TRABALHO DE GRADUAÇÃO

EZ3D: Sistema de Rastreamento Visual de Movimentos Faciais sem Marcadores para Modelos de Animação Tridimensionais

Juarez Aires Sampaio Filho

Rodrigo de Assis Ramos Lima

Brasília, dezembro de 2016

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Controle e Automação

TRABALHO DE GRADUAÇÃO

EZ3D: Sistema de Rastreamento Visual de Movimentos Faciais sem Marcadores para Modelos de Animação Tridimensionais

Juarez Aires Sampaio Filho

Rodrigo de Assis Ramos Lima

Relatório submetido como requisito parcial de obtenção de grau de Engenheiro de Controle e Automação

Banca Examinadora

Prof. Flávio de Barros Vidal, CIC/UnB _____
Orientador

Prof. Wilson Henrique Veneziano, CIC/UnB _____
Examinador interno

Prof. Dianne Magalhães Viana, ENM/UnB _____
Examinadora interna

Brasília, dezembro de 2016

FICHA CATALOGRÁFICA

SAMPAIO FILHO, JUAREZ AIRES; LIMA, RODRIGO DE ASSIS RAMOS;
EZ3D: Sistema de Rastreamento Visual de Movimentos Faciais sem Marcadores para Modelos de Animação Tridimensionais,

[Distrito Federal] 2016.

xiii, 68p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2016). Trabalho de Graduação – Universidade de Brasília.Faculdade de Tecnologia.

1. Animação Auxiliada por Computação 2.Rastreamento Visual
3. Mistura de Poses

I. Mecatrônica/FT/UnB

REFERÊNCIA BIBLIOGRÁFICA

Juarez Aires Sampaio Filho e Rodrigo de Assis Ramos Lima(2016). EZ3D: Rastreamento Visual de Movimentos Faciais sem Marcadores para Modelos de Animação Tridimensionais. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-$n°$33/2016, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 68p.

CESSÃO DE DIREITOS

AUTORES: Juarez Aires Sampaio Filho e Rodrigo de Assis Ramos Lima

TÍTULO DO TRABALHO DE GRADUAÇÃO: EZ3D: Sistema de Rastreamento Visual de Movimentos Faciais sem Marcadores para Modelos de Animação Tridimensionais.

GRAU: Engenheiro de Controle e Automação ANO: 2016

Juarez Aires Sampaio Filho
70856-150 Brasília – DF – Brasil

Rodrigo de Assis Ramos Lima
70355-090 Brasília – DF – Brasil

Dedicatórias

a Ana Valderez Ayres Neves de Alencar, porque não teria me tornado engenheiro se não tivesse começado cedo

Juarez Aires Sampaio Filho

Ao meu pai Esmaragdo e à minha mãe Gilmara.

Rodrigo de Assis Ramos Lima

Agradecimentos

Agradeço aos meus pais Juarez Aires Sampaio e Raquel Maria do Couto por fornecerem as condições para que eu dedicasse todos esses anos ao desenvolvimento das habilidades que melhor utilizam meus talentos, aos professores das Universidade de Brasília por dedicarem seu trabalho a extrair o melhor dos alunos, e aos amigos e familiares por me ajudarem a chegar são ao final desta jornada. Agradeço os modelos gráficos especialmente desenvolvidos para este trabalho ao designer Max Von Behr. Por último, mas não menos importante, agradeço a Felipe Sampaio Wense e Ataias Pereira Reis pela ajuda com a revisão do texto.

Juarez Aires Sampaio Filho

Primeiramente gostaria de agradecer a minha família, que sempre apoiou minhas decisões. Em especial aos meu pais Esmaragdo Ramos Lima e Gilmara de Assis Ramos Lima, que me deram a educação necessária para eu me tornar quem sou.
Agradeço ao meu orientador Flávio Vidal pelo suporte e dedicação. Também quero agradecer ao meu amigo Juarez, com quem escrevi este trabalho, por ser uma ótima dupla.
Agradeço aos amigos que fiz durante o meu curso na UnB, eles tornaram todos esses anos de esforço acadêmico em anos muito mais descontraídos. Agradeço também aos amigos que fiz durante meu intercâmbio, por fazerem parte dessa experiência tão importante na minha vida. Por fim agradeço aos meus amigos de longa data, por não deixar que o estresse universitário me consumisse, sempre me proporcionando com excelentes memórias.

