma probabilidade inicial para essa otimização a partir de sequências de animação pré-gravadas.

Para a criação do sistema, adotou-se, então, a arquitetura modular ilustrada na Figura 6.1 semelhante aos trabalhos encontrados na literatura.

Figura 6.1 – Arquitetura e *pipeline* do sistema desenvolvido.



Fonte: elaborada pelo autor.

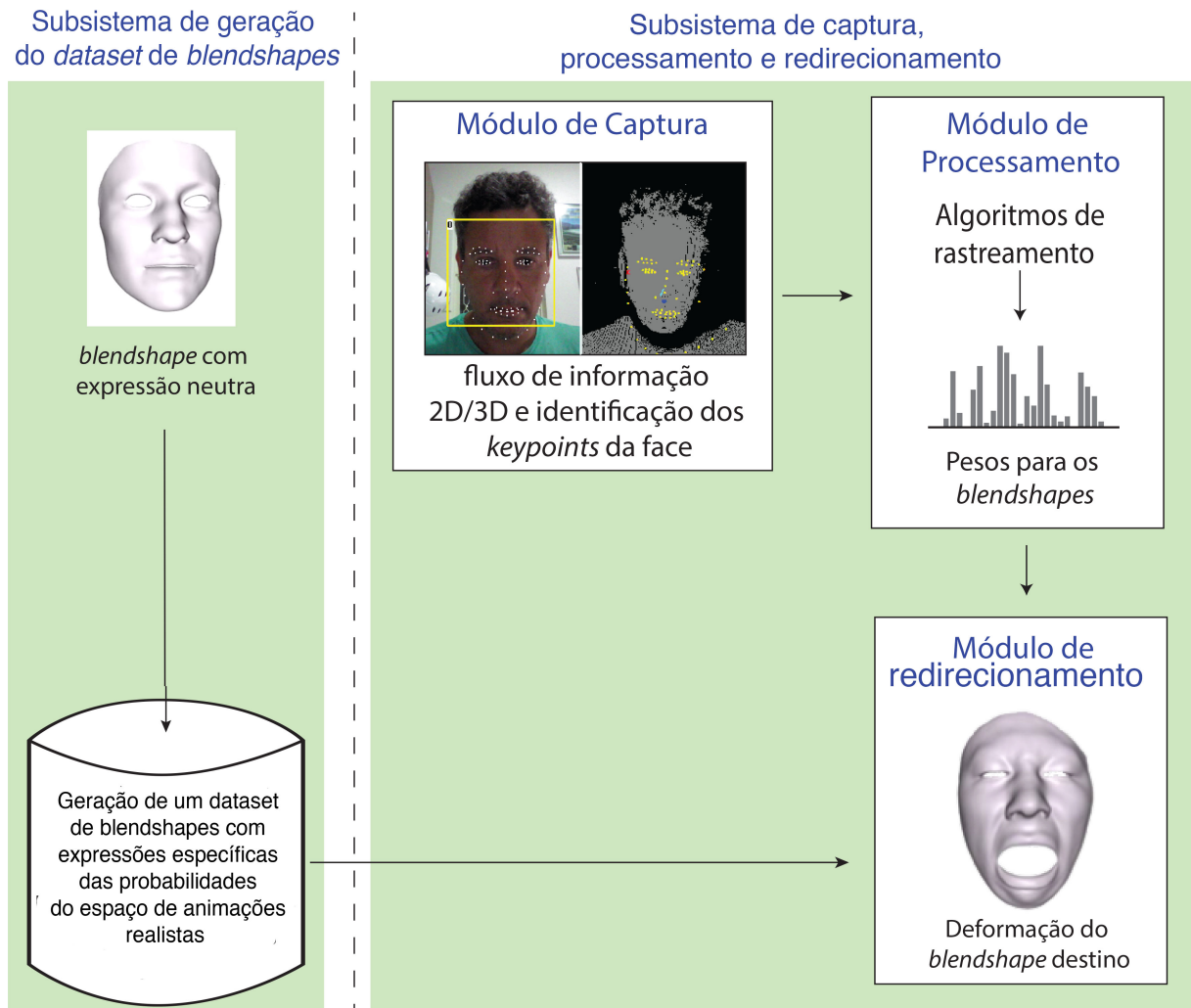
Pode-se perceber, na Figura 6.1, a diversidade das técnicas empregadas para a criação dos coeficientes de transferência. Em alguns casos, são empregadas mais de duas técnicas.

Portanto, foi criado neste projeto um sistema modular que estabelece um *pipeline* para animação facial baseada em performance, utilizando as informações RGB-D no qual se pode utilizar diferentes algoritmos na etapa de processamento. O *pipeline* desenvolvido pode ser visto na Figura 6.2.

O sistema é dividido em dois subsistemas: geração do *dataset* de *blendshapes* e captura, processamento e redirecionamento. O primeiro subsistema gera um *dataset* de *blendshapes* com expressões específicas para a criação de animações realistas.

O segundo subsistema é composto de um Módulo de Captura do ator onde serão extraídas as informações 2D e 3D que serão processadas no Módulo de Processamento e,

Figura 6.2 – Os subsistemas da arquitetura desenvolvida.



Fonte: elaborada pelo autor.

por fim, pelo Módulo de redirecionamento, onde será feita a transferência dos pesos da expressão facial para o *blendshape*, que representa o modelo da face do personagem virtual.

Cabe ressaltar que, no desenvolvimento do sistema, adotaram-se as seguintes restrições:

- A) a transferência de expressões faciais foi direcionada a um *blendshape* representando um modelo de face humana diferente da face de entrada, mas sem aplicação de textura; e
- B) não foram consideradas na criação dos *blendshapes* e na etapa de redirecionamento dos movimentos faciais, as informações relativas ao cabelo, dentes, olhos e língua.

7 Implementação do protótipo do sistema

Neste capítulo, a implementação do protótipo para o sistema de animação, baseado em performance, é descrita. Cada etapa é apresentada em função de suas características, os algoritmos utilizados e as soluções adotadas.

7.1 Subsistema de geração de *dataset* de *blendshapes*

Este subsistema configura um *dataset* de *blendshapes* com vários tipos de expressões baseadas em FACS que serão utilizadas no redirecionamento. A Figura 7.1 mostra as etapas necessárias para a geração desde *dataset*. Como base de dados, foram utilizados os *blendshapes* gerados pelo aplicativo *FaceGen Modeller 3.3*.

7.1.1 Geração dos *blendshapes* das faces

Foi utilizado um conjunto de *blendshapes* gerado pelo *software FaceGen Modeller 3.3* (Figura 7.2), que segundo Roesch et al. (2011), é uma ferramenta comercial projetada para a criação de faces 3D de forma realista, muito utilizada em jogos virtuais. Ela se baseia em um banco de dados com milhares de rostos humanos digitalizadas em 3D. As faces criadas pelo *software* variam em gênero, idade e etnia e podem ser alteradas através da interface de manipulação do *software* (Figura 7.3).

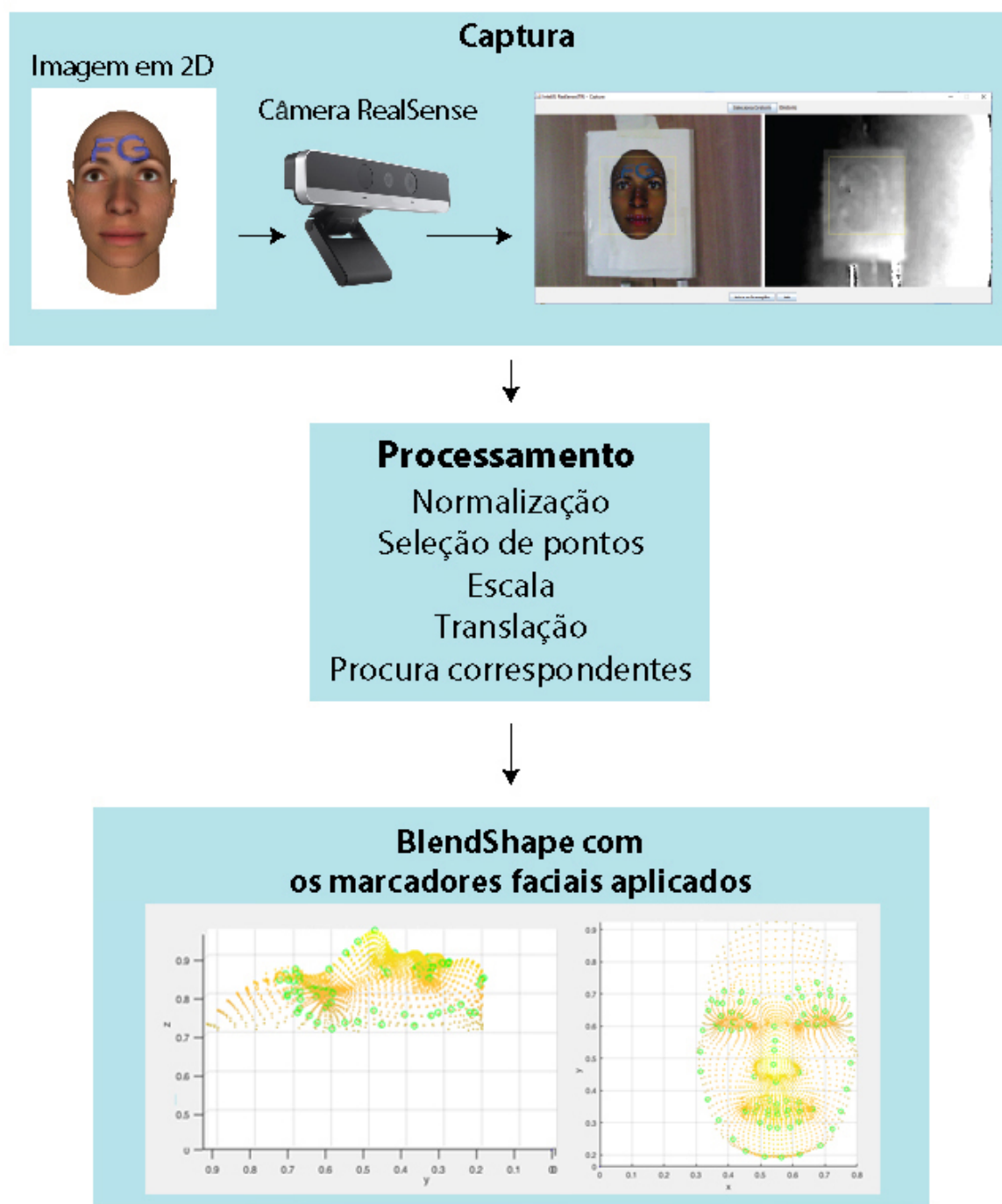
O *FaceGen Modeller 3.3* permite criar uma malha 3D personalizada através de fotografias, como pode ser vista na Figura 7.4. Ao exportar o objeto criado, são geradas cerca de 82 expressões baseadas nas FACS sendo uma delas a expressão neutra.

Por se tratar de um *software* comercial, foi utilizada a versão gratuita que possui a restrição de exportar os modelos com uma sigla FG localizado na região da testa, não comprometendo os resultados.

7.1.2 Inclusão de marcadores faciais

O subsistema de captura, processamento e redirecionamento utiliza as informações 3D dos marcadores faciais. No *dataset* dos *blendshapes* deve existir uma correspondência para estes dados, possibilitando a transferência de expressões.

Para o posicionamento dos marcadores, foi utilizada, no subsistema de geração de *dataset* e *blendshapes*, uma câmera *RealSense* que faz a identificação da face e dos seus pontos principais, como olhos, boca, nariz, sobrancelha, entre outros, em uma imagem 2D.

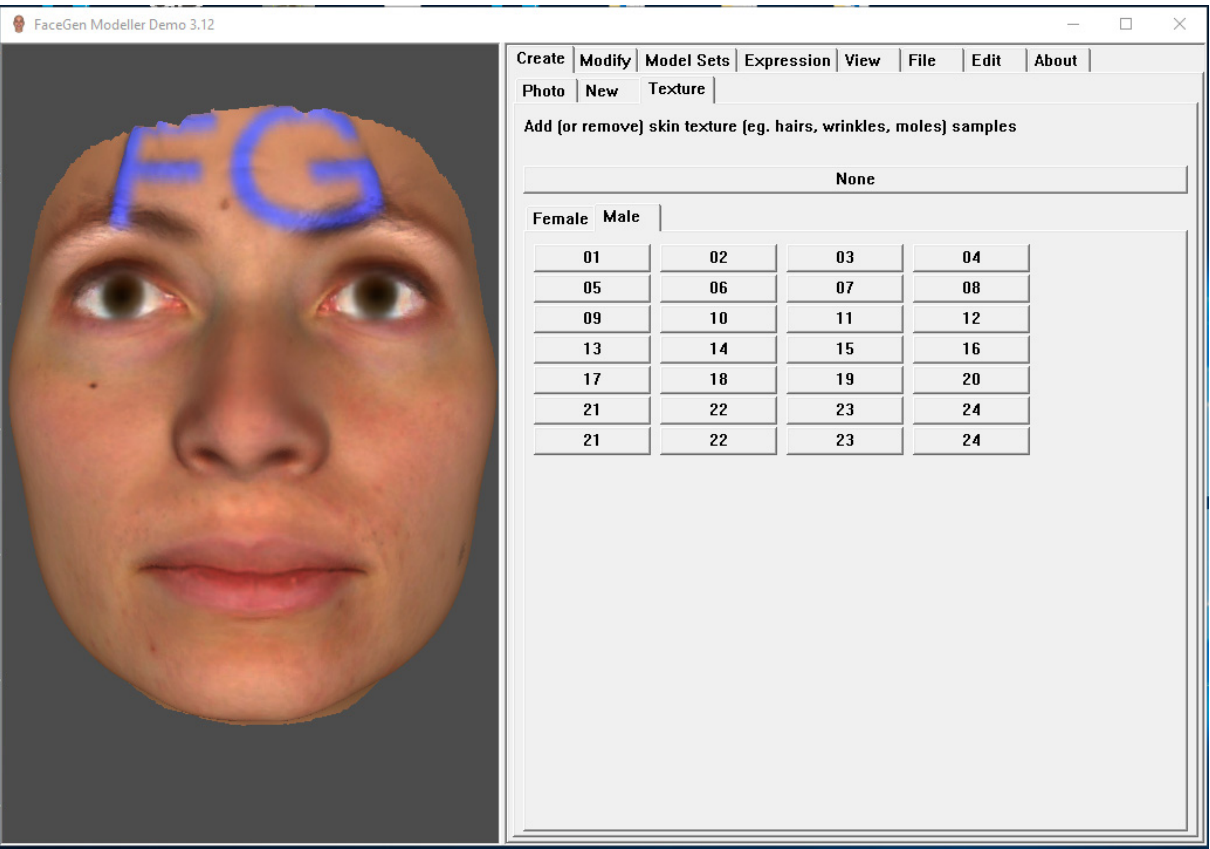
Figura 7.1 – Etapas da geração de *blendshapes* com associações de marcadores faciais.

Fonte: elaborada pelo autor.

Através de uma imagem gerada pelo *software* desenvolvido para as capturas, foi feita a aquisição inicial dos marcadores, como pode ser visto na Figura 7.5.

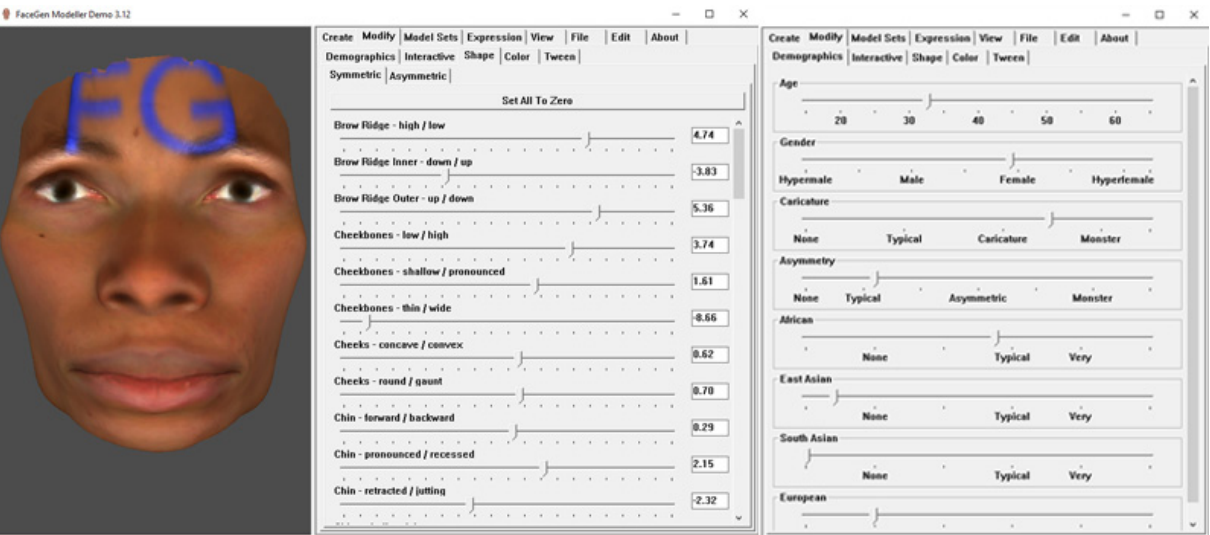
Depois da captura inicial, é feito um processamento para ajustar de forma semi-automática os pontos da imagem 2D com o modelo 3D da face neutra fornecido pelo *FaceGen Modeller 3.3*.

Figura 7.2 – Software FaceGen Modeller 3.3.



Fonte: elaborada pelo autor.

Figura 7.3 – Interface de manipulação do software FaceGen Modeller 3.3.



Fonte: elaborada pelo autor.

Figura 7.4 – Interface de manipulação do *software FaceGen Modeller 3.3*.

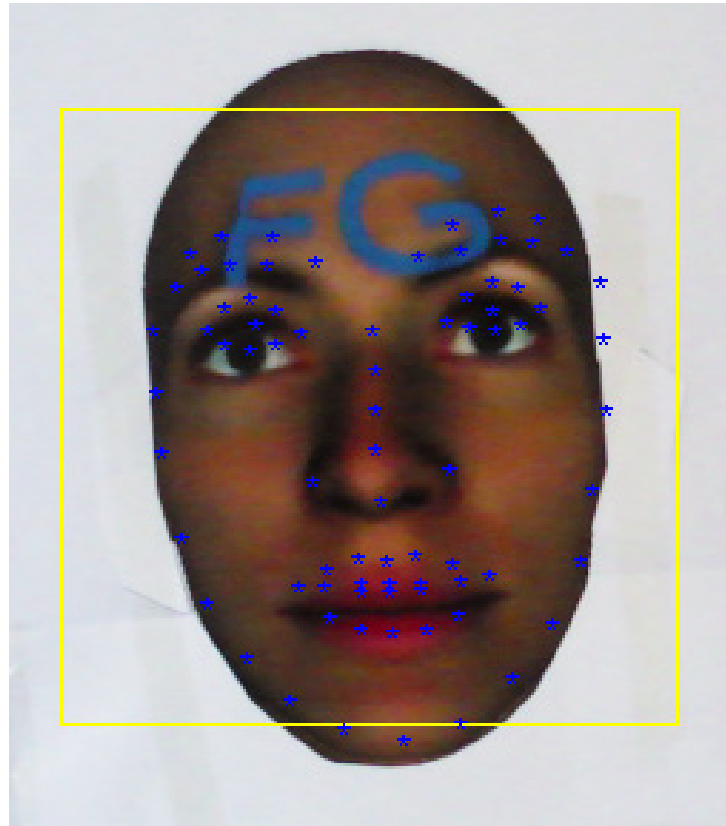
Fonte: elaborada pelo autor.