Rodrigo de Assis Ramos Lima

RESUMO

Tarefa presente na indústria cinematográfica e de jogos digitais, a modelagem e a animação de objetos tridimensionais permite gerar sequências de imagens dificilmente filmadas de outra forma. Mesmo em produções com atores reais, as técnicas de Animação Computacional se fazem presente, seja transformando o cenário do filme, projetando sobre o ator uma textura que modifica suas feições ou adicionando à obra um personagem completamente digital. Presente em praticamente qualquer grande produção, as técnicas de animação tridimensional apresentam custos proibitivos para aplicações independentes. Visando reduzir o número de horas de trabalho criativo requerido pelo processo de animação, técnicas de Animação Auxiliada por Computação são empregadas para auxiliar os artistas gráficos. Em particular, uma das técnicas consiste em transferir movimentos e expressões de um ator para a malha tridimensional do modelo. Este trabalho propõe uma aplicação de baixo custo que utiliza rastreamento visual sem marcadores, estimação de tridimensionalidade, filtragem digital e mistura de poses para transferir movimentos faciais para uma malha tridimensional a partir de uma sequencia de imagens capturadas por um par de câmeras.

Palavras Chave: animação auxiliada por computação, rastreamento visual, mistura de poses

ABSTRACT

Present within the movie and digital games industries, tridimensional computer modeling and animation allow generation of image sequencies hardly shot in another way. Even in live- action productions, Computer Animation techniques are used during the film making process, be it transforming the stage, rendering over the actor a realistic texture or adding to the piece a complete digital character. Despite commonly used in big productions, techniques of tridimensional animation impose prohibitive costs over independent productions. Targeting to reduce the number of creative work hours required by the animation process, techniques of Computer Aided Animation are applied to auxiliate graphic designers. In particular, one of those techniques consists of transfering movements and expressions from a live actor to a tridimensional computacional mesh. This work presents a low cost application that applies Markless Visual Tracking, Tridimensionality Estimation, Digital Filtering and Misture of Poses to transfer facial movements to a model mesh from a sequence of images captured with a pair of cameras.

Keywords: computer assisted animation, visual tracking, mixture of poses

SUMÁRIO

1 Introdução 1

2 Fundamentos 4

2.1 Definições Básicas 4

2.2 Ajuste de Modelo Deformável 5

2.2.1 Ajuste de Modelo Deformável por Deslocamento Regularizado de Média de Pontos Chaves 5

2.2.2 Modelo de Distribuição de Pontos 6

2.2.3 Análise de Componentes Principais 7

2.2.4 Conceitos de Probabilidade e Estatística 8

2.2.5 Formulação Estatística para o Ajuste de Parâmetros 11

2.2.6 Otimização de Probabilidade por Deslocamento Regularizado de Média 12

2.3 Estimação de Profundidade 13

2.3.1 Modelo da Câmera 13

2.3.2 Triangulação 16

2.4 Renderização de Objetos Tridimensionais 17

2.5 Mistura de Poses 21

2.6 Estabilização do movimento 25

2.6.1 Sequências e Sistemas Lineares Invariantes ao Deslocamento 25

2.6.2 A Transformada Z 26

2.6.3 Resposta em Frequência 26

2.6.4 Equação de Diferenças 27

2.6.5 Filtros Digitas 27

2.6.6 Filtros Digitais de Resposta Finita ao Impulso 28

2.6.7 Projeto de Filtro Passa-Baixa pela Técnica de Janela 28

3 Metodologia 31

3.1 Etapas de Desenvolvimento 31

3.1.1 Posicionamento das Câmeras para Captura das Imagens de Entrada 31

3.1.2 Rastreamento de Pontos do Rosto 34

3.1.3 Estimação da Profundidade 36

3.1.4 Atualização dos Pesos de Mistura - Razão de Distância 38

3.1.5 Filtros 39
3.1.6 Mistura de Poses 41
3.2 Experimentos de Validação dos Métodos Utilizados 46
3.2.1 Rastreamento de Pontos da Face e Estimação do Tridimensional ... 47
3.2.2 Filtragem Digital 48

4 Resultados 49
4.1 Calibração das Câmeras 49
4.2 Rastreamento de Pontos da Face 50
4.3 Estimação de Tridimensionalidade 53
4.4 Filtros 55
4.5 Mistura de Poses 57
4.6 Sistema em Funcionamento 57