7.1.3 Ajuste dos marcadores faciais

O ajuste dos marcadores faciais, capturados na imagem 2D para o modelo 3D, é iniciado com a normalização dos pontos do modelo 3D e da imagem 2D. Esta normalização é feita para o ajuste da escala. Para ajustar a escala, são selecionados de forma manual, no modelo 3D, os pontos correspondentes aos dois olhos. Na imagem capturada pela câmera *RealSense*, estes pontos são conhecidos, pois a câmera faz a marcação automática. Após destacar estas referências entre o modelo 3D e os pontos 2D, é feito um ajuste de escala, usando como referência a distância entre os dois pontos e a diferença de tamanho entre eles. No final deste processo, todos os pontos são transladados, usando como origem o ponto do modelo 3D mais próximo da câmera, que é a ponta do nariz, e o ponto correspondente do modelo 2D, que é informado pela câmera. O processo pode ser visto na Figura 7.6

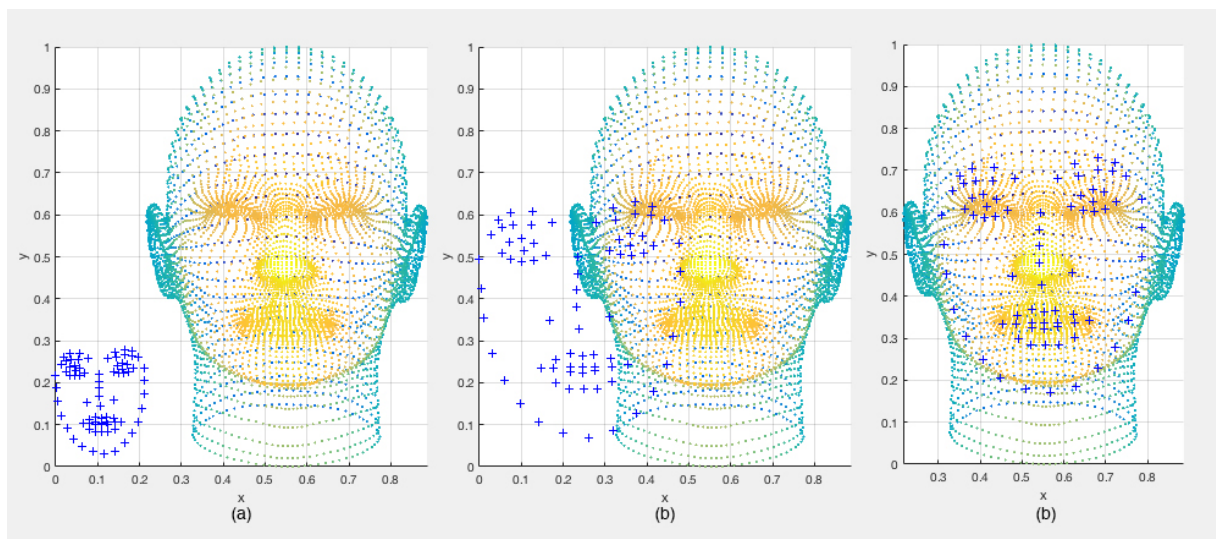
A última etapa do processamento é a identificação dos pontos 2D que correspondem ao modelo 3D. Para isto, é feita uma pesquisa entre cada ponto 2D através da distância euclidiana entre todos os pontos do modelo 3D para identificar qual é o ponto mais próximo, sendo este designado o ponto correspondente no modelo 3D para aquela posição. Depois que este processo é realizado, é armazenada a posição de cada ponto da matriz com relação aos seus vértices para aplicação nos outros modelos do *dataset*.

Figura 7.5 – Identificação da posição dos marcadores faciais, em uma imagem 2D, com a câmera *RealSense*.



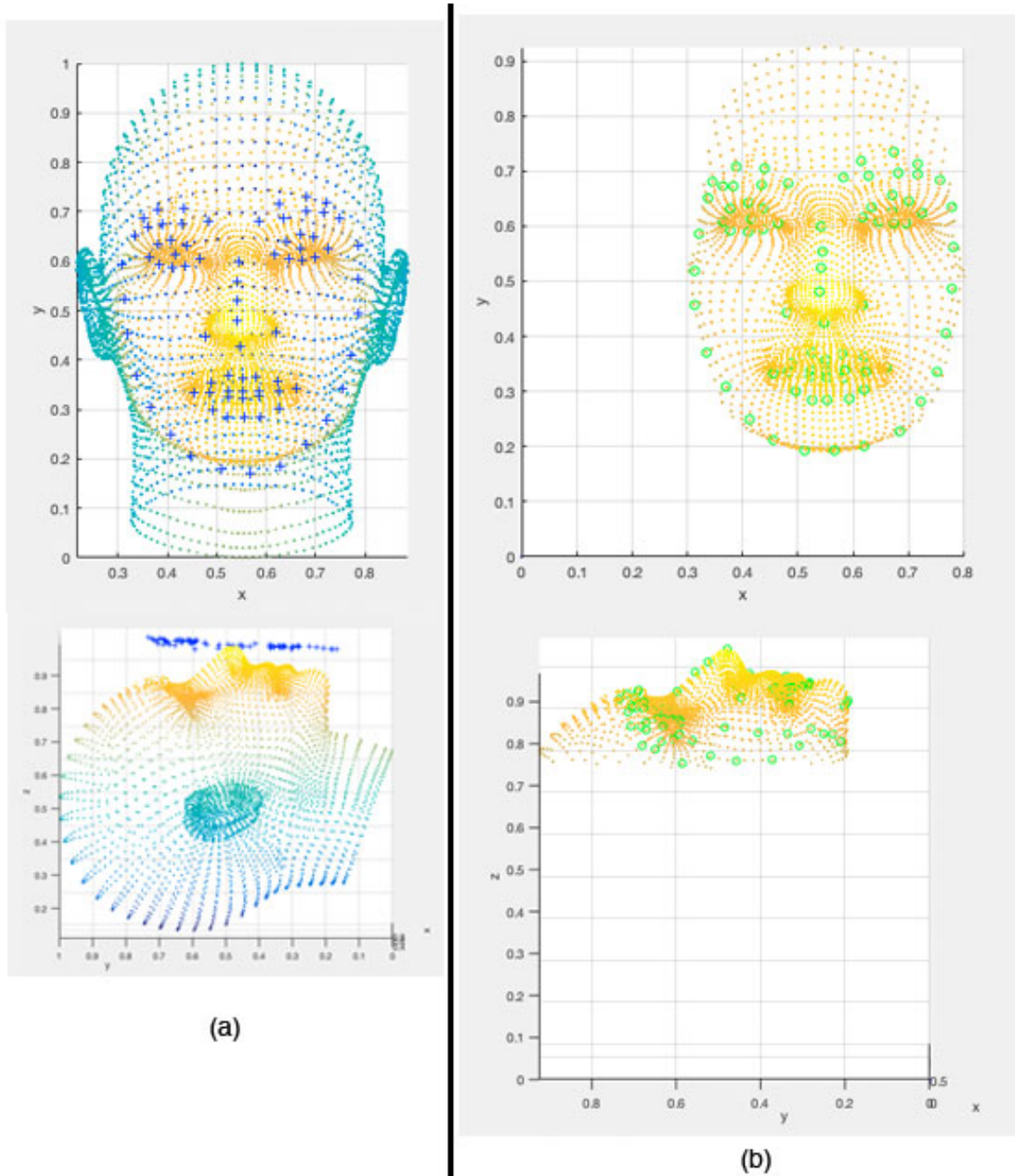
Fonte: elaborada pelo autor.

Figura 7.6 – Processo de normalização e escala dos modelos 2D e 3D. (a) modelo normalizado; (b) aplicado um ajuste nos pontos 2D para que a escalas entre os modelos fique igual; (c) Pontos 2D ajustados para o modelo 3D usando como referência o ponto mais próximo da câmera, que é a ponta do nariz.



Fonte: elaborada pelo autor.

Figura 7.7 – Correspondência dos pontos na imagem 2D com o modelo 3D. (a) Modelo 3D com os pontos 2D. Não existe um posicionamento quanto à profundidade dos pontos; (b) Pontos 3D identificados no modelo.



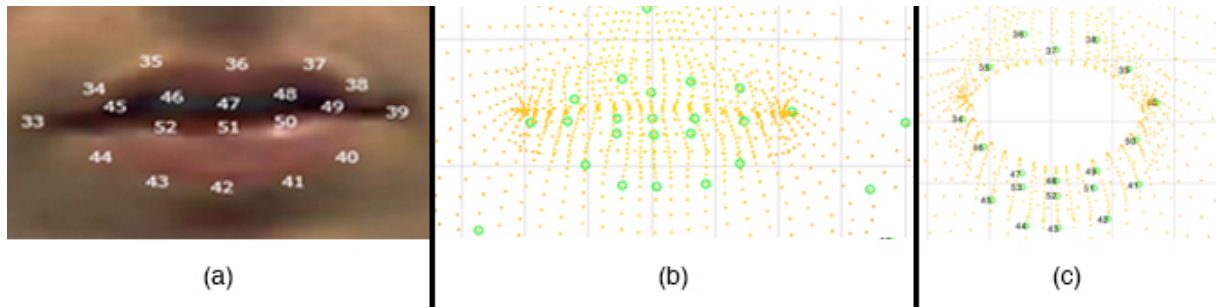
Fonte: elaborada pelo autor.

7.1.4 Ajuste dos marcadores faciais da região da boca

No processo de ajuste automático dos marcadores faciais do modelo 3D, a região da boca não é ajustada adequadamente, necessitando de um ajuste manual. Isto ocorre porque os pontos entre os lábios são muito próximos, sendo difícil, através do cálculo de distância, encontrar a sua correspondência correta como é feita com os outros pontos. Isto

pode ser observado na Figura 7.8, onde é visto que os pontos do lábio superior ficaram abaixo dos pontos do lábio inferior, depois do processo de ajuste automático.

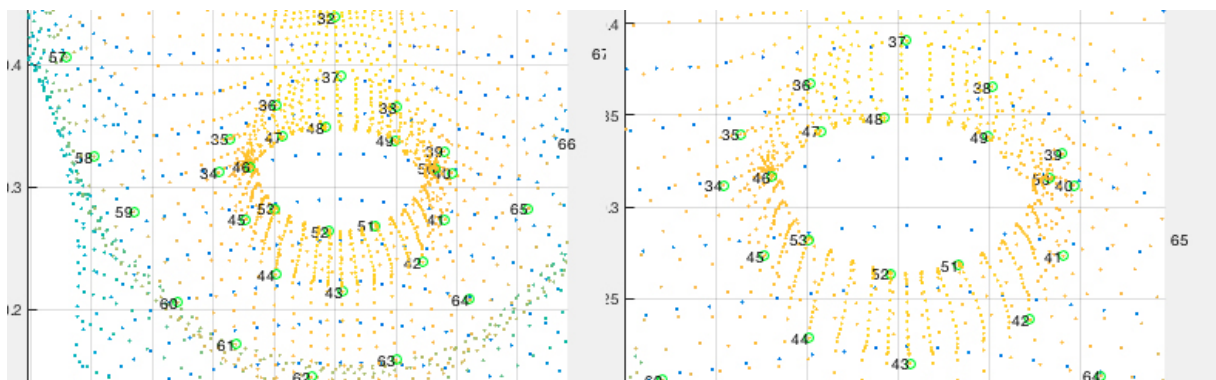
Figura 7.8 – Erro no processo automatizado de ajuste de marcadores. (a) posição correta dos marcadores faciais; (b) captura automática na região da boca, aparentemente correta; (c) marcadores faciais ajustados de maneira errada.



Fonte: elaborada pelo autor.

Após o ajuste manual, os pontos ficam na posição correta facilitando o cálculo dos pesos para os *blendshapes*. O ajuste pode ser visto na Figura 7.9.

Figura 7.9 – Marcadores faciais da região da boca ajustados manualmente.



Fonte: elaborada pelo autor.

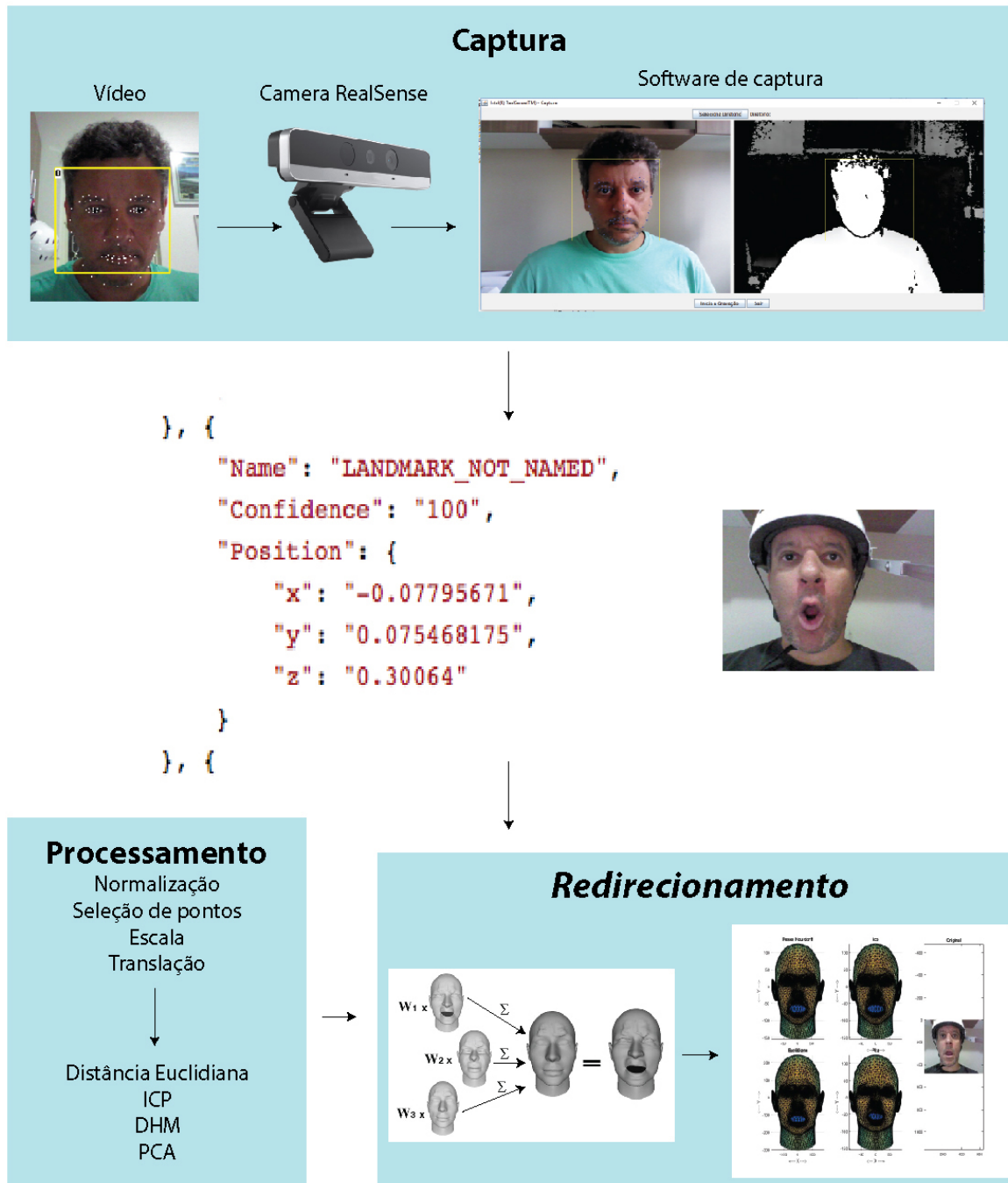
7.2 Subsistema de captura, processamento e redirecionamento

Este subsistema utiliza a câmera *RealSense* para a captura do vídeo, a posição da face na imagem e os marcadores faciais e suas informações em 3D. Este processo é realizado quadro a quadro. Na Figura 7.10, é mostrado o *pipeline* implementado neste subsistema.

7.2.1 Captura

A primeira etapa do *pipeline* é a captura da face de um ator, utilizando a câmera *RealSense*. Para isto, foi desenvolvido um capacete que faz com que a distância entre a

Figura 7.10 – Subsistema de captura, processamento e redirecionamento.

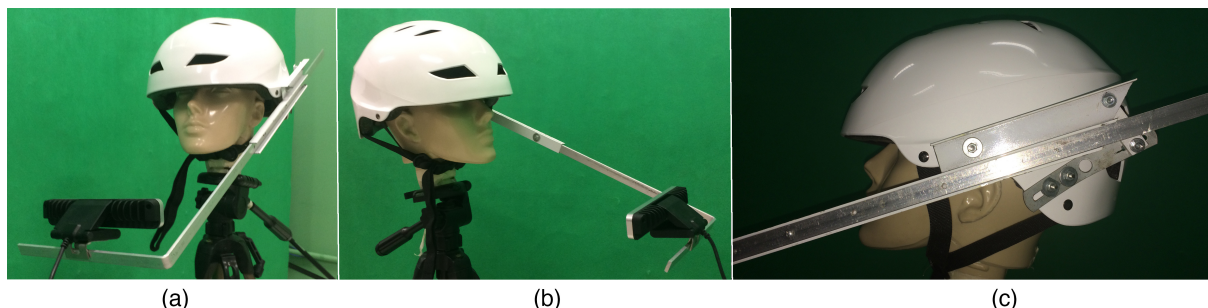


Fonte: elaborada pelo autor.

face do ator e a câmera permaneçam inalteradas. A câmera, desta forma, permanece em uma posição estacionária com relação aos movimentos da cabeça do ator. Este capacete permite um ajuste da distância da câmera e a face do ator bem como o ajuste de elevação, pois possui uma haste móvel como pode ser visto na Figura 7.11.

A câmera *RealSense* é colocada apoiada na haste do capacete e também pode ser

Figura 7.11 – Capacete utilizado para as capturas dos vídeos na fase inicial do processo. (a) e (b), visão do capacete e o dispositivo para suporte da câmera *RealSense*; (c) visão do ajuste da haste, possibilitando uma regulagem de distância da câmera e sua altura.



Fonte: elaborada pelo autor.

ajustada horizontalmente como visto na Figura 7.12, permitindo uma controle mais preciso para cada ator.

Figura 7.12 – Suporte da câmera *RealSense* na haste do capacete.



Fonte: elaborada pelo autor.

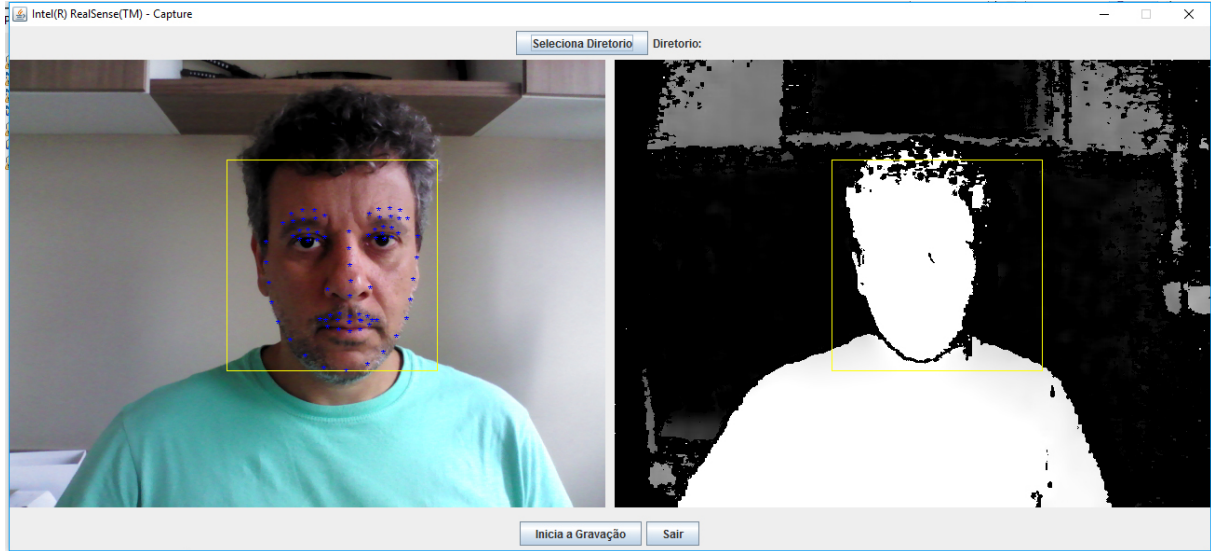
Para esta fase de captura, também foi desenvolvido um *software* que utiliza as bibliotecas que acompanham a câmera *RealSense* que possui algoritmos proprietários. Este *software* possui uma interface que mostra, em tempo real, a captura e os marcadores na face do ator assim como a imagem de profundidade (Figura 7.13).