5 Conclusões 64
5.1 Trabalhos Futuros 65

REFERÊNCIAS BIBLIOGRÁFICAS 67

LISTA DE FIGURAS

1.1 Exemplo de utilização de captura de movimentos e expressões na indústria cinematográfica. Pontos cuidadosamente colocados no rosto do ator são utilizados para transferir expressões para o modelo tridimensional do personagem (retirado de [1]). . . 2

2.1 Exemplo de regiões de interesse que devem ser marcadas em uma imagem. 5
2.2 Imagem retirada de [2]. Exemplo de PCA com D = 2 e L = 1. 9
2.3 Distribuição Normal Bidimensional. 10
2.4 Projeção de um ponto no mundo para o plano da imagem (adaptado de [3]). 14
2.5 Mapeamento de um ponto no mundo para o plano da imagem (retirado de [3]). 14
2.6 Transformação entre as coordenadas globais e as coordenadas da câmera por rotação e translação (retirado de [3]). 16
2.7 Disposição das câmeras (adaptado de [4]). 17
2.8 Indexação do VBO (adaptado de [5]). 18
2.9 Efeito da Matriz do Modelo (adaptado de [5]). 20
2.10 Efeito da Matriz de Visualização (adaptado de [5]). 20
2.11 Efeito da Matriz de Projeção (retirado de [5]) 21
2.12 Diagrama do efeito das matrizes (adaptado de [5]). 21
2.13 Tela do programa gratuito de código aberto *Blender* (retirado de [6]). Cada um dos elementos em cinza é um osso do modelo. O termo utilizado no meio artístico é que se trata de um modelo *rigado*. 22
2.14 Três modelos a serem misturados pela técnica de *Blend Shapes* (retirado de [7]). Respectivamente da esquerda para a direita, os modelos representam a pose neutra, a expressão de felicidade e de raiva. Nota-se a diferença na forma dos lábios e nas sobrancelhas. 23
2.15 Resultado obtido ao combinar 0.5 da pose alegre, 0.3 da pose com raiva e 0.2 da pose neutra (retirado de [7]). 23
2.16 Em (a) é observada a resposta ao impulso em frequência ideal de um filtro passa-baixas. A frequência de corte projetada é $\omega_c = \frac{\pi}{3}$ rad/segundos. Em (b) são observados alguns pontos de $h(n)$. Na realidade a sequência é infinita para ambos os lados do eixo. 28

2.17 Em (a) é observada a resposta ao impulso do filtro realizável obtido cortando-se o lobo principal mais dois lobos para cada lado da resposta ideal. Note que o eixo horizontal foi transladado para permitir uma resposta realizável. Em (b) é observada a resposta em frequência obtida com esse filtro. Nota-se que o eixo vertical é mostrado em decibéis. 29
2.18 Em (a) é observada a resposta ao impulso do filtro realizável obtido cortando-se o lobo principal somente. Em (b) é observada a resposta em frequência obtida com esse filtro. 30
2.19 Filtro projetado com 3 lobos e frequência de corte $w_c = \frac{\pi}{10}$. 30

3.1 Diagrama completo para a animação de cada *frame*. As setas verticais representam o fluxo entre as técnicas dentro de cada iteração do programa. As setas horizontais indicam dados provenientes de uma etapa de calibração do sistema. 32
3.2 Fixação das câmeras. Estrutura utilizada neste trabalho para a captura das imagens de entrada. 33
3.3 Pontos retornados pelo rastreamento facial e a numeração associada a cada ponto. A imagem sobre a qual se realiza o rastreamento foi retirada de [8]. 35
3.4 Imagens extraídas da câmera 1 (a) e da câmera 2 (b) em uma mesma iteração do programa em execução. Ao comparar as duas imagens, nota-se que o usuário aparece deslocado horizontalmente em uma imagem quando comparado a outra. Os pontos de interesse foram rastreados e marcados com círculos azuis. 37
3.5 Exemplo de imagens utilizadas em uma calibração (retirado de [9]). 37
3.6 Magnitude e fase da resposta em frequência para alguns dos filtros utilizados na aplicação. 42
3.7 Pose base (neutra). 43
3.8 Exemplos de modelos usados como poses pré-definidas. 44
3.9 Exemplos de modelos usados como poses pré-definidas. 45
3.10 Montagem experimental utilizada para captura repetitiva de fotos do mesmo alvo em várias posições controladas. 47

4.1 Gráfico da variação do desvio padrão, em pixeis, em relação a distância, em centímetros, de pontos rastreados nas imagens capturadas pela câmera 1. 51
4.2 Gráfico da variação do desvio padrão, em pixeis, em relação a distância, em centímetros, de pontos rastreados nas imagens capturadas pela câmera 2. 52
4.3 Gráfico da variação do desvio padrão, em centímetros, em relação a distância, em centímetros, de pontos estimados no sistema de coordenadas do mundo. 54
4.4 Gráfico da variação média, em centímetros, em relação a distância, em centímetros, de pontos estimados no sistema de coordenadas do mundo. 54
4.5 Peso de mistura para a Pose Olho Esquerdo. 56
4.6 Peso de mistura para a Pose Boca Aberta. 56
4.7 Peso de mistura para a Pose Sorriso. 57

4.8 Exemplo de misturas geradas configurando os parâmetros de mistura manualmente. Pesos da Figura 4.12(a): Olho Esquerdo - 0.55, Boca Fechada - 0.35, Bochecha Esquerda - 0.75 e Bravo - 0.6. Pesos da Figura 4.12(b): Sobrancelha Direita - 0.95, Boca Aberta - 0.6 e Sorrindo - 0.4. Pesos da Figura 4.12(c): Olho Esquerdo - 1.0, Olho Direito - 1.0 e Boca Aberta - 0.6. Pesos da Figura 4.12(f): Sorrindo - 0.7 e Bravo - 0.6. Pesos da Figura 4.8(e): Boca Aberta - 1.0 e Bravo - 0.9. Pesos da Figura 4.8(f): Boca Fechada - 1.0, Bochecha Esquerda - 1.0, Bochecha Direita - 0.65, e Bravo - 1.0. 58
4.9 Imagens que demonstram o sistema em funcionamento. Em relação ao movimento da boca, as Figuras 4.9(d) e 4.9(f) deixam claro que há dois movimentos sendo rastreados independentemente: abrir a boca verticalmente como no ato de bocejar e abri-la horizontalmente como no ato de sorrir. 60
4.10 Imagens que demonstram o sistema em funcionamento. Compare a Figura 4.9(d) com a 4.10(d) para notar que o avatar está sorrindo com a boca fechada nesta e aberta naquela. 61
4.11 Imagens que demonstram o sistema em funcionamento. Novamente, o avatar pode sorrir com a boca fechada ou com a boca aberta, podendo o olho estar fechado e aberto. As poses são combinadas independentemente. 62
4.12 Imagens que demonstram o sistema em funcionamento *frame* a *frame*. A sequência mostra a abertura gradual da boca enquanto o personagem conversa com as câmeras. 63

LISTA DE TABELAS

3.1 Pares de pontos utilizados para razão de distâncias. O peso w_k está associado com o par (p_{i_k}, p_{j_k}). Os índices dos pontos podem ser comparados com os índices da Figura 3.3 40
3.2 Mapeamento do índice da pose no vetor de pesos com ação realizada pela aplicação da pose. 40
3.3 Coeficientes para os filtros projetados com técnica de janela, utilizando a janela de Hamming. 41

4.1 Parâmetros intrínsecos da câmera medidos em pixeis. 49
4.2 Comprimento horizontal dos lábios medido em pixeis a partir de imagens capturadas do rosto alvo pela câmera 1 em distâncias variáveis. 50
4.3 Comprimento horizontal dos lábios medido em pixeis a partir de imagens capturadas do rosto alvo pela câmera 2 em distâncias variáveis. 51
4.4 Comprimento horizontal dos lábios medido em centímetros a partir de imagens capturadas do rosto alvo em distâncias variáveis. 55
4.5 Valores de distância mínima e distância máxima definidos, para cada pose, após calibração. 59