Ao iniciar a captura por meio do *software*, o ator deve permanecer em frente à câmera com uma expressão neutra nos primeiros segundos. Após este tempo, o ator pode começar a atuar. Para a etapa de processamento, que será descrita adiante, é utilizado o primeiro quadro capturado, como sendo a expressão neutra. É através deste quadro que é realizada a calibração entre a face do ator e o *dataset*.

O módulo de captura exporta, para cada quadro, a posição dos marcadores faciais em 3D em um arquivo no formato JSON ¹ e uma imagem que corresponde à captura do

¹ JSON é um acrônimo para "JavaScript Object Notation", é um formato leve para intercâmbio de dados computacionais. JSON é um subconjunto da notação de objeto de JavaScript, mas seu uso não requer JavaScript exclusivamente

Figura 7.13 – Visão da interface do *software* que faz a captura das imagens, utilizando as bibliotecas de desenvolvimento da câmera *RealSense*. Do lado direito, a imagem com os marcadores faciais; do lado esquerdo, a imagem de profundidade.



Fonte: elaborada pelo autor.

quadro do vídeo, como pode ser visto na Figura 7.14. Estas informações são armazenadas para posterior processamento.

7.2.2 Processamento e a geração dos pesos dos *blendshapes*

Nesta etapa, são calculados os pesos dos *blendshapes* para cada quadro capturado. Isto envolverá uma série de etapas para ajuste dos pontos, antes que seja submetido aos algoritmos de reconhecimento de padrões.

A primeira etapa do processamento é o ajuste da face neutra da captura com o ajuste da face neutra do *dataset*. Este ajuste é feito através da normalização dos pontos 3D de cada captura juntamente com cada expressão do *dataset*, como pode ser visto na Figura 7.15.

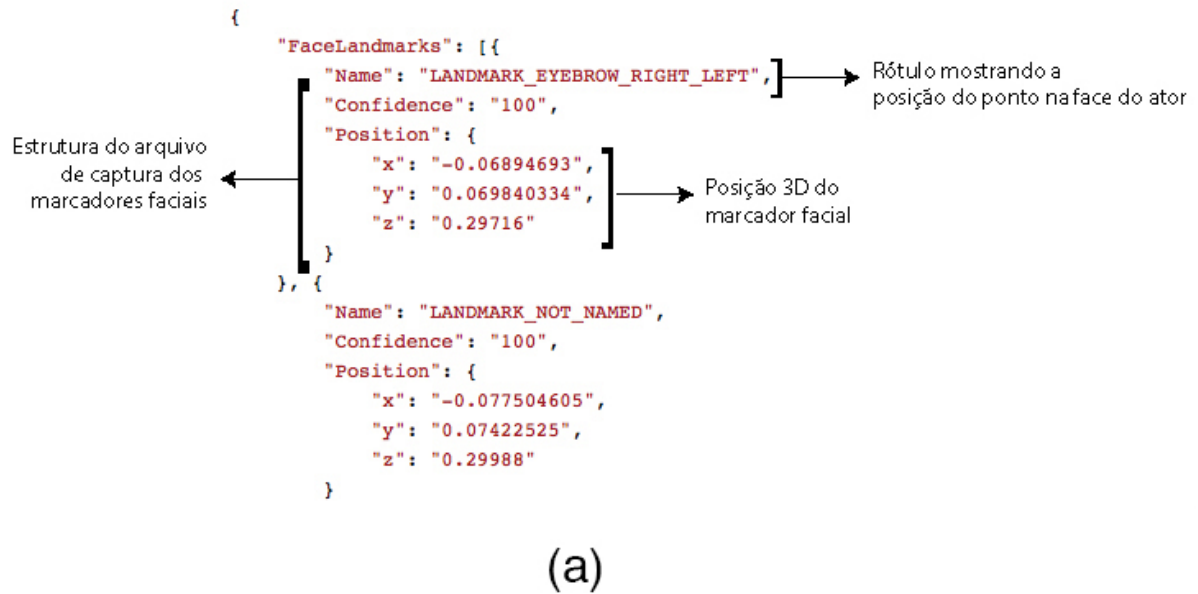
A normalização é realizada, aplicando-se a equação 7.1.

$$NNorm = (N - \min(N)) / (\max(N - \min(N))); \quad (7.1)$$

onde N é um vetor 3D.

A próxima etapa do ajuste é a escala. Ela é calculada por meio de um conjunto de pontos que têm uma correspondência entre si. No caso, foram escolhidos os pontos de número 27 a 30 do conjunto de marcadores faciais previamente capturados e ajustados no *dataset*. Para isto, é feito o seguinte cálculo: dado dois conjuntos de pontos denominados

Figura 7.14 – Exemplos de arquivos exportados pelo *software* de captura. (a) arquivo *json* com as informações dos marcadores faciais; (b) imagem do quadro correspondente às informações do arquivo *json*.

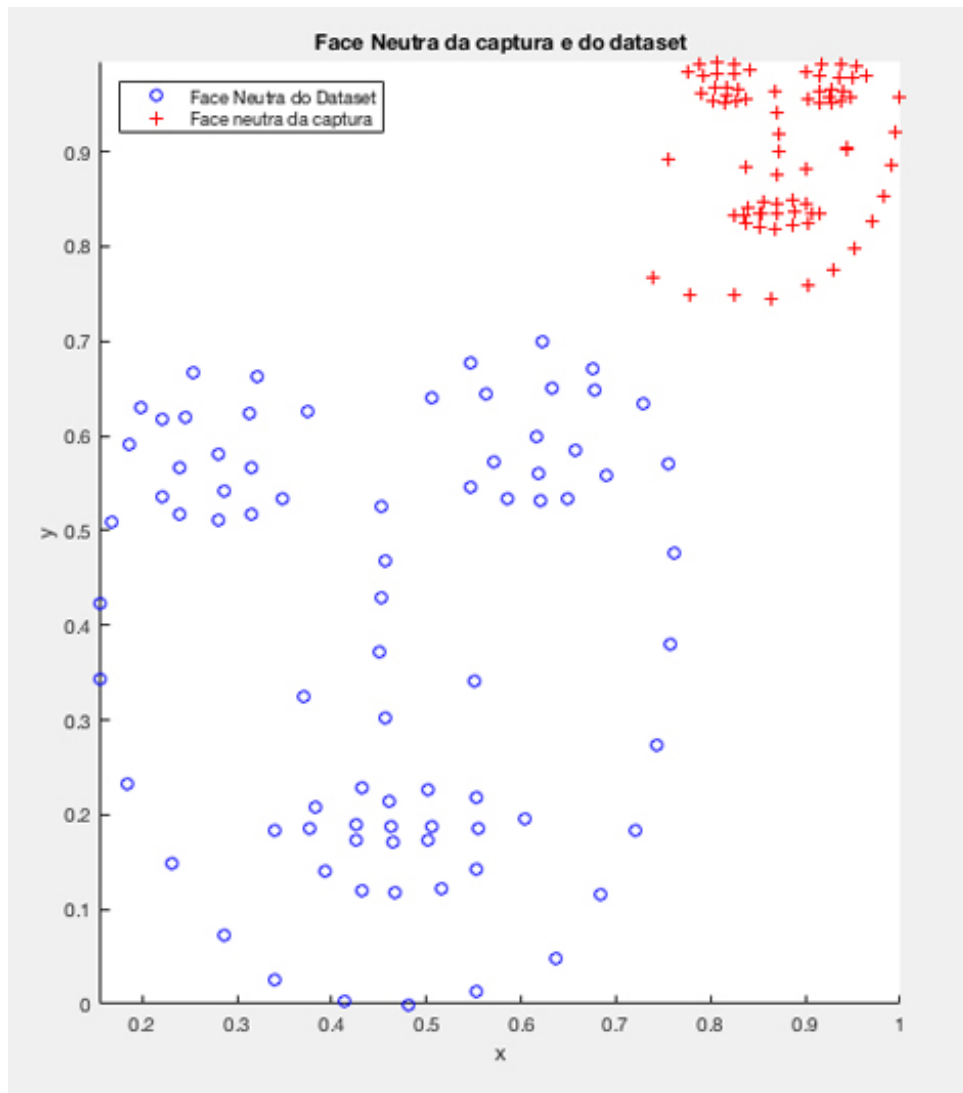


Fonte: elaborada pelo autor.

A e B , é calculada a razão entre as distâncias dos pontos de cada conjunto (equação 7.2), sendo esta o fator de escala. O resultado pode ser visto na Figura 7.16.

$$(\sqrt{(A_{x1} - A_{x2})^2 + (A_{y1} - A_{y2})^2}) / (\sqrt{(B_{x1} - B_{x2})^2 + (B_{y1} - B_{y2})^2}) \quad (7.2)$$

A próxima etapa é repetida para todos os quadros da captura, envolvendo a leitura

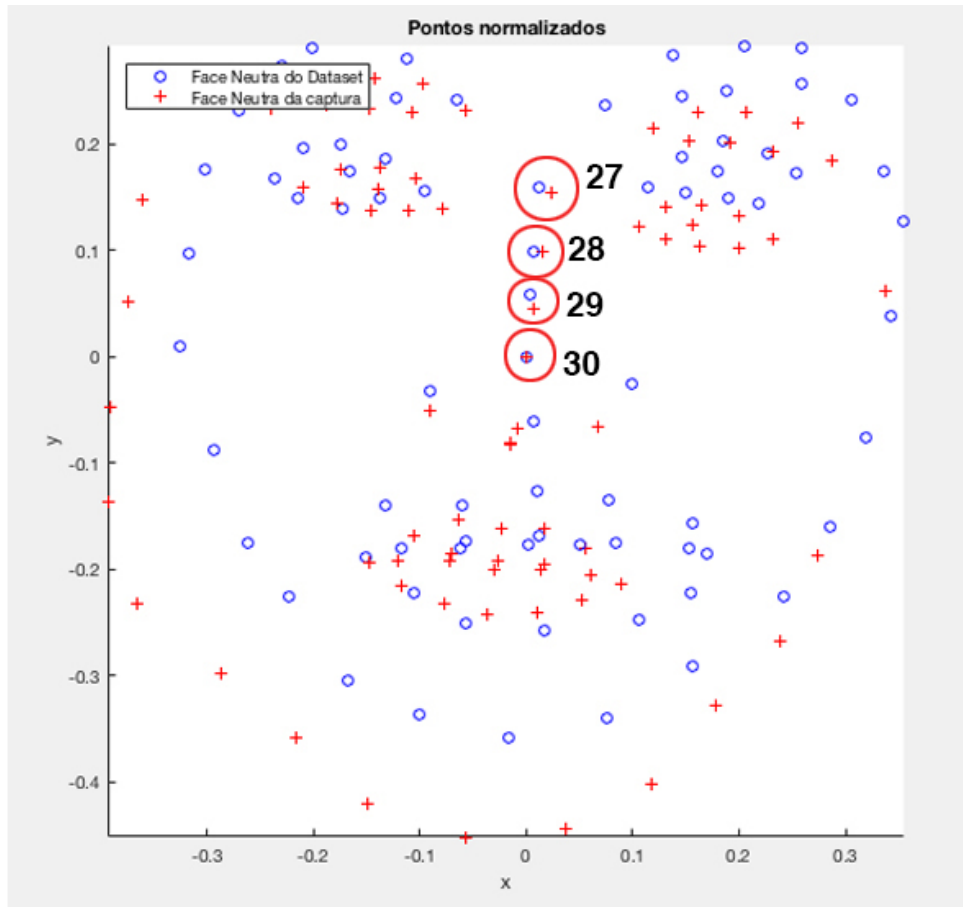
Figura 7.15 – Ajuste dos pontos da face neutra do *dataset* e da captura.

Fonte: elaborada pelo autor.

das informações de cada quadro através do arquivo *json* e seu ajuste com a face neutra ajustada anteriormente. Isto envolve novamente as etapas de normalização e escala. O posicionamento final é a centralização dos pontos através do ponto mais próximo da câmera, que é o da ponta do nariz. Esta translação ocorre para minimizar as distâncias entre os pontos. O resultado pode ser visto na Figura 7.17.

Após o ajuste inicial, para cada quadro capturado, é calculado o deslocamento dos pontos para a face neutra. Este deslocamento é aplicado à face neutra do *dataset*, formando, assim, um conjunto de pontos que tenha a geometria dos *blendshapes*, mas com o deslocamento da captura. Com isto, pode-se utilizar *blendshapes* com características faciais diferentes, pois o posicionamento dos pontos da captura são transportados para a face neutra do *dataset*. O resultado pode ser visto na Figura 7.18.

A face neutra ajustada do *dataset* é submetida aos algoritmos de reconhecimento

Figura 7.16 – Pontos ajustados entre a face neutra da captura e do *dataset*.

Fonte: elaborada pelo autor.

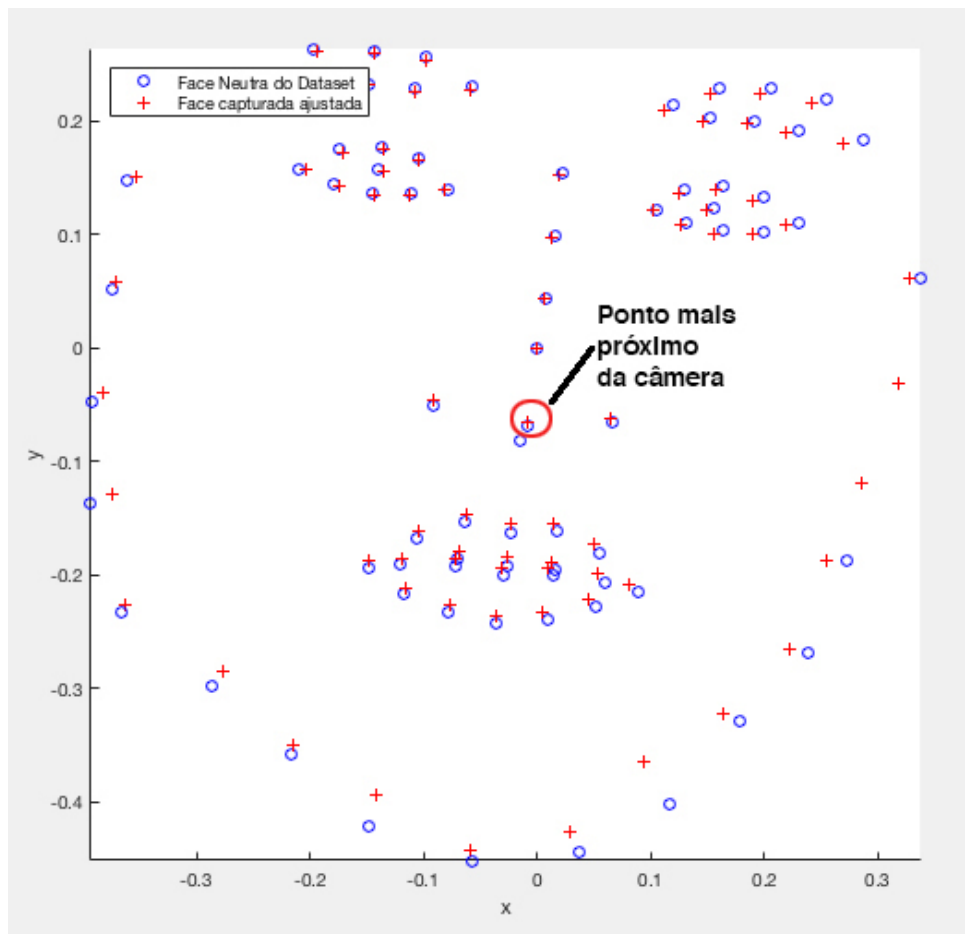
de padrões selecionados, apresentados no Capítulo 4. Os algoritmos devem calcular a distância entre os pontos da face do ator com os pontos pré-definidos nos *blendshapes*. Para os resultados encontrados, calculando a distância da face do ator com todos os *blendshapes* do *dataset*, é feita uma normalização entre os pontos, deixando-os dentro do intervalo de $[0, 1]$, sendo 1 o valor mais próximo entre o conjunto de pontos, e zero o valor mais distante entre os pontos. Na Figura 7.19, é mostrado um exemplo de cálculo das distâncias dos pontos da face capturada do ator com o conjunto de *blendshapes*.

7.2.3 Redirecionamento

O módulo de redirecionamento utiliza os valores encontrados nos cálculos de distância entre os pontos da face do ator e os *blendshapes*. Esta fase gera um novo modelo que, a partir da face neutra do *dataset*, mistura vários outros modelos levando em conta a proximidade da face capturada.

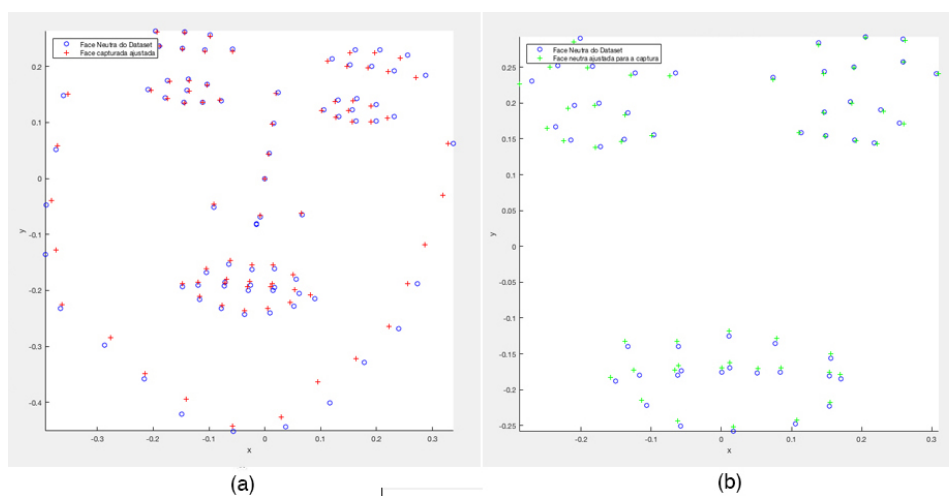
Para o redirecionamento, são utilizados os algoritmos abordados no Capítulo 4. Os pesos encontrados correspondem ao cálculo da distância entre os *blendshapes* e a face

Figura 7.17 – Pontos centralizados, usando o ponto mais próximo da câmera que é o ponto localizado na frente do nariz.



Fonte: elaborada pelo autor.

Figura 7.18 – Exemplo do ajuste dos pontos em função da face neutra. (a) pontos da captura; (b) pontos ajustados à face neutra do *dataset*.



Fonte: elaborada pelo autor.

Figura 7.19 – Exemplo do cálculo das distâncias entre os pontos da face capturada do ator com os *blendshapes*.