Lista de Símbolos

API *Application program interface*

FIR *Finite Impulse Response*

LTI *Linear time-invariant*

MP *Mistura de Poses*

OpenCV *Open Source Computer Vision*

OpenGL *Open Graphics Library*

PCA *Principal Components Analysis*

PDM *Point Distribution Model*

RLMS *Regularized Landmark Mean-Shift*

SDK *Software Development Kit*

VAO *Vertex Array Object*

VBO *Vertex Buffer Object*

Capítulo 1

Introdução

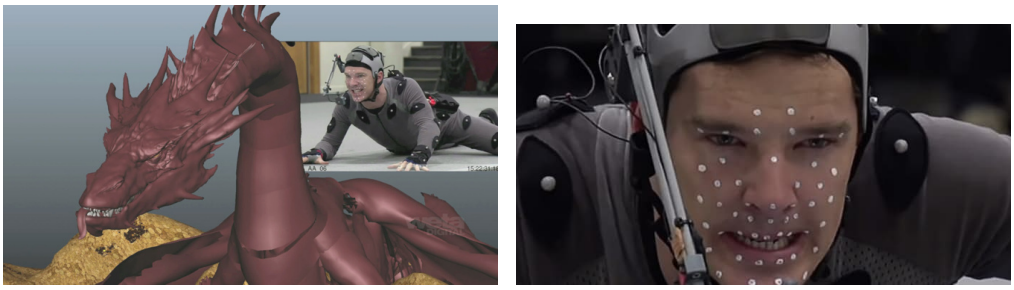
O mercado global de animação e jogos foi avaliado em $122.20 bilhões em 2010 e é esperado que esta cifra atinja $242.93 bilhões em 2016. Esse mercado global pode ser dividido em mercados específicos voltados para a Educação, o Desenvolvimento para Web e a Animação para Entretenimento, podendo esse último ser ainda subdividido em várias categorias [10], sendo **Filmes** e **Efeitos Visuais** categorias de especial interesse para este trabalho.

Uma tarefa recorrente dentro da indústria cinematográfica é a modelagem e animação de objetos tridimensionais. Tais objetos não estão sujeitos às restrições do mundo físico e permitem gerar cenas e movimentos dificilmente filmados de outra forma. Mesmo em produções com atores reais as técnicas de Animação Computacional se fazem presentes, seja transformando o cenário do filme, projetando sobre o ator uma textura que modifica suas feições ou adicionando à obra um personagem completamente digital. Presentes em praticamente qualquer novo *blockbuster*, as técnicas de animação tridimensional ainda apresentam custos elevados e proibitivos para certas aplicações.

O orçamento para a produção de um filme inclui custos de pré-produção, filmagem, pós-produção e divulgação, devendo-se levar em conta os direitos pelo roteiro, salários dos atores, salários da equipe de produção, construção do set de filmagens, efeitos especiais, figurino e tudo o mais [11]. Apesar da Animação Computacional diminuir o custo de alguns dos componentes citados, ela é em si uma técnica cara. Dessa forma, ao passo que a animação computacional reduz custos com montagem de cenários e até mesmo maquiagem, muitas vezes um único quadro de um filme pode requerer a animação de milhões de partes móveis. No filme *Monstros SA (2001)*, por exemplo, foram utilizados mais de 2 milhões de fios de cabelo individualmente nomeados para a construção do personagem Sully. Um único quadro com o personagem custou em média de 11 a 12 horas de trabalho criativo [12]. Com tantas horas de trabalho criativo gastas durante o processo de animação, é evidente a necessidade do desenvolvimento de técnicas que auxiliem os animadores.

Uma técnica importante que vem ao auxílio dos animadores consiste em transferir movimentos e expressões de um ator para um modelo computacional. Metodologias para captura ótica de performance teatral capazes de registrar expressões de um ator em geometria detalhada são utilizadas na indústria já há algum tempo. Exemplo conhecido do uso dessa tecnologia é o personagem *Gol-*

lum no filme *O Senhor dos Anéis: A Sociedade do Anel* (2001), onde o ator Andy Serkis utiliza uma roupa especial com marcadores visuais durante as gravações para que mais tarde a equipe de efeitos visuais renderize sobre ele um modelo computacional tridimensional da criatura amaldiçoada. Mais recentemente, a mesma tecnologia - em estado aprimorado e muito mais rica em detalhes - é utilizada em *O hobbit* (2015), onde o ator Benedict Cumberbatch dá vida ao dragão Smaug. A Figura 1.1 ilustra o processo. Na Figura 1.1(a) observa-se o modelo que está sendo animado pela performance do ator. Na Figura 1.1(b) mostra-se em detalhes pontos colocados no rosto do ator para auxiliar a captura de expressões.



(a) (b)

Figura 1.1: Exemplo de utilização de captura de movimentos e expressões na indústria cinematográfica. Pontos cuidadosamente colocados no rosto do ator são utilizados para transferir expressões para o modelo tridimensional do personagem (retirado de [1]).

Nesse tipo de aplicação cinematográfica requer-se do sistema de animação computacional uma alta precisão no ajuste do modelo computacional ao ator em cena. Para esse fim, utilizam-se ambientes especiais de filmagem , iluminação controlada e marcadores visuais, sendo que muitas vezes a captura da imagem é feita por um arranjo de câmeras ou ainda por câmeras de alta resolução. Além disso, o resultado do processamento recebe um ajuste fino realizado por vários artistas para que o resultado apresentado seja o mais convincente possível. Vale notar que não é possível realizar esse ajuste fino caso não haja um longo prazo disponível entre a captura da imagem e o instante em que os resultados precisam ser apresentados ao público.