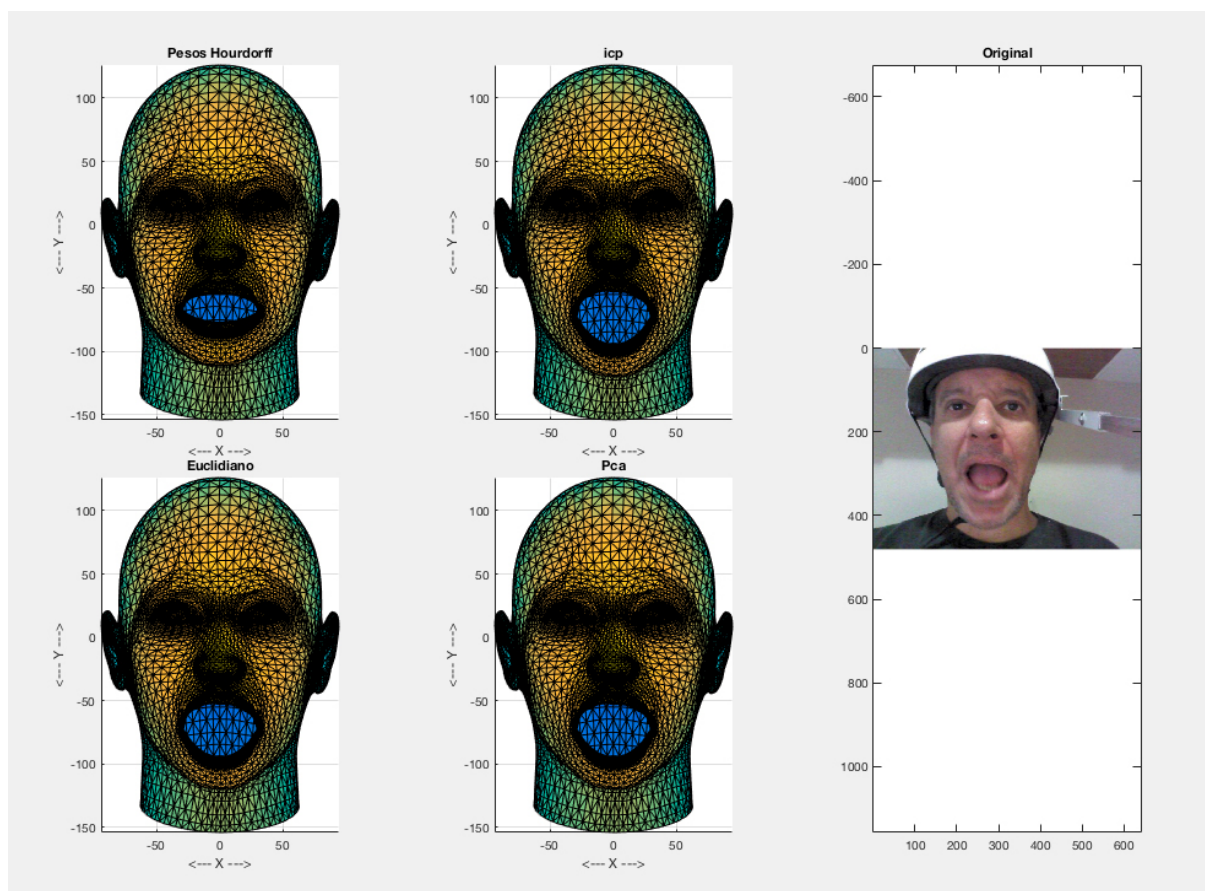
Fonte: elaborada pelo autor.

capturada. Neste módulo, podem ser configuradas várias combinações de *datasets* com números diferentes de expressões. Dependendo dos tipos de expressões presentes no *dataset*, o resultado final se altera.

Para as aplicações dos pesos, os valores do processo de cálculo de distância entre os pontos é normalizado entre os valores 0 e 1, sendo o valor 1 o mais próximo da expressão do *dataset*, e zero o mais distante. Foi usada a seguinte equação:

$$f = \sum_{k=0}^n W_k B_k \quad (7.3)$$

onde W_k é o valor encontrado no cálculo das distâncias, e B_k o *blendshape* correspondente. Estes valores são somados e, no final, é somado à face neutra do *dataset*. O resultado pode ser visto na Figura 7.20.

Figura 7.20 – Exemplo de aplicação de pesos nos *blendshapes*.

Fonte: elaborada pelo autor.

8 Testes e análise de resultados

Este capítulo apresenta a análise dos resultados obtidos a partir de testes realizados utilizando-se o protótipo do sistema.

8.1 Ambiente experimental

O ambiente experimental utilizado neste trabalho é descrito a seguir.

8.1.1 Componentes de *Hardware*

8.1.1.1 Computador

Para o desenvolvimento do projeto, foram utilizados dois *notebooks*. O primeiro possui a seguinte configuração básica e foi utilizado para as capturas com a câmera *RealSense*:

- *Notebook* Asus
- Sistema Operacional de 64 *bits*, processador com base em x64, Windows 10 Pro
- Placa gráfica integrada
- Processador: Intel Core(TM) i5-3317U CPU (1.7GHz);
- Memória RAM: 4 GB;
- Disco rígido: 500 GB;

O segundo possui a seguinte configuração e foi utilizado para o processamento dos quadros capturados:

- *Notebook MacBook Pro*
- Sistema Operacional de 64 *bits*, processador com base em x64, Mac os Sierra
- Placa gráfica Intel HD *Graphics* 4000
- Processador: Intel Core(TM) i7 CPU (2.9GHz);
- Memória RAM: 8 GB;
- Disco rígido SSD: 250 GB;
- Disco rígido SATA: 500 GB;

8.1.1.2 Câmera *RealSense*

A animação facial baseada em performance pode ser feita utilizando-se dois métodos: rastreamento de imagem e aquisição geométrica. Com novas pesquisas nesta área, é possível digitalizar a geometria da face usando-se *scanners* para criar uma malha facial e animá-la através da simulação dinâmica de tecidos faciais e músculos. Dispositivos como câmeras com capturas RGB-D, também podem ser usadas nesse processo. Como exemplo, pode-se citar o *Kinect* e a *RealSense* (Figura 8.1).

Figura 8.1 – Dispositivos com captura de imagens RGB-D. À esquerda o *Kinect* da Microsoft, e à direita o *RealSense* da Intel.



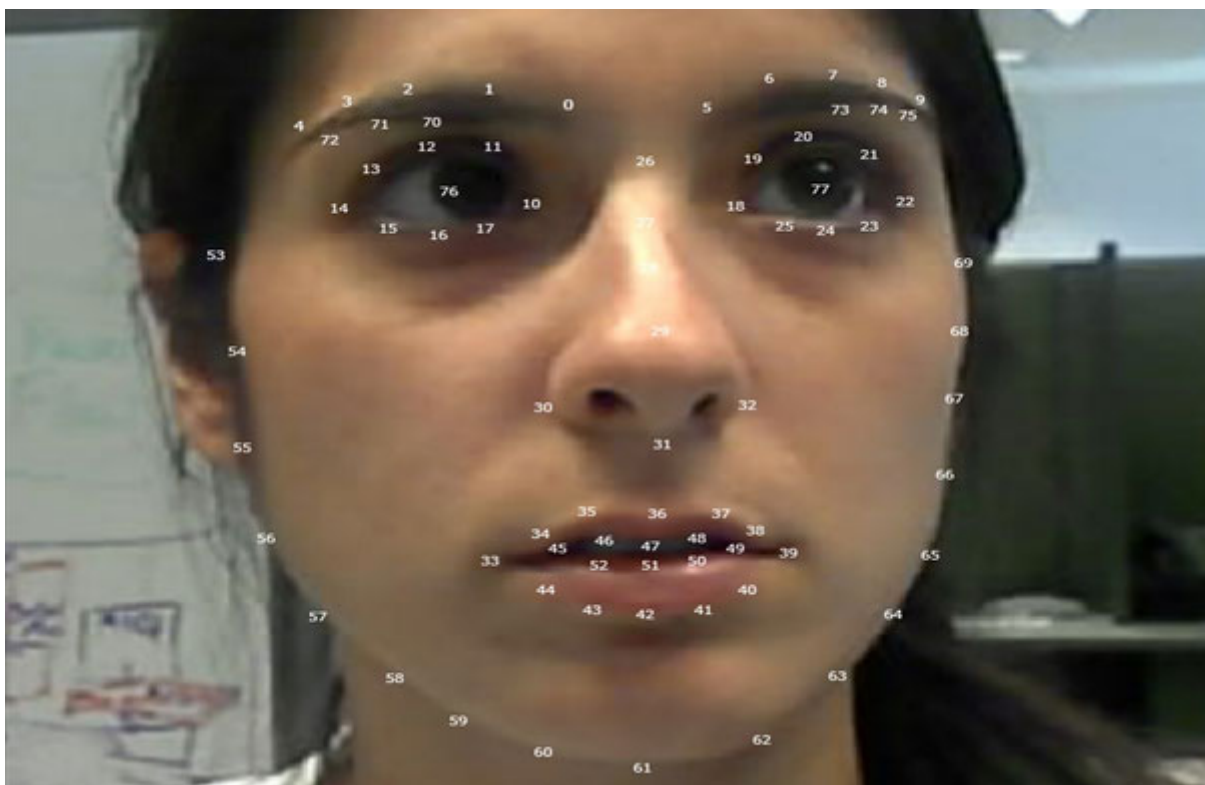
Fonte: elaborada pelo autor.

A câmera *RealSense* é um dispositivo desenvolvido pela Intel que, além de possuir a capacidade de captura de imagem RGB-D, possui no seu *kit* de desenvolvimento (*SDK*) uma série de algoritmos implementados como identificação de expressões faciais, identificação facial, identificação de face em uma imagem e seus pontos de destaque, que são os marcadores faciais entre outras implementações. Por ser uma câmera comercial, os seus algoritmos não são de domínio público.

Para esta pesquisa, foi utilizada a capacidade da câmera para a detecção da face e seus marcadores faciais 2D e 3D. Estes marcadores faciais são identificados por meio de sua numeração como mostrado na Figura 8.2. No total, a câmera *RealSense* captura 78 pontos na face.

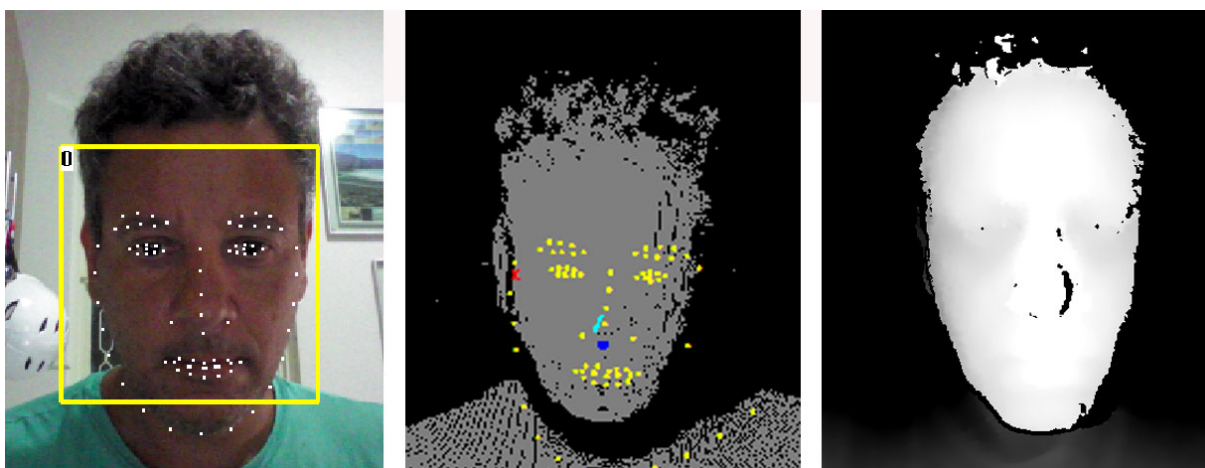
Cada marcador facial possui um rótulo de identificação, sendo agrupado pela área da face onde se encontra. Por exemplo, os marcadores de número 76 e 77 correspondem à área central dos olhos, enquanto os marcadores 26, 29, 30, 31 e 32 correspondem à área do nariz. Estas informações são apresentadas usando o posicionamento da câmera, ou seja, apenas informações de 2D e a as informações em 3D usando as coordenadas do ambiente. Para os marcadores faciais, a câmera *RealSense* apresenta as informações em metros, enquanto que as informações de profundidade são informadas em milímetros.

Além dos marcadores faciais, a câmera detecta a posição de uma face em um vídeo, como mostrado na Figura 8.3. O dispositivo tem a capacidade de detectar várias faces ao

Figura 8.2 – Marcadores faciais capturados pela câmera *RealSense* da Intel.

Fonte: Duffy Bobby (2014)

mesmo tempo.

Figura 8.3 – Detecção da face por meio da câmera *RealSense* da Intel.

Fonte: elaborada pelo autor.

8.1.2 Componentes de *Software*

Neste trabalho foi desenvolvido um protótipo utilizando-se os seguintes softwares:

- Plataforma de desenvolvimento Java Eclipse, versão Neon, 64 *bits*;
- Linguagem de desenvolvimento Java, versão 1.8.

Para os testes, foi utilizado o *software* Matlab, versão R2016b, com os seguintes *plugins* instalados:

- *MATLAB and Simulink Student Suite*;
- *Computer Vision System Toolbox*; e
- *Neural Network Toolbox*.

8.2 Experimentos realizados

Para a realização dos testes, utilizo-se, no módulo inicial do *pipeline* implementado pelo protótipo, um *dataset* gerado através do *software* *FaceGen Modeller 3.3*, composto por 82 expressões faciais baseadas nas FACS, sendo uma destas a expressão neutra. Estas expressões correspondem às emoções básicas, como alegria, tristeza, medo, nojo, espanto, choro entre outras. O *dataset* completo pode ser visto no Apêndice A.

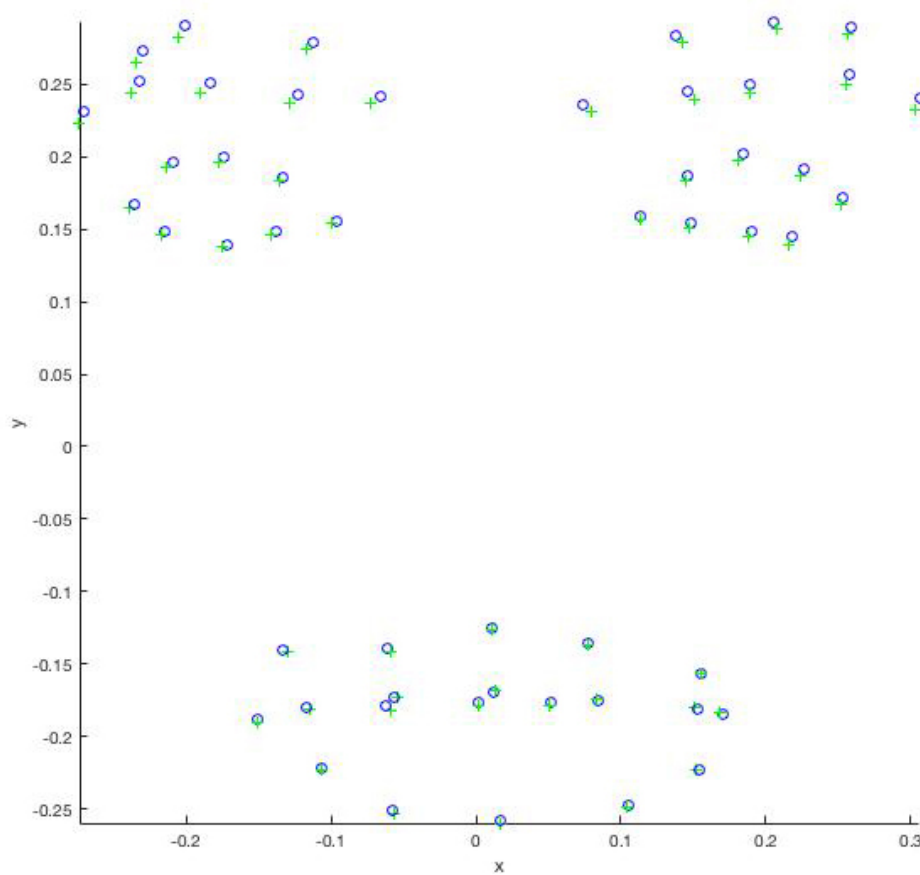
Por meio do uso do protótipo e do capacete apresentados no Capítulo 7, foi capturado um vídeo de 3 minutos a uma velocidade de 14 quadros por segundo, resultando em uma amostra de 2691 quadros.

Para a determinação da acurácia dos algoritmos de rastreamento selecionados para teste, procedeu-se no cálculo da diferença entre o posicionamento dos marcadores faciais capturados no módulo inicial (considerados como referência ou *ground truth*) e o posicionamento dos marcadores obtidos após a aplicação dos algoritmos, no módulo de processamento. O cálculo desse erro foi feito por meio da distância euclidiana.

Para o experimento, foram selecionados, aleatoriamente, 4 conjuntos de *blendshapes*. O primeiro, contendo cerca de 25% do total do *dataset* (21 *blendshapes*), o segundo conjunto com 50% do total (41 *blendshapes*), o terceiro com 75% do total (62 *blendshapes*) e o último com 100% do conjunto (81 *blendshapes*).

Foram selecionados, do total de 78 marcadores faciais, apenas 52 que correspondem aos pontos internos do rosto que abrangem a área da boca, olhos e sobrancelhas como visto na Figura 8.4. Estas são as áreas utilizadas por Weise et al. (2011) e Li et al. (2013) nos seus respectivos *pipelines* apresentados no Capítulo 5.

Figura 8.4 – Pontos selecionados para os testes, totalizando 52 marcadores faciais.



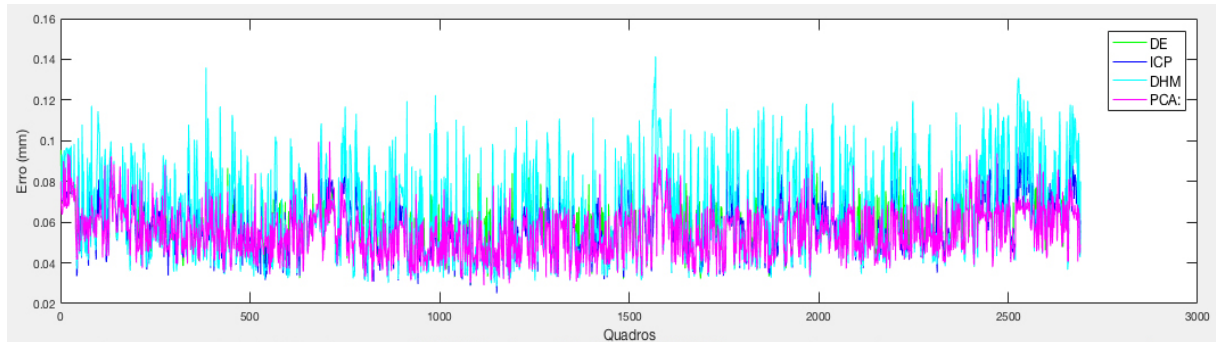
Fonte: elaborada pelo autor.

8.2.1 Resultados obtidos na comparação dos algoritmos

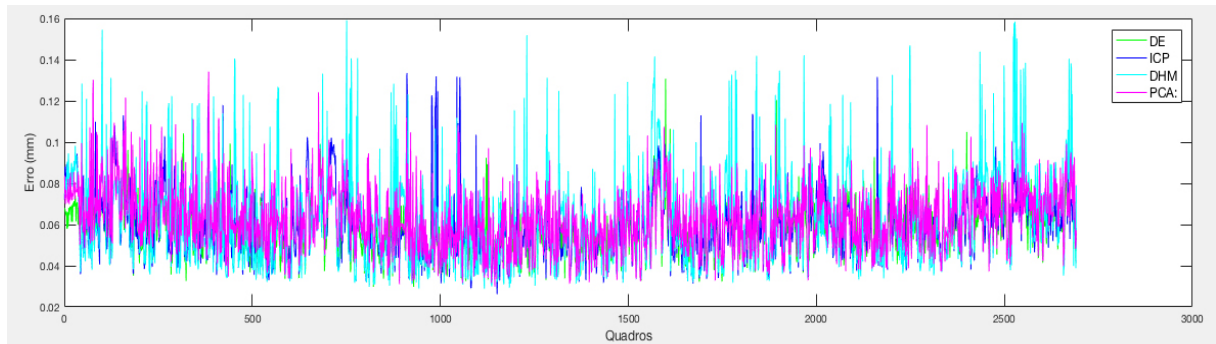
São apresentados, na Figura 8.5, os resultados obtidos no processamento de cada conjunto de *blendshapes* submetidos aos algoritmos selecionados na literatura e discutidos no Capítulo 4. (a) 25% (21 *blendshapes*); (b) 50% (41 *blendshapes*); (c) 75% (62 *blendshapes*) e (d) 100% (81 *blendshapes*).

As Figuras 8.6, 8.7, 8.8 e 8.9 exibem a variação do erro entre as marcações faciais obtidas a partir da captura da face (*ground truth*) e as marcações geradas após a aplicação dos algoritmos de rastreamento, para todos os quadros do vídeo de entrada. As figuras também destacam, como exemplo, o quadro que apresenta o maior erro, para cada algoritmo.

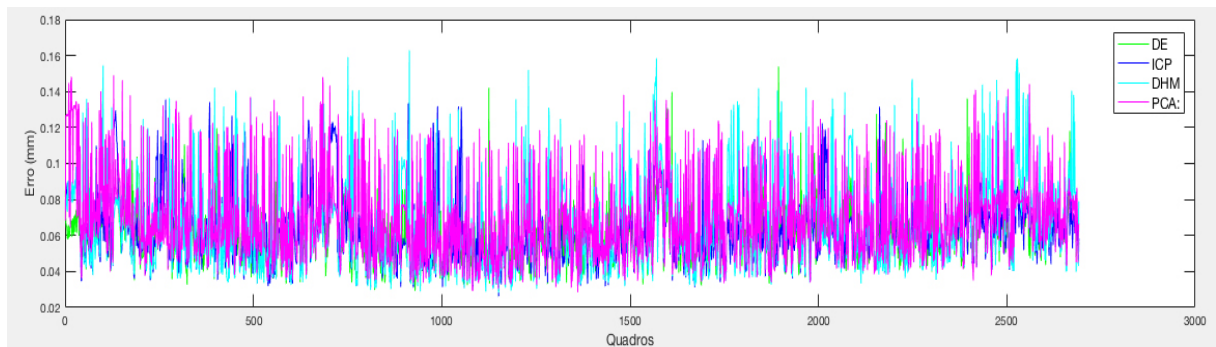
Figura 8.5 – Variação do erro entre o posicionamento dos marcadores faciais gerados no módulo de processamento em função dos capturados no módulo inicial para a quantidade de *blendshapes* selecionados.



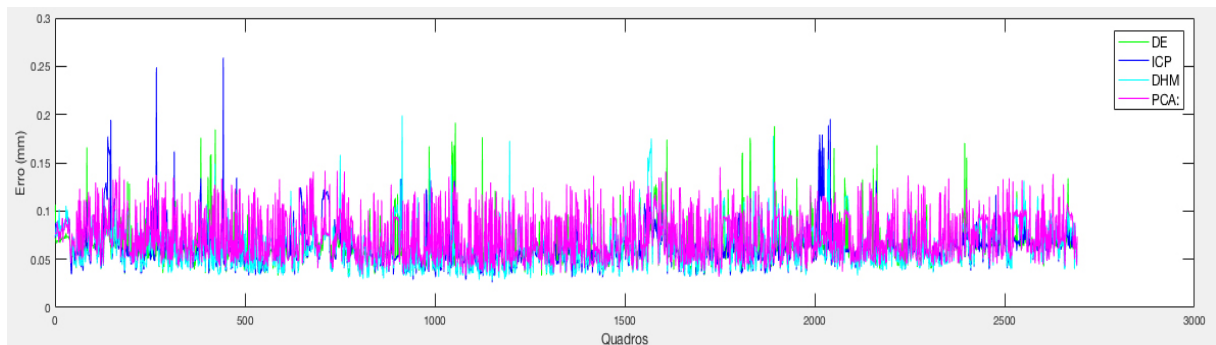
(a)



(b)



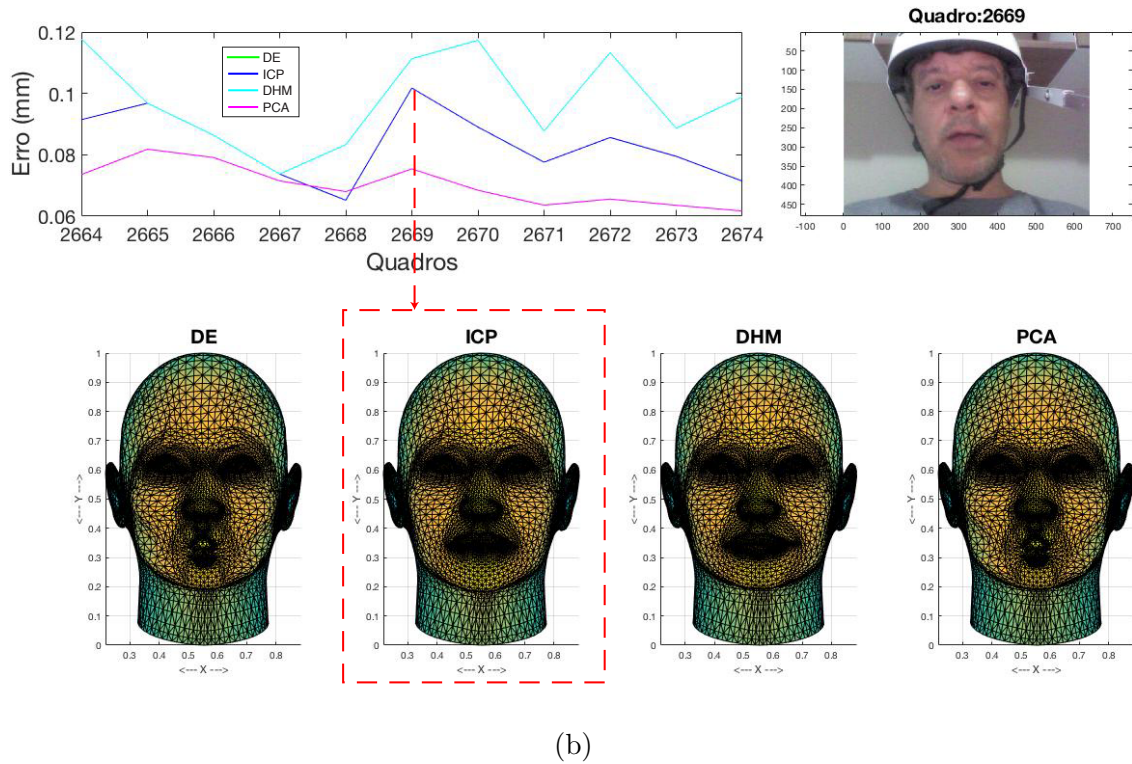
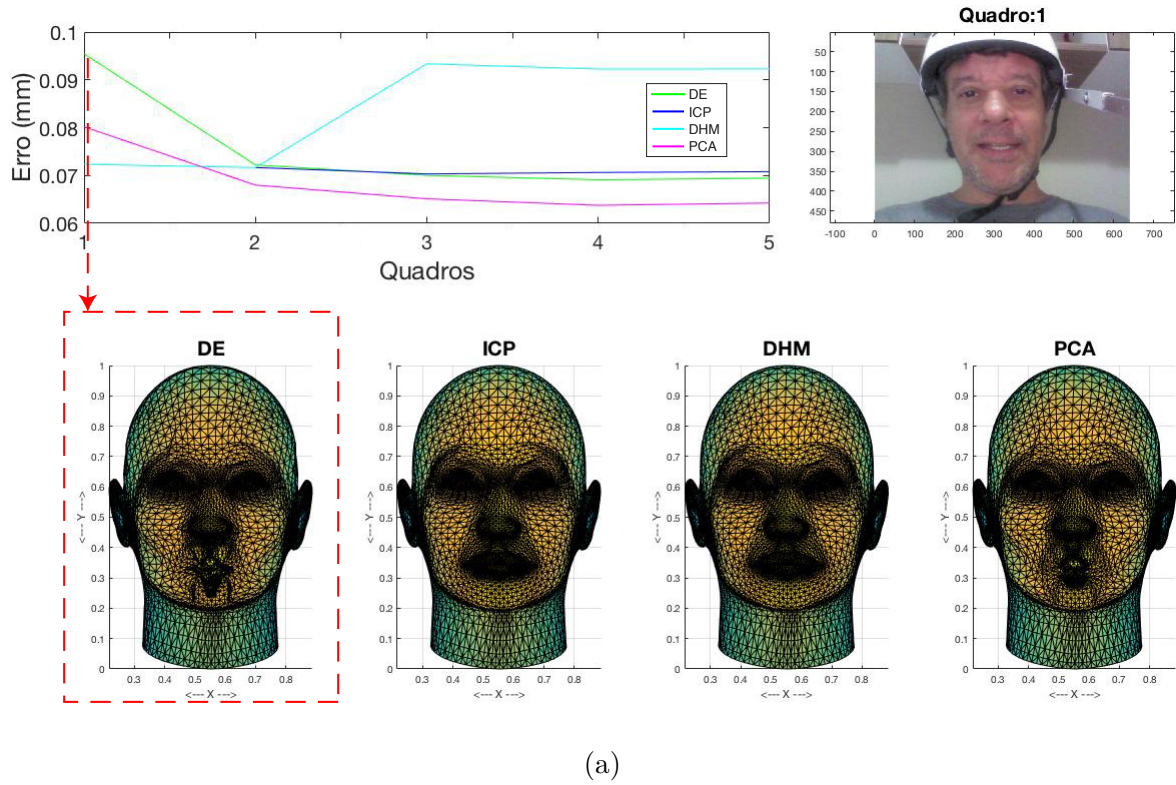
(c)

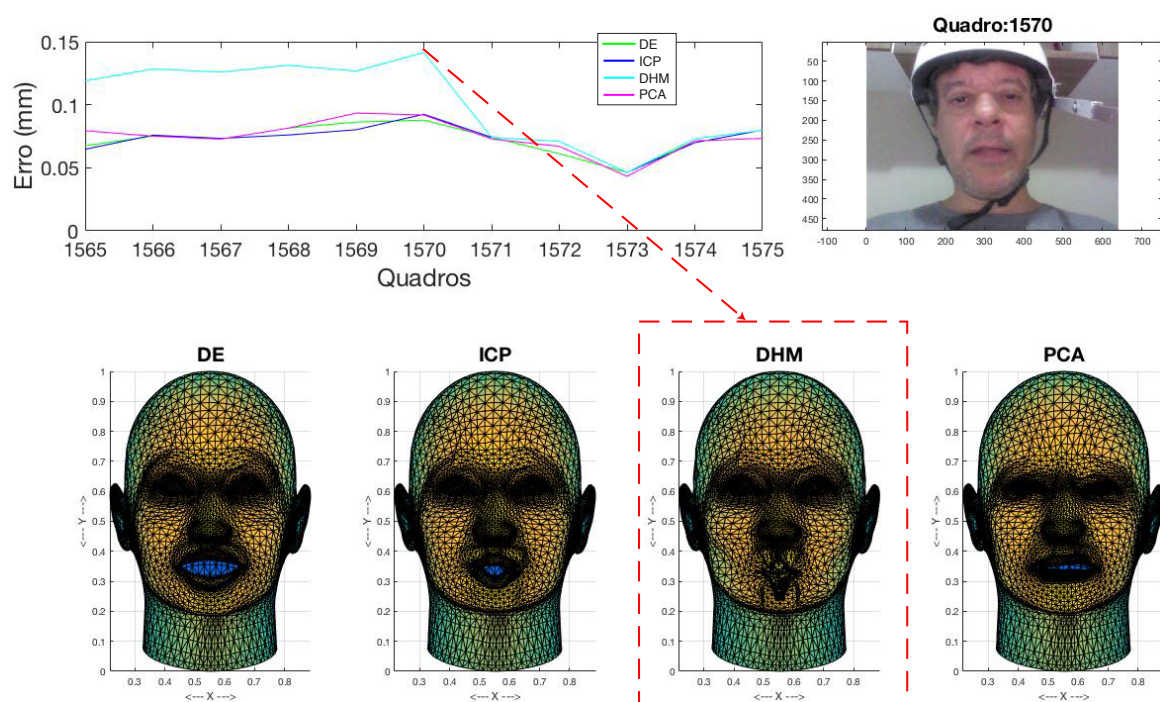


(d)

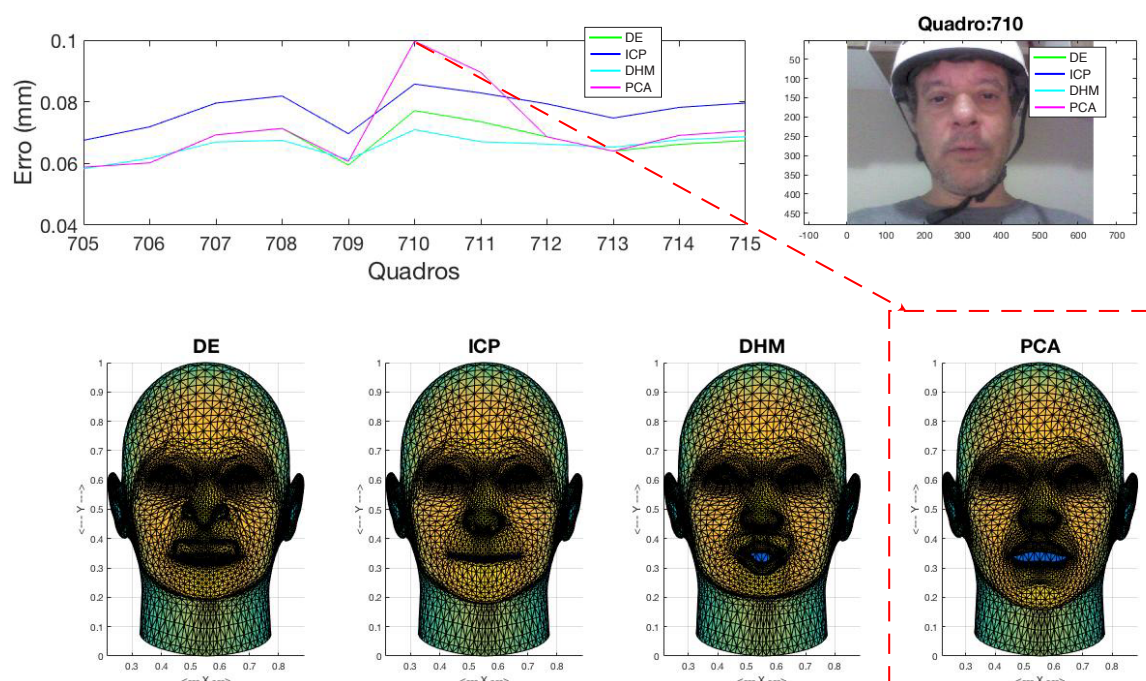
Fonte: elaborada pelo autor.

Figura 8.6 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 25% do *dataset*. (a) Distância Euclidiana; (b) ICP; (c) DHM; (d) PCA.





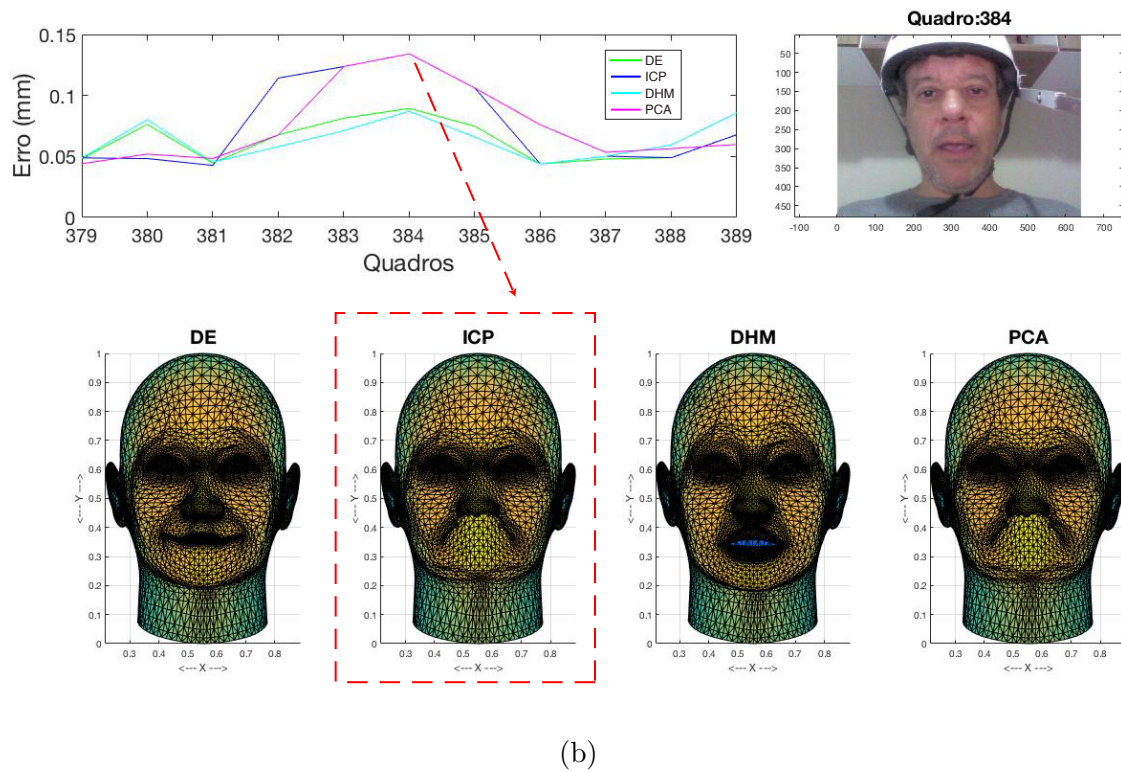
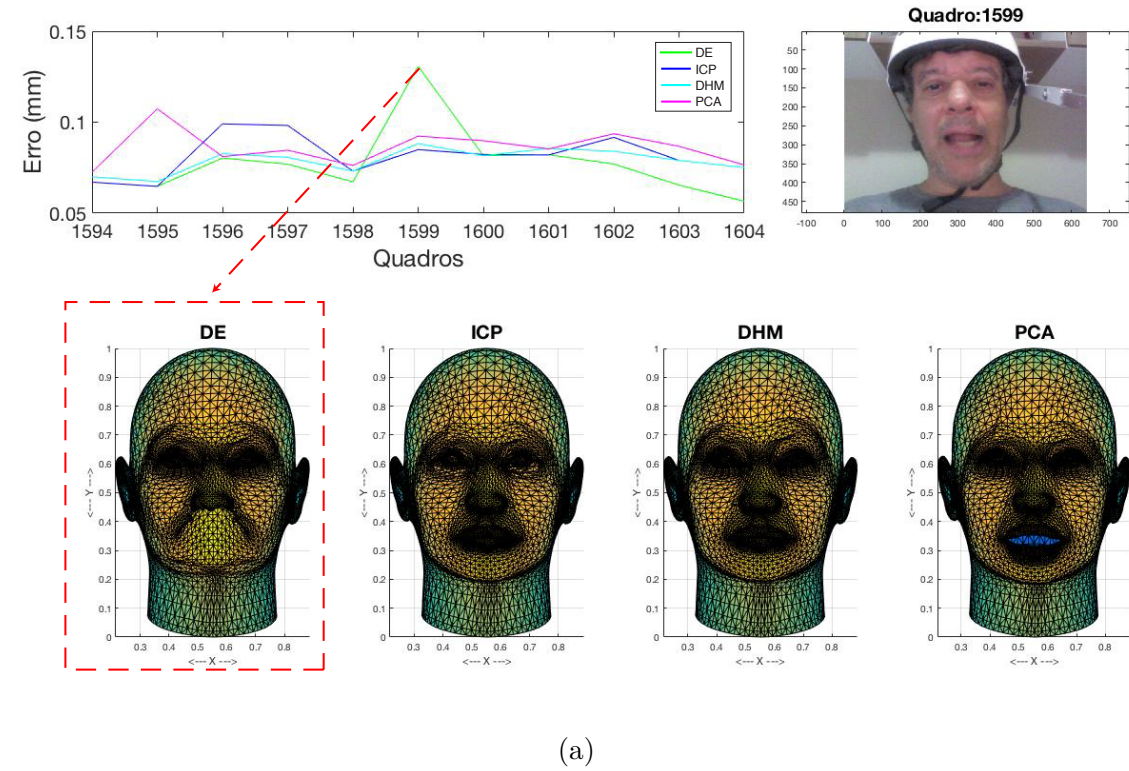
(c)