Uma outra utilização das técnicas de animação auxiliada por captura de vídeo é a produção de shows com fantoches digitais. Nesse espetáculo do entretenimento, visto em programas televisivos e em parques de diversão ao redor do mundo, uma equipe em uma câmara escondida anima em tempo real um modelo projetado em tela e responde ao vivo às perguntas do público. O personagem animado apresenta expressões corporais e faciais que encantam o público, deixando os mais novos convencidos de que conversaram com seu personagem preferido e os mais velhos se perguntando como aquilo pode ser possível. Vale observar que nesse caso os requisitos de tempo são muito mais severos, uma vez que o resultado da animação computacional deve ser apresentado rapidamente ao usuário para que a interação entre público e personagem digital aconteça de forma dinâmica. Com isso em mente, não é possível o ajuste fino de uma equipe de artistas, mas ainda sim é possível fazer uso de iluminação controlada, câmera(s) de alta qualidade, marcadores visuais. A equipe por trás do personagem pode ainda fazer uso de controles pré-programados.

Apesar da tecnologia existente apresentar resultados adequados para as aplicações citadas, ela

apresenta um custo elevado. Dessa forma, ainda que grandes empresas se disponham a bancar o preço da tecnologia atual, os custos com software, hardware e com a equipe técnica envolvida podem dificultar a utilização de técnicas de animação computacional em produtos desenvolvidos por empresas de recursos mais modestos. Portanto, o desenvolvimento de produtos que realizem as mesmas tarefas a custos mais baixos é certamente de interesse do mercado de animação. Tal produto poderia, por exemplo, ser utilizado por desenvolvedores de animações independentes para acelerar e baratear seus projetos.

Um exemplo de animações de baixo orçamento pode ser encontrado na produção de programas televisivos infantis cujo objetivo seja trabalhar uma temática de interesse social ou atingir um público que não o público geral. Por exemplo, um nicho de crianças comumente não atendido pelas opções de programas infantis é o de crianças autistas. Apesar das dificuldades encontradas por essas crianças variarem muito de um indivíduo para o outro, é comum que crianças autistas apresentem dificuldade em manter contato visual, mesmo que seja com personagens de um filme. Tal fato não é levado em consideração por animações convencionais, o que as torna inadequadas para serem assistidas por crianças com a dificuldade citada. Tendo em vista que animações infantis podem ser utilizadas como uma importante ferramenta educacional, há certamente motivação para o desenvolvimento de desenhos infantis próprios para as crianças autistas.

Uma empresa interessada em desenvolver tais animações muito provavelmente tem o objetivo de atender a um interesse social acima do objetivo de atender fins lucrativos. Consequentemente, a equipe dificilmente terá os mesmos recursos financeiros de uma que trabalhe com o público geral e poderá enfrentar dificuldades em arcar com os custos envolvidos. Objetiva-se com esse trabalho que o produto desenvolvido possa ser utilizado como ferramenta para auxiliar o desenvolvimento de programas infantis com foco em crianças autistas.

Colocando de forma clara, o objetivo desse trabalho é desenvolver um sistema de animação auxiliada por captura de vídeo que seja capaz de transferir expressões faciais do usuário para uma avatar computacional. Além disso, o sistema não deve requirir nenhuma estrutura elaborada: ele deve rodar em hardware comum e ser robusto à iluminação, ruídos presentes no processo de captura de imagem e não deve utilizar marcadores visuais. A intenção é que o sistema funcione em uma máquina de configurações acessíveis e utilize câmeras de qualidade comparável à *webcams* comumente encontradas em computadores pessoais.

O Capítulo 2 deste trabalho trata da fundamentação teórica por trás das técnicas utilizadas. Reconhecimento de pontos do rosto, renderização de objetos tridimensionais, filtragem digital e estimação de profundidade são tópicos abordados. No Capítulo 3 as técnicas introduzidas são relacionados para compor a metodologia do sistema proposto neste trabalho. No Capítulo 4 os resultados obtidos são apresentados e discutidos. O último capítulo elabora conclusões e aponta propostas de melhorias futuras para o sistema de auxílio a animação computacional desenvolvido.