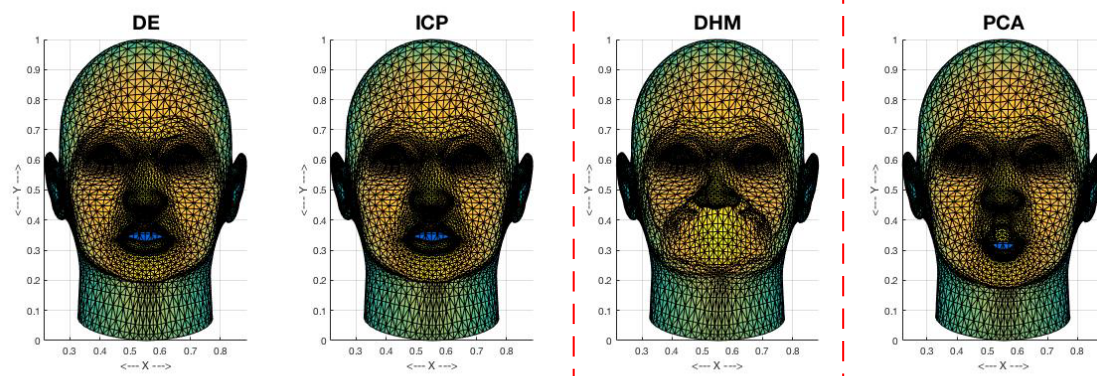
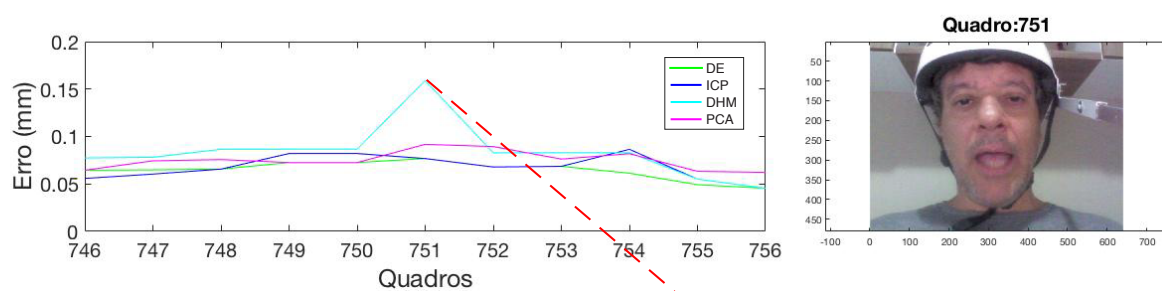


(d)

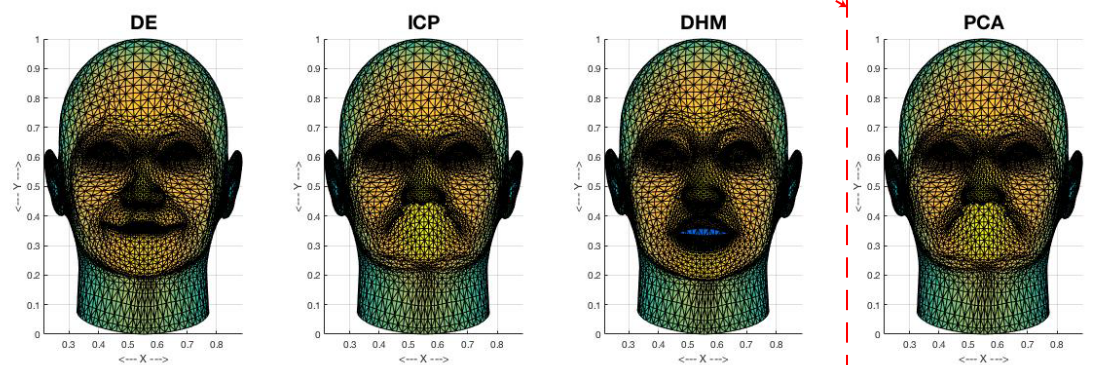
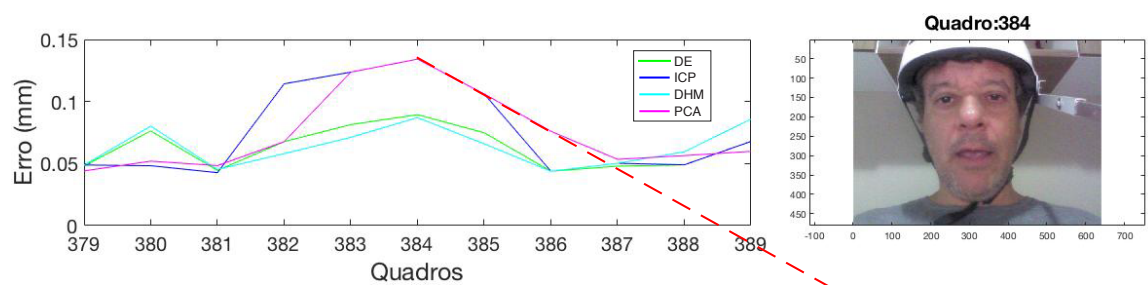
Fonte: elaborada pelo autor.

Figura 8.7 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 50% do *dataset*. (a) Distância Euclidiana; (b) ICP; (c) DHM; (d) PCA.





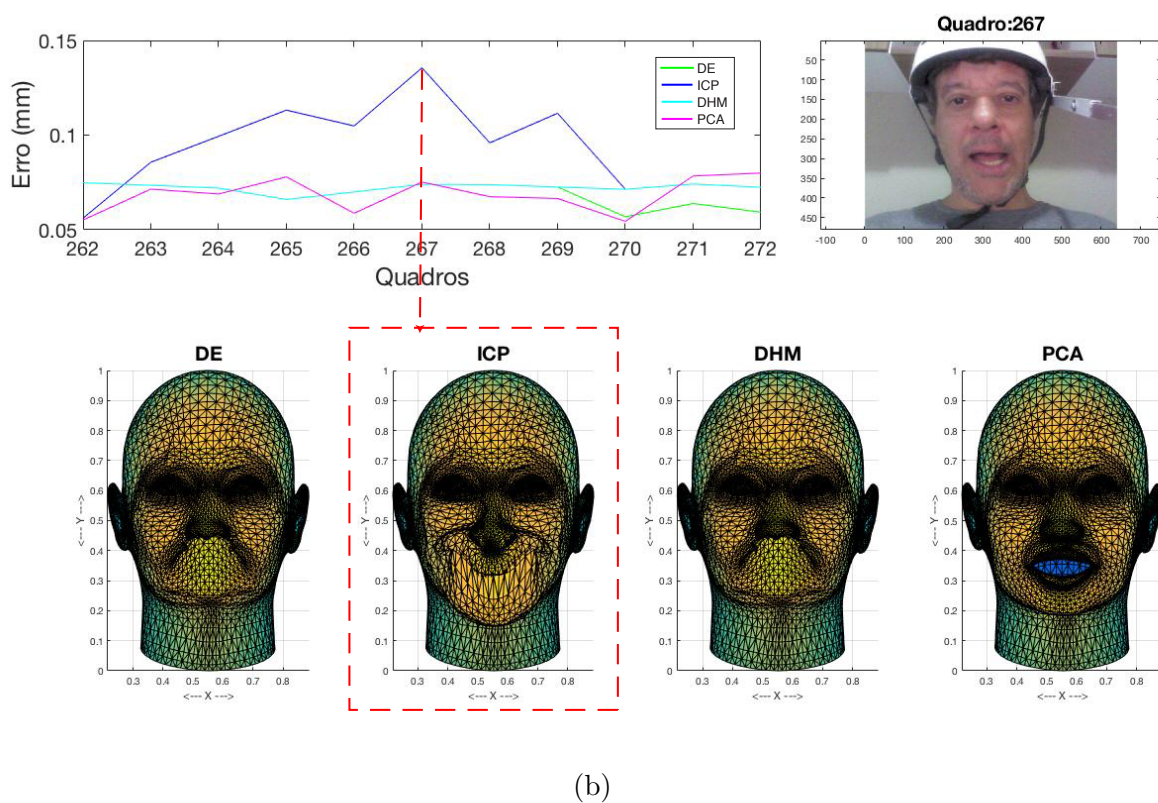
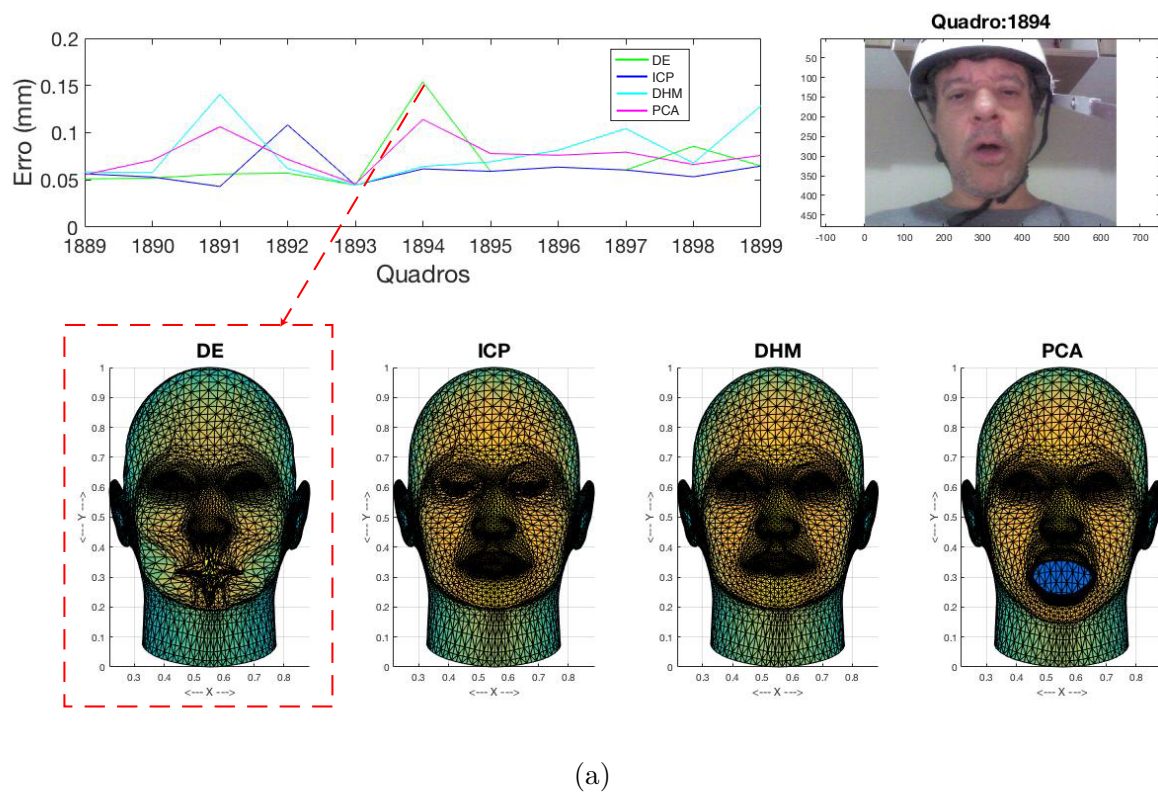
(c)

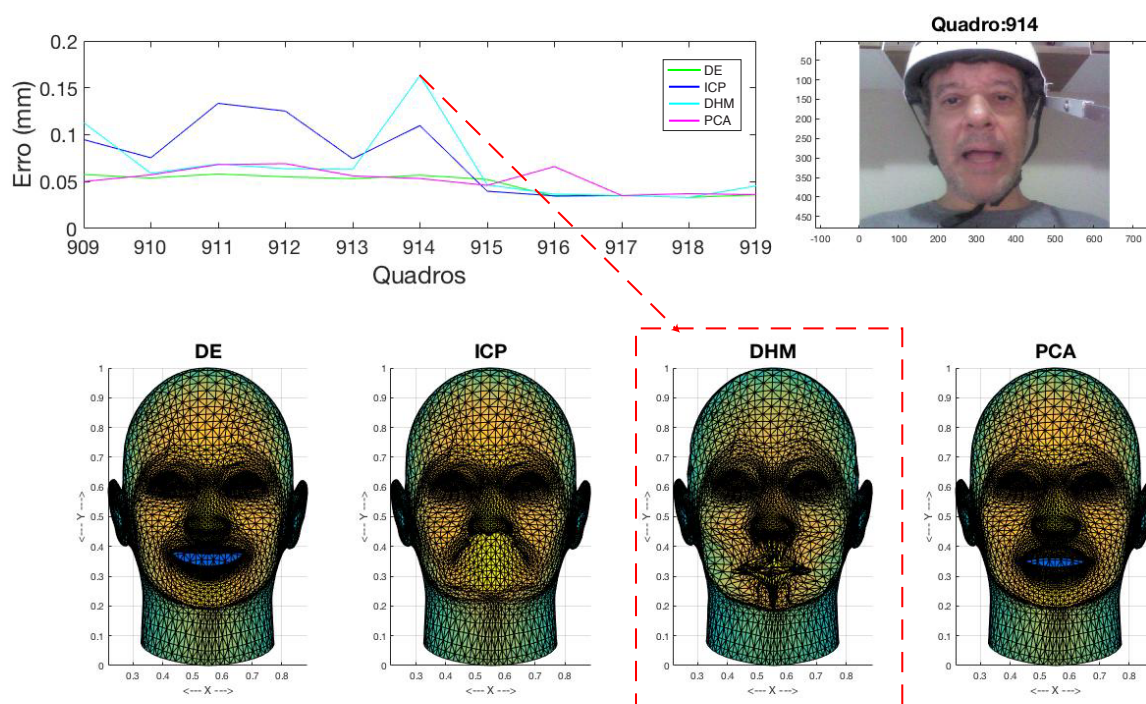


(d)

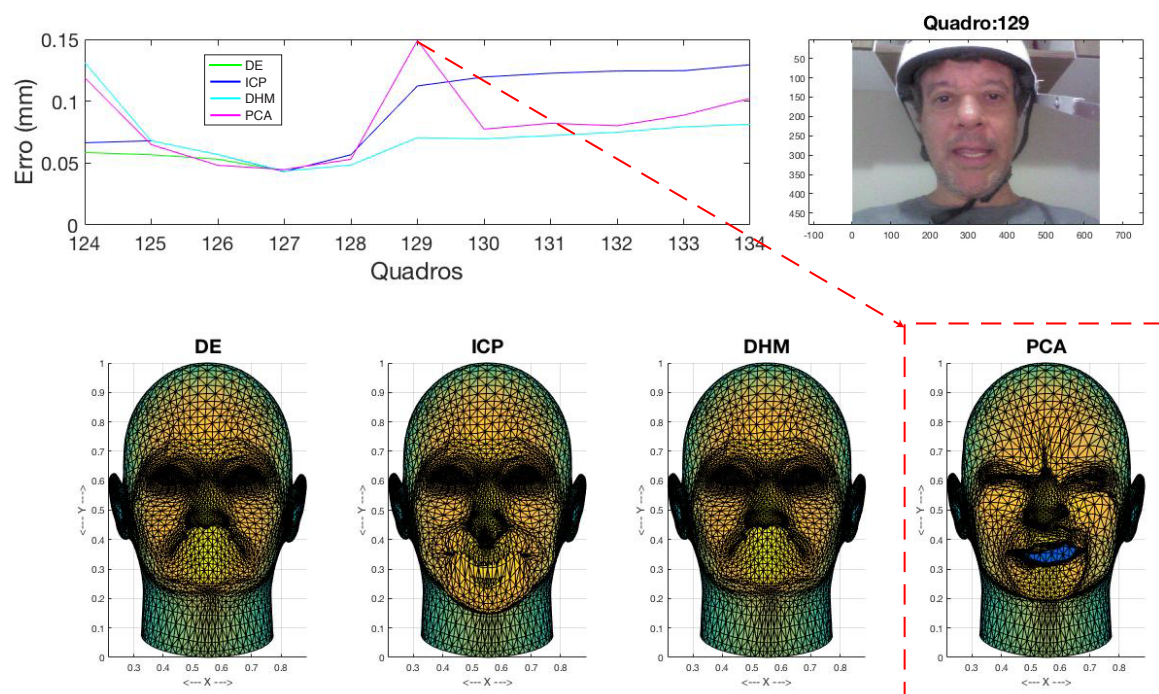
Fonte: elaborada pelo autor.

Figura 8.8 – Visualização, para cada algoritmo, do quadro com maior erro, utilizando 75% do *dataset*. (a) Distância Euclidiana; (b) ICP; (c) DHM; PCA.





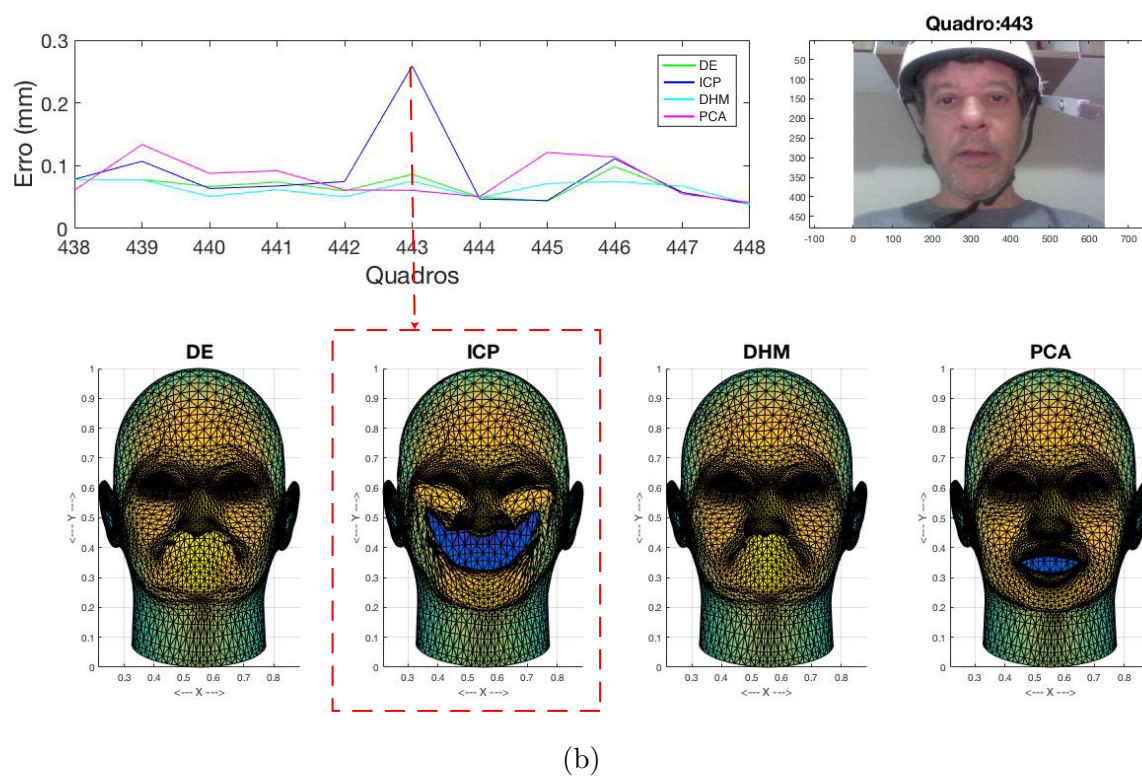
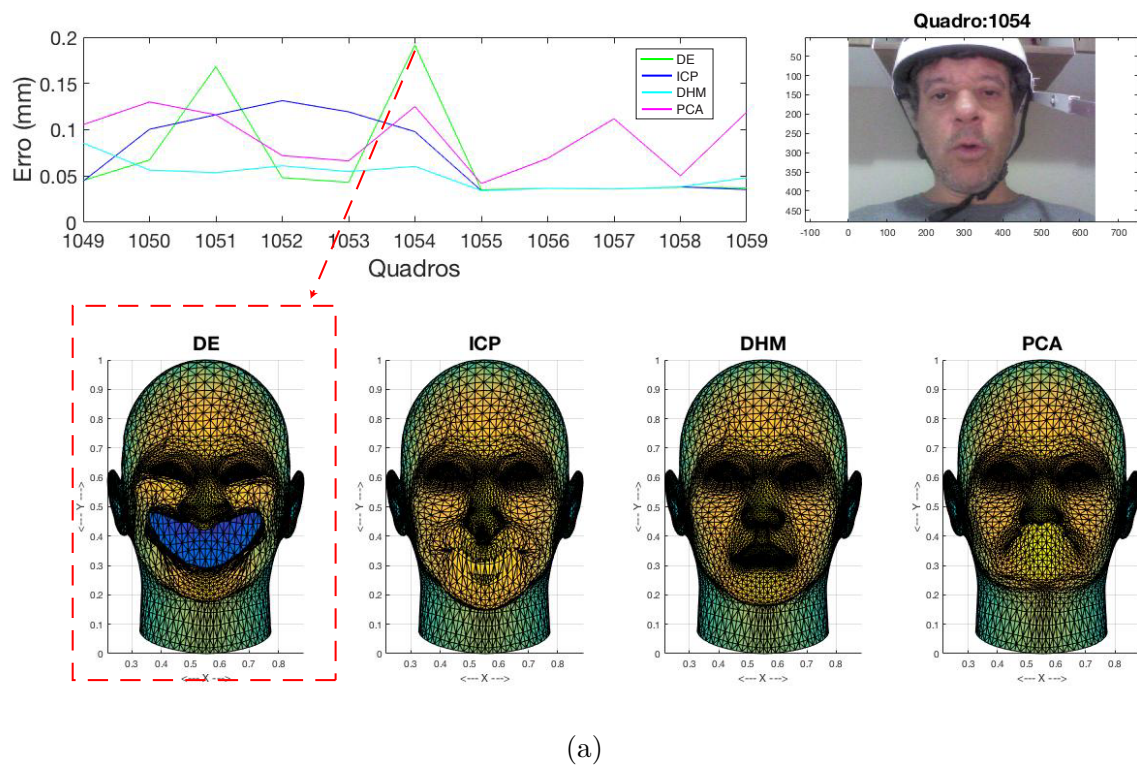
(c)