Capítulo 2

Fundamentos

Esta seção introduz os conceitos por trás das técnicas utilizadas no sistema de animação computacional sem maracadores proposto neste trabalho. O objetivo deste capítulo é introduzir o necessário da teoria das técnicas utilizadas, para que se entenda as propriedades dosresultados obtidos. A aplicação consiste do encadeamento de diversas técnicas oriundas dos domínios de Visão Computacional, Geometria Computacional e Filtragem Digital. Como frequentemente acontece, as técnicas de Visão Computacional muitas vezes empregam conceitos de Aprendizado de Máquina.

2.1 Definições Básicas

- Imagem

  Uma imagem pode ser definida como uma função bidimensional, $f(x, y)$, onde x e y são coordenadas espaciais e a amplitude de f em qualquer par de coordenadas (x, y) é dita a intensidade ou intensidade de cinza da imagem naquele ponto. Quando x, y e o os valores de intensidade de f são todos finitos e discretos, diz-se que a imagem é uma uma imagem digital. Uma imagem digital é composta de um número finito de elementos, cada um destes possuindo um valor e uma localização particular. Estes elementos são mais comumente chamados de pixeis [13].

- Sequência de Imagens

  Uma sequência de imagens é uma função tridimensional $\hbar(x, y, t)$ que possui uma ou mais imagens $f(x, y)$ tomadas em instantes de tempo discretos t [14].

- Rastreamento

  Rastreamento é o problema de inferir o movimento de um objeto dada uma sequência de imagens. Em um problema típico de rastreamento, tem-se um modelo para o movimento do objeto e um conjunto de medidas oriundas de uma sequência de imagem. Não é garantido que essas medidas sejam relevantes, podendo elas conter informação de outros objetos que não o objeto de interesse [15].

2.2 Ajuste de Modelo Deformável

Ajuste de Modelo Deformável é o problema de ajustar os parâmetros de um modelo paramétrico a uma imagem de forma que os pontos chaves correspondam com localizações do objeto de interesse. É uma tarefa difícil uma vez que envolve uma otimização em um espaço de alta dimensão, no qual a forma de um objeto pode variar drasticamente entre instâncias do objeto devido a condições de iluminação, ruído na imagem, resolução e fontes intrínsecas de variabilidade [16].

Ainda segundo [16], uma das abordagens mais proeminentes para o problema consiste em modelar um objeto utilizando observações espaço-temporalmente coerentes de imagens locais (*image patches*) centradas nos pontos de interesse dentro do objeto. Nesta abordagem, assume-se que os *patches* são condicionalmente independentes uns dos outros para fins computacionais e de generalização da técnica. Os detectores locais são tipicamente aprendidos, a partir de imagens marcadas de treinamento, para cada ponto proeminente do objeto.

A Figura 2.1 mostra pontos de interesse que podem ser utilizados em um modelo deformável para a face humana.

Devido ao pequeno suporte destes detectores e a alta variabilidade de aparência nos dados de treinamento, estes detectores locais estão fadados à ambiguidade [16].

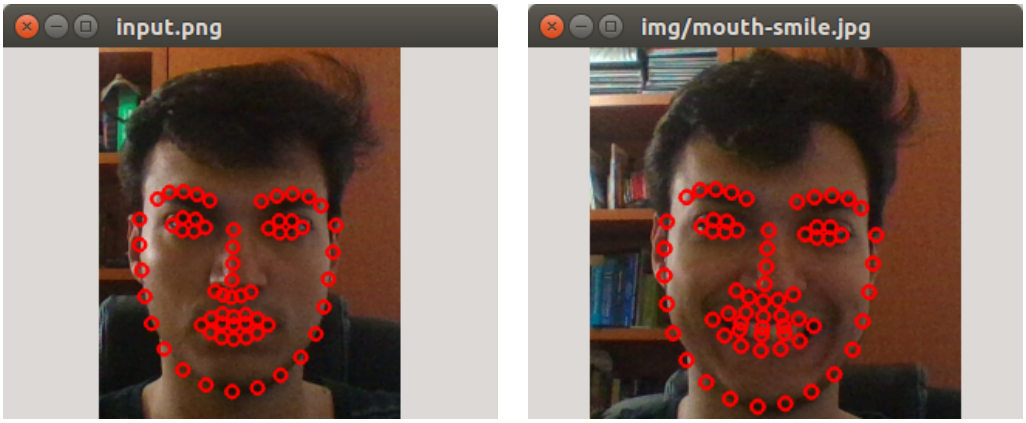


Figura 2.1: Exemplo de regiões de interesse que devem ser marcadas em uma imagem.