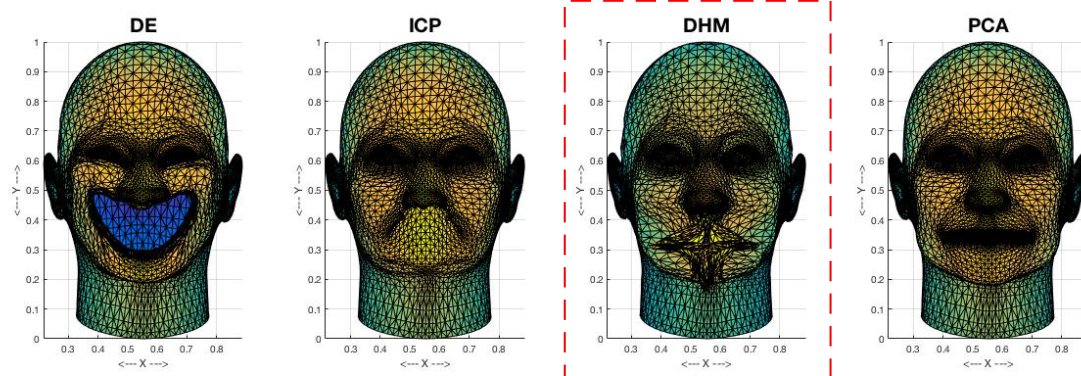
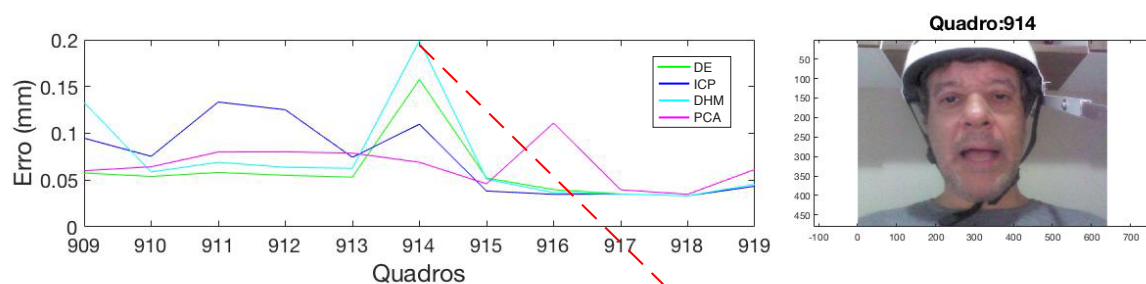


(d)

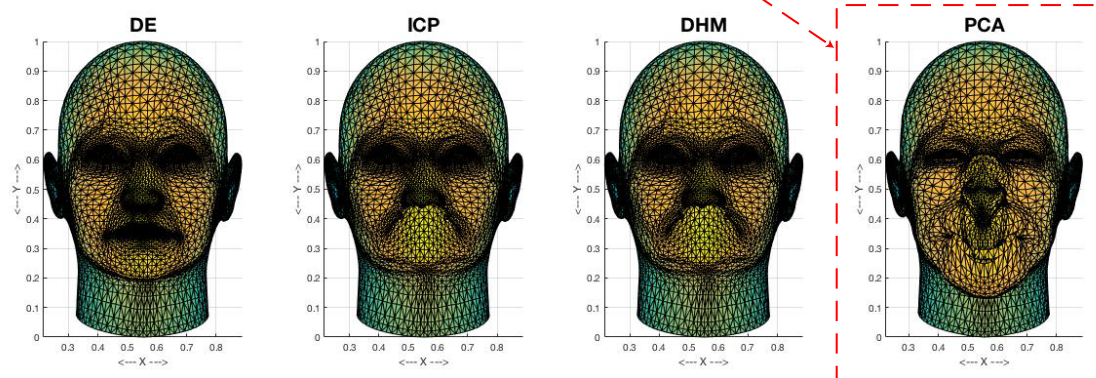
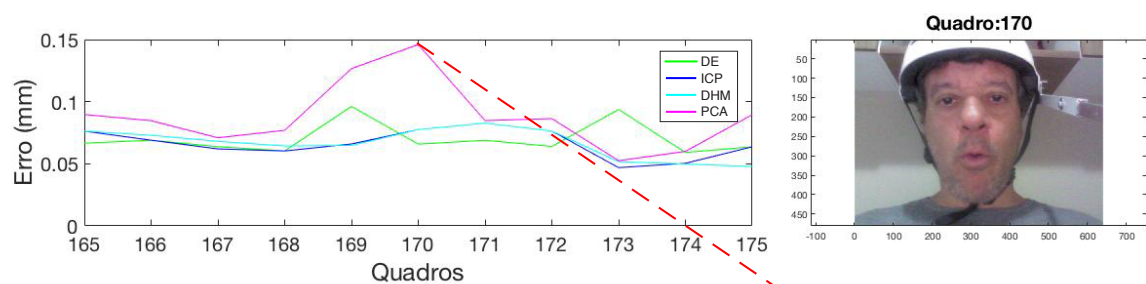
Fonte: elaborada pelo autor.

Figura 8.9 – Visualização, para cada algoritmo, do quadro com maior erro utilizando 100% do *dataset*. (a) Distância Euclidiana; (b) ICP; (c) DHM; PCA.





(c)



(d)

Fonte: elaborada pelo autor.

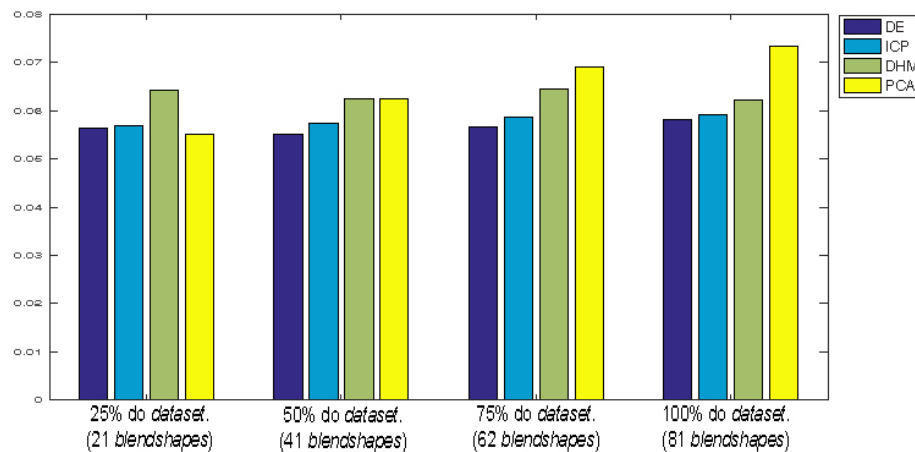
A Tabela 1 exibe as médias dos erros (em milímetros) dos quadros capturados em função da quantidade de *blendshapes* selecionados. Percebe-se que o algoritmo que teve a menor média foi a PCA com 25% dos *blendshapes*. O resultado pode ser visualizado na Figura 8.10.

Tabela 1 – Médias (em milímetros) dos erros por algoritmo de rastreamento dos conjuntos de *blendshapes*.

<i>Blendshapes</i>	D. Euclidiana	ICP	DHM	PCA
25% (21 <i>blendshapes</i>)	0.056384	0.056958	0.064223	0.055039
50% (41 <i>blendshapes</i>)	0.055113	0.057366	0.062512	0.062419
75% (62 <i>blendshapes</i>)	0.056637	0.058794	0.064641	0.069147
100% (81 <i>blendshapes</i>)	0.058285	0.059124	0.062229	0.073401

Fonte: elaborada pelo autor.

Figura 8.10 – Médias dos erros por algoritmo de rastreamento.



Fonte: elaborada pelo autor.

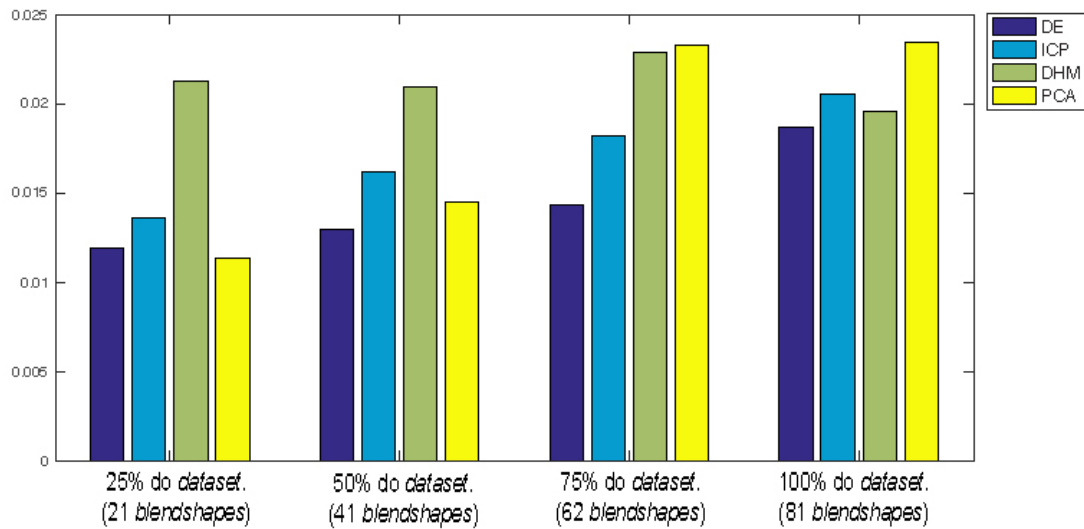
Foi calculado o desvio padrão da média para entender a variação de cada algoritmo para o conjunto de quadros analisados. O resultado pode ser visto na Tabela 2, como também na Figura 8.11.

Tabela 2 – Desvio padrão do erro por algoritmo, considerando-se a quantidade de *blendshapes*.

<i>Blendshapes</i>	D. Euclidiana	ICP	DHM	PCA
25% (21 <i>blendshapes</i>)	0.011856	0.013554	0.021274	0.01134
50% (41 <i>blendshapes</i>)	0.01296	0.016209	0.020964	0.014441
75% (62 <i>blendshapes</i>)	0.014324	0.018183	0.022837	0.02325
100% (81 <i>blendshapes</i>)	0.018636	0.020552	0.019524	0.023441

Fonte: elaborada pelo autor.

Figura 8.11 – Desvio padrão do erro por algoritmo, considerando-se a quantidade de blendshapes.



Fonte: elaborada pelo autor.

O resultado mostra que o algoritmo PCA, com 25% dos *blendshapes*, obteve o menor desvio padrão. No entanto, considerando-se o *dataset* completo, o algoritmo da Distância Euclidiana obteve o menor desvio padrão.

8.2.1.1 Resultados obtidos com relação ao tempo de processamento de cada algoritmo

Além da acurácia, mediu-se, também, o desempenho com relação à execução dos algoritmos. Os resultados podem ser visualizados nos gráficos da Figura 8.12. (a) 25 % (21 *blendshapes*); (b) 50 % (41 *blendshapes*); (c) 75 % (62 *blendshapes*) e (d) 100 % (81 *blendshapes*).

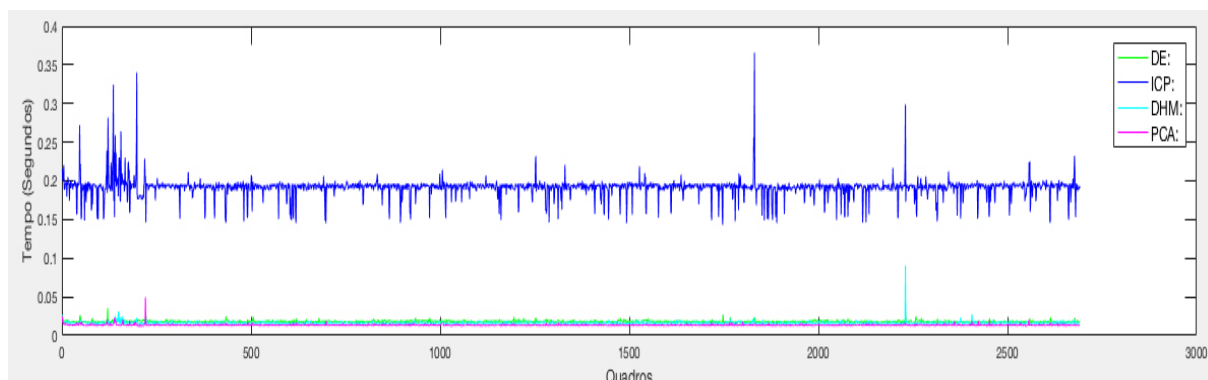
A Tabela 3 exhibe os tempos, na média, de processamento obtidos a partir da execução de todo o *pipeline*, desde a captura da face até a geração da face virtual final, para cada quadro, segundo a aplicação de cada algoritmo de rastreamento. Como pode ser observado, o algoritmo PCA, usando 25% do *dataset* teve o menor tempo de processamento para cada quadro. Os resultados também podem ser observados na Figura 8.13.

Tabela 3 – Médias dos tempos de processamento dos quadros para cada algoritmo.

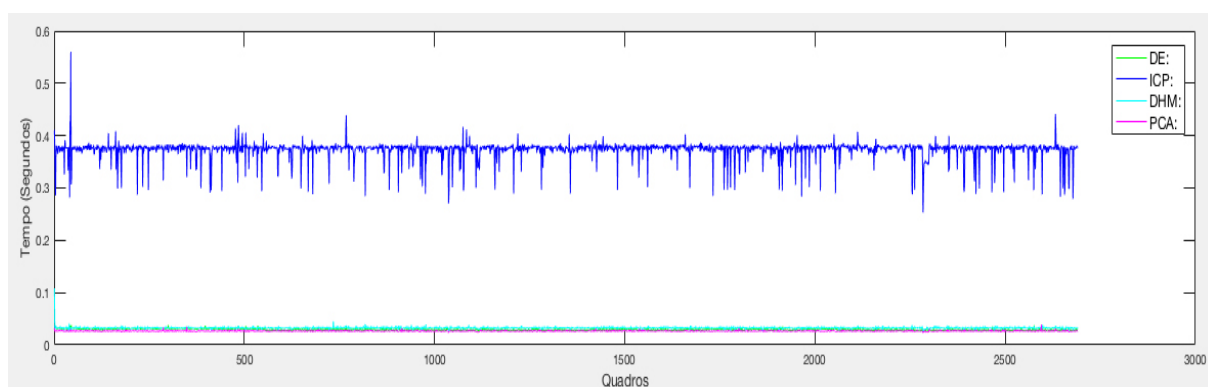
<i>Blendshapes</i>	D. Euclidiana	ICP	DHM	PCA
25% (21 <i>blendshapes</i>)	0.018016	0.19137	0.016419	0.013575
50% (41 <i>blendshapes</i>)	0.029521	0.37284	0.032577	0.026639
75% (62 <i>blendshapes</i>)	0.040968	0.51951	0.046189	0.038669
100% (81 <i>blendshapes</i>)	0.052301	0.72316	0.061883	0.050505

Fonte: elaborada pelo autor.

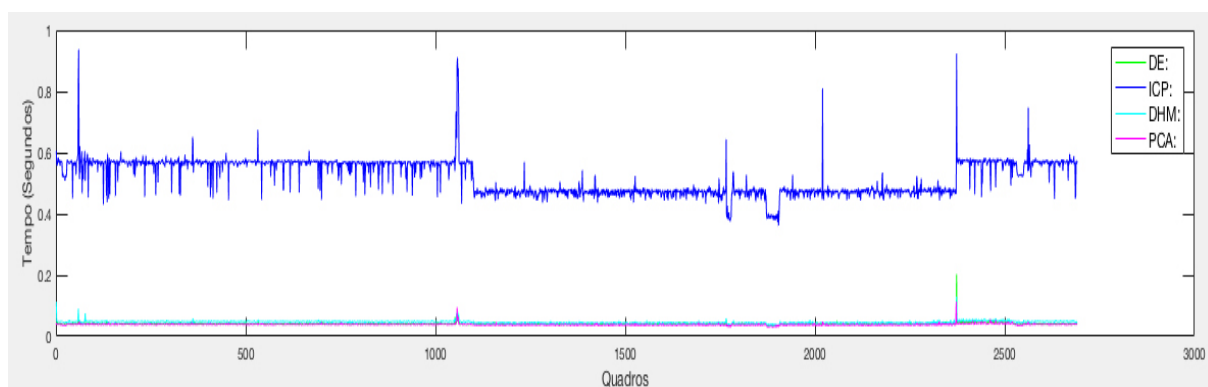
Figura 8.12 – Tempo gasto por algoritmo, quadro a quadro, para a quantidade de *blendshapes* selecionados.



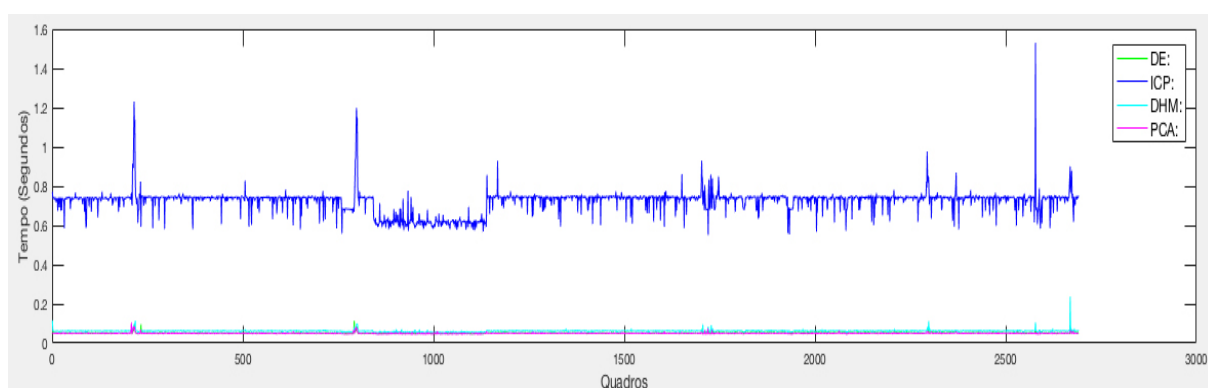
(a)



(b)



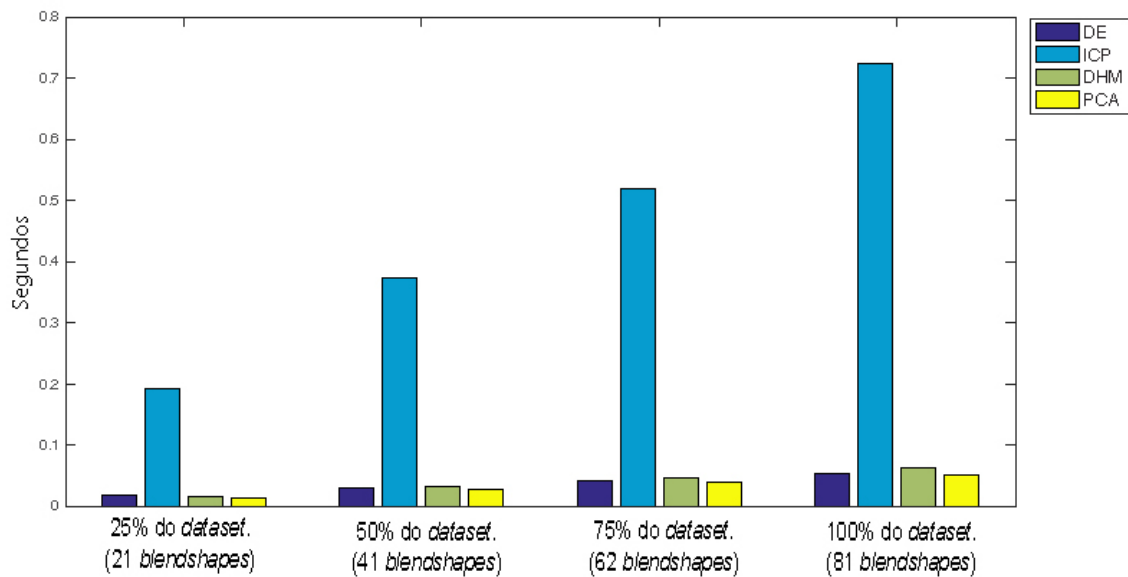
(c)



(d)

Fonte: elaborada pelo autor.

Figura 8.13 – Resultado obtido pelos algoritmos em função do tempo de processamento.



Fonte: elaborada pelo autor.

Observando o gráfico da Figura 8.13, percebe-se, como esperado, que a quantidade de *blendshapes* influencia no tempo de processamento, ou seja, quanto maior o número de *blendshapes* analisados, maior será o tempo de processamento. Nota-se, também, que segundo Microsoft (2003), em geral, o mínimo de quadros necessários para obter-se tempo real, é de 30 quadros por segundo. Nestas circunstâncias, com base nos valores obtidos, pode-se afirmar que os algoritmos que apresentaram desempenho no processamento do quadro menor que 0.3 segundos se enquadram no tempo necessário para obter o tempo real como a Distância Euclidiana, ICP, DHM e PCA usando 25% dos *blendshapes* e o algoritmo da Distância Euclidiana usando 50% dos *blendshapes*. Para *games* a quantidade de quadros por segundos, em alguns casos, chega a taxa de 60. No caso, os algoritmos que possuem um processamento de até 0.016 segundos poderiam ser usados como é o caso do PCA usando 25% dos *blendshapes*.

9 Conclusões

Uma etapa importante, no processo de animação facial baseado em performance, é a qualidade com que o redirecionamento da face capturada do ator é transferida para uma face virtual. Para este tipo de tarefa, foram criados *pipelines* específicos, mas a maioria encontra-se na indústria, não permitindo a sua reprodução pois possuem algoritmos proprietários.

Estes *pipelines* buscam a criação de faces virtuais muito próximas da face real, pois o objetivo é atingir o maior grau de realismo na animação facial. Para isto, os *pipelines*, além de usarem algoritmos como os apresentados neste trabalho também podem combiná-los em uma abordagem híbrida.

Para este processo, até pouco tempo, eram exigidos equipamentos de alto custo, o que dificultava a implementação dos *pipelines* destinados a animação baseada em performance. No entanto, a recente popularização de câmeras RGB-D de baixo custo, como o *Kinect* da Microsoft, e a *RealSense* da Intel tem viabilizado a implementação de novos *pipelines* com bom desempenho e a um baixo custo.

Com base no uso de um equipamento deste tipo, a câmera *RealSense* da Intel, apresentou-se um ambiente para animação facial baseado em performance, o qual é capaz de incorporar algoritmos de rastreamento utilizando *blendshapes*. Os resultados obtidos a partir dos testes realizados, utilizando-se quatro algoritmos de rastreamento encontrados na literatura, indicaram que o *pipeline* adotado na criação do ambiente é viável, inclusive podendo atingir um desempenho compatível com aplicações de tempo real.

Como trabalhos futuros, vislumbra-se que os seguintes desdobramentos são possíveis:

- a) Aperfeiçoamento do módulo de criação de *datasets*, incorporando os algoritmos encontrados na literatura para a transferência de formas entre as malhas, permitindo a criação de *datasets* customizados para cada ator;
- b) Aperfeiçoamento do cálculo dos pesos aplicados aos *blendshapes*, por meio de outros algoritmos de rastreamento e do uso de redes neurais;
- c) Incorporação de bases de dados de *blendshapes* disponíveis para pesquisa que sejam rotuladas de acordo com as FACs, como a base de dados *Bosphorus*(FANG et al., 2011) e a *Face Warehouse*(CAO et al., 2014);
- d) Inclusão de um módulo para renderização das faces, utilizando-se, por exemplo um motor de jogos como o Unity 3D;

- e) Utilização de outras câmeras para captura dos dados como o *Kinect* da Microsoft; e
- f) Tratamento das características da face humana que não foram consideradas neste trabalho, tais como: olhos, dentes, língua e cabelo.

Referências

- AHLBERG, J. *CANDIDE-3 - An Updated Parameterised Face*. [S.l.], 2001.
- BAKER, S.; MATTHEWS, I. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, Springer, v. 56, n. 3, p. 221–255, 2004.
- BEHRENS, S.; AL-HAMADI, A.; REDWEIK, E.; NIESE, R. Automatic realtime user performance-driven avatar animation. In: IEEE. *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. [S.l.], 2013. p. 2694–2699.
- BELLEKENS, B.; SPRUYT, V.; BERKVEN, R.; WEYN, M. A survey of rigid 3d pointcloud registration algorithms. In: CITESEER. *Fourth International Conference on Ambient Computing, Applications, Services and Technologies*. [S.l.], 2014. p. 8–13.
- BISHOP, C. M. *Pattern recognition and machine learning*. [S.l.]: springer, 2006.
- BLANZ, V.; VETTER, T. A morphable model for the synthesis of 3d faces. In: ACM PRESS/ADDISON-WESLEY PUBLISHING CO. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. [S.l.], 1999. p. 187–194.
- BRAUN, A. Aprendizado e utilização do estilo de movimento facial na animação de avatares. Pontifícia Universidade Católica do Rio Grande do Sul, 2014.
- CAO, C.; WENG, Y.; ZHOU, S.; TONG, Y.; ZHOU, K. Facewarehouse: A 3d facial expression database for visual computing. *Visualization and Computer Graphics, IEEE Transactions on*, v. 20, n. 3, p. 413–425, March 2014. ISSN 1077-2626.
- CHAI, J.-x.; XIAO, J.; HODGINS, J. Vision-based control of 3d facial animation. In: EUROGRAPHICS ASSOCIATION. *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. [S.l.], 2003. p. 193–206.
- Cornell University. *Hausdorff-Based Image Comparison*. 1994. <<http://www.cs.cornell.edu/vision/hausdorff/hausmatch.html>>. Accessed: 2016-11-06.
- DENG, Z.; NEUMANN, U. Data-driven 3d facial animation. Springer, 2008.
- DUBUISSON, M.-P.; JAIN, A. K. A modified hausdorff distance for object matching. In: IEEE. *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*. [S.l.], 1994. v. 1, p. 566–568.
- Duffy Bobby. *RealSense - Your Face as a Game Controller using Javascript*. 2014. <<https://software.intel.com/en-us/blogs/2014/12/08/realsense-your-face-as-a-controller-using-javascript>>. Accessed: 2016-11-13.
- DUTREVE, L.; MEYER, A.; BOUAKAZ, S. Feature points based facial animation retargeting. In: ACM. *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*. [S.l.], 2008. p. 197–200.
- EKMAN, P. *Facial Action Coding System: The Manual*. 2002. CD-ROM.

- EKMAN, P.; ROSENBERG, E. L. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. [S.l.]: Oxford University Press, 1997.
- FALK, R.; MINTER, D.; VERNON, C.; ARETOS, G.; MODESTO, L.; LAMORLETTE, A.; WALKER, N.; CHEUNG, T.; RENTEL-LAVIN, J.; MAX, H. Art-directed technology: Anatomy of a shrek2 sequence. In: *ACM SIGGRAPH 2004 Course Notes*. New York, NY, USA: ACM, 2004. (SIGGRAPH '04). Disponível em: <<http://doi.acm.org/10.1145/1103900.1103913>>.
- FAN, Z.; LIU, E.; XU, B. Weighted principal component analysis. In: _____. *Artificial Intelligence and Computational Intelligence: Third International Conference, AICI 2011, Taiyuan, China, September 24-25, 2011, Proceedings, Part III*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 569–574. ISBN 978-3-642-23896-3. Disponível em: <http://dx.doi.org/10.1007/978-3-642-23896-3_70>.
- FANG, T.; ZHAO, X.; OCEGUEDA, O.; SHAH, S.; KAKADIARIS, I. 3d facial expression recognition: A perspective on promises and challenges. In: *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 603–610.
- FRATARCANGELI, M. Computational models for animating 3d virtual faces. Linköping University Electronic Press, 2013.
- HONG, P.; WEN, Z.; HUANG, T. S. Real-time speech-driven face animation with expressions using neural networks. *Neural Networks, IEEE Transactions on*, IEEE, v. 13, n. 4, p. 916–927, 2002.
- HOSSAIN, M. J.; DEWAN, M. A. A.; AHN, K.; CHAE, O. A linear time algorithm of computing hausdorff distance for content-based image analysis. *Circuits, Systems, and Signal Processing*, v. 31, n. 1, p. 389–399, 2012. ISSN 1531-5878. Disponível em: <<http://dx.doi.org/10.1007/s00034-011-9284-y>>.
- LEWIS, J.; ANJYO, K.-i. Direct manipulation blendshapes. *IEEE Computer Graphics and Applications*, IEEE, v. 30, n. 4, p. 42–50, 2010.
- LEWIS, J. P.; ANJYO, K.; RHEE, T.; ZHANG, M.; PIGHIN, F. H.; DENG, Z. Practice and theory of blendshape facial models. In: CITESEER. *Eurographics (State of the Art Reports)*. [S.l.], 2014. p. 199–218.
- LI, D.; SUN, C.; HU, F.; ZANG, D.; WANG, L.; ZHANG, M. Real-time performance-driven facial animation with 3ds max and kinect. In: *Consumer Electronics, Communications and Networks (CECNet), 2013 3rd International Conference on*. [S.l.: s.n.], 2013. p. 473–476.
- LI, H.; WEISE, T.; PAULY, M. Example-based facial rigging. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2010. v. 29, n. 4, p. 32.
- LI, H.; YU, J.; YE, Y.; BREGLER, C. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.*, ACM, New York, NY, USA, v. 32, n. 4, p. 42:1–42:10, jul. 2013. ISSN 0730-0301. Disponível em: <<http://doi.acm.org/10.1145/2461912.2462019>>.
- LOW, K.-L. Linear least-squares optimization for point-to-plane icp surface registration. *Chapel Hill, University of North Carolina*, v. 4, 2004.

- LUO, C.; YU, J.; JIANG, C.; LI, R.; WANG, Z. Real-time control of 3d facial animation. In: *Multimedia and Expo (ICME), 2014 IEEE International Conference on*. [S.l.: s.n.], 2014. p. 1–6.
- MACEDO, M. C. de F.; SOUZA, A. C. dos S. Reconstrução de modelos 3-d em tempo real utilizando kinect e gpu. 2012.
- MARAR, J. F.; JUNIOR, H. F.; HOLDSCHIP, R.; SEMENTILLE, A. C.; BUSATO, S. L. Emoção & design: Quando a face revela o sucesso do produto. In: CORES, E. das Letras e (Ed.). *Metodologia em design : inter-relacoes*. [S.l.], 2011. p. 162–183.
- Microsoft. *Understanding Frames Per Second (FPS)*. 2003. <<https://support.microsoft.com/en-us/kb/269068>>. Accessed: 2017-01-15.
- MIRANDA, J. C. *Interaction Between Virtual Characters and Humans or Others Avatars in Rehabilitation Domain*. Tese (Doutorado) — Ph. D. Dissertation Proposal, The Faculty of Engineering of the University of Porto, Porto, Portugal, 2008. DOI= http://paginas.fe.up.pt/~aas/pub/Aulas/PI/Avaliacao/Ano07_08/JoseCarlosMiranda.pdf, 2008.
- MOHAMMADI, M.; FATEMIZADEH, E.; MAHOOR, M. Intensity estimation of spontaneous facial action units based on their sparsity properties. *Cybernetics, IEEE Transactions on*, PP, n. 99, p. 1–1, 2015. ISSN 2168-2267.
- NENDYA, M. B.; YUNIARNO, E. M.; GANDANG, S. Facial rigging for 3d character. *Int. J. Comput. Graph. Animat*, v. 4, n. 3, p. 21–29, 2014.
- Oliveira O. *Máximos e mínimos condicionados e multiplicadores de Lagrange*. 2012. <<https://www.ime.usp.br/~oliveira/MAT211LAGRANGE.pdf>>. Accessed: 2016-11-05.
- ORVALHO, V.; BASTOS, P.; PARKE, F.; OLIVEIRA, B.; ALVAREZ, X. A facial rigging survey. In: *in Proc. of the 33rd Annual Conference of the European Association for Computer Graphics-Eurographics*. [S.l.: s.n.], 2012. p. 10–32.
- PANDZIC, I. S.; FORCHHEIMER, R. Mpeg-4 facial animation. *The standard, implementation and applications*. Chichester, England: John Wiley&Sons, Wiley Online Library, 2002.
- PANDZIC, I. S.; FORCHHEIMER, R. (Ed.). *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. New York, NY, USA: John Wiley & Sons, Inc., 2003. ISBN 0470854626.
- PAULY, M. Realtime performance-based facial avatars for immersive gameplay. In: ACM. *Proceedings of Motion on Games*. [S.l.], 2013. p. 23–28.
- PÉREZ, P.; GANGNET, M.; BLAKE, A. Poisson image editing. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2003. v. 22, n. 3, p. 313–318.
- RAPOSO, C.; BATISTA, J. *4D Facial Dynamics*. 2012. <<http://dynface4d.isr.uc.pt/database.php>>. Accessed: 2016-11-07.
- RERIS, R.; BROOKS, P. Principal component analysis and optimization: A tutorial. *14th INFORMS Computing Society Conference*, p. 162:1–162:10, 2015. ISSN 0730-0301.

- RITCHIE, K.; CALLERY, J.; BIRI, K. *The Art of Rigging: A definitive guide to character technical direction with Alias Maya*. [S.l.]: CG Toolkit, 2005.
- ROBISON, S. *Geometria Plana*. 2014. <<http://www.infoescola.com/geometria-plana/>>. Accessed: 2016-12-05.
- ROESCH, E. B.; TAMARIT, L.; REVERET, L.; GRANDJEAN, D.; SANDER, D.; SCHERER, K. R. FacsGen: A tool to synthesize emotional facial expressions through systematic manipulation of facial action units. *Journal of Nonverbal Behavior*, Springer, v. 35, n. 1, p. 1–16, 2011.
- RUSINKIEWICZ, S.; LEVOY, M. Efficient variants of the icp algorithm. In: IEEE. *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*. [S.l.], 2001. p. 145–152.
- RUSU, R. B. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, Springer, v. 24, n. 4, p. 345–348, 2010.
- SEOL, Y.; LEWIS, J.; SEO, J.; CHOI, B.; ANJYO, K.; NOH, J. Spacetime expression cloning for blendshapes. *ACM Transactions on Graphics (TOG)*, ACM, v. 31, n. 2, p. 14, 2012.
- SEOL, Y.; SEO, J.; KIM, P. H.; LEWIS, J. P.; NOH, J. Artist friendly facial animation retargeting. *ACM Trans. Graph.*, ACM, New York, NY, USA, v. 30, n. 6, p. 162:1–162:10, dez. 2011. ISSN 0730-0301. Disponível em: <<http://doi.acm.org/10.1145/2070781.2024196>>.
- SHLENS, J. A tutorial on principal component analysis. In: *Systems Neurobiology Laboratory, Salk Institute for Biological Studies*. [S.l.: s.n.], 2005.
- STOIBER, N.; SEGUIER, R.; BRETON, G. Automatic design of a control interface for a synthetic face. In: *Proceedings of the 14th International Conference on Intelligent User Interfaces*. New York, NY, USA: ACM, 2009. (IUI '09), p. 207–216. ISBN 978-1-60558-168-2. Disponível em: <<http://doi.acm.org/10.1145/1502650.1502681>>.
- SUMNER, R. W.; POPOVIĆ, J. Deformation transfer for triangle meshes. *ACM Transactions on Graphics (TOG)*, ACM, v. 23, n. 3, p. 399–405, 2004.
- TIPPING, M. E.; BISHOP, C. M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, Wiley Online Library, v. 61, n. 3, p. 611–622, 1999.
- VELUSAMY, S.; KANNAN, H.; ANAND, B.; SHARMA, A.; NAVATHE, B. A method to infer emotions from facial action units. In: *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 2028–2031. ISSN 1520-6149.
- VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: IEEE. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. [S.l.], 2001. v. 1, p. I–511.
- WEISE, T.; BOUAZIZ, S.; LI, H.; PAULY, M. Realtime performance-based facial animation. In: ACM. *ACM Transactions on Graphics (TOG)*. [S.l.], 2011. v. 30, n. 4, p. 77.

- XU, X.; WU, Z.; WANG, X.; ZHOU, M. Retargeting 3d facial expressions in real time based on kinect. In: ACM. *Proceedings of the 13th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*. [S.l.], 2014. p. 209–213.
- YUAN, S.; LU, Y.; HE, H. Midi-based software for real-time network performances. In: IEEE. *Cryptography and Network Security, Data Mining and Knowledge Discovery, E-Commerce & Its Applications and Embedded Systems (CDEE), 2010 First ACIS International Symposium on*. [S.l.], 2010. p. 226–230.
- ZENG, M.; LIANG, L.; LIU, X.; BAO, H. Video-driven state-aware facial animation. *Computer animation and virtual worlds*, Wiley Online Library, v. 23, n. 3-4, p. 167–178, 2012.
- ZHANG, Q.; LIU, Z.; QUO, G.; TERZOPOULOS, D.; SHUM, H.-Y. Geometry-driven photorealistic facial expression synthesis. *Visualization and Computer Graphics, IEEE Transactions on*, IEEE, v. 12, n. 1, p. 48–60, 2006.
- ZHANG, Z. Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, v. 19, n. 2, p. 4–10, Feb 2012. ISSN 1070-986X.
- ZHOU, M.; LIANG, L.; SUN, J.; WANG, Y. Aam based face tracking with temporal matching and face segmentation. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. [S.l.], 2010. p. 701–708.

A Apêndice, o *dataset* completo