2.2.1 Ajuste de Modelo Deformável por Deslocamento Regularizado de Média de Pontos Chaves

O ajuste de modelo deformável por Deslocamento Regularizado de Média de Pontos Chaves, do inglês *Regularized Landmark Mean-Shift* (RLMS) é uma técnica que trata o problema de ajuste de modelo deformável empregando sinergicamente as informações de cada detector local não paramétrico dos pontos proeminentes [1] , enquanto limita o efeito de suas ambiguidades ao ajustar os parâmetros do modelo deformável.

[1] Ao longo do texto os termos 'pontos chave', 'pontos proeminentes', 'pontos de interesse' e 'marcadores' são sinônimos e referem-se ao conjunto de pontos rastreados do modelo deformável.

Como é visto a seguir, o modelo deformável paramétrico consiste em um conjunto de pontos chaves do objeto de interesse. Para cada um desses pontos chaves, o RMLS utiliza um detector treinado independentemente dos demais. Treiná-los assim é mais fácil, mas os detectores aprendidos são ruidosos e muitas vezes produzem resultados ambíguos. O RMLS é uma técnica que permite utilizar as informações de todos os outros rastreadores para detectar a posição de um ponto proeminente e não só o rastreador que foi treinado para aquele ponto em específico. Em um problema de detecção facial, por exemplo, pode-se pensar que se o rastreador para nariz está em dúvida sobre qual de dois pontos é, mais certamente, o centro de um nariz, ele pode utilizar informação dos rastreadores dos olhos e da boca para escolher qual dos dois pontos melhor representa o nariz.

O exemplo anterior deve ser entendido como uma motivação para a utilização sinérgica de rastreadores e não mais do que isso. O RLMS é uma técnica de propósito geral e não assume nenhuma forma para o modelo ou mesmo requer que a forma como os pontos se arranjam seja informada explicitamente como entrada. O RLMS aprende conformações prováveis para o conjunto de pontos a partir de dados de treinamento e poderia ser utilizado tanto para detectar pontos da face humana como pontos no contorno de um carro. A única diferença seriam os dados de entrada e, consequentemente, as matrizes aprendidas no processo.

Quando requerido para ajustar o modelo paramétrico a uma imagem de entrada, pode-se dizer simplificadamente que o RMLS tentará buscar na imagem de entrada uma combinação das configurações com as quais ele já foi familiarizado. Como em muitos outros métodos de Aprendizagem de Máquina, a busca pelos parâmetros que melhor se adequam é realizada por meio de um processo de otimização de uma equação de probabilidade. Como parte do objetivo desta fundamentação teórica, alguns detalhes do RLMS serão apresentados.

2.2.2 Modelo de Distribuição de Pontos

Essa seção especifica o modelo paramétrico utilizado pelo RLMS. O Modelo de Distribuição de Pontos, do inglês *Point Distribution Model* (PDM), propõe que o objeto a ser entendido deve ser caracterizado como um conjunto de pontos chaves bidimensionais no domínio da imagem. O PDM modela linearmente variações de forma não-rígidas e as compõe com uma transformação rígida global, colocando o i-ésimo ponto de interesse $\boldsymbol{v_i}$ em:

$$\boldsymbol{v_i} = s\boldsymbol{R}(\boldsymbol{v}_{i,0} + \boldsymbol{\Phi}_i\boldsymbol{q})\boldsymbol{t} \tag{2.1}$$

onde $\mathbf{p} = \{s, \boldsymbol{R}, \boldsymbol{t}, \boldsymbol{q}\}$ denota os parâmetros do modelo e consiste do fator de escala s, da matriz de rotação $\boldsymbol{R}$, do vetor de translação $\boldsymbol{t}$ e de um conjunto de parâmetros não rígidos $\boldsymbol{q}$. É esse conjunto $\mathbf{p}$ que deve ser calculado ao ajustar o modelo a uma imagem dada, todos os outros termos da equação podem ser previamente calculados em um processo conhecido como treinamento. Na Equação 2.1 $\boldsymbol{v_{i,0}}$ denota a posição neutra do i-ésimo ponto de interesse e $\boldsymbol{\Phi_i}$ é uma submatriz da base de variações previamente aprendida pertinente ao i-ésimo ponto de interesse. Nesta equação o termo $\boldsymbol{\Phi}_i\boldsymbol{q}$ corresponde à transformação não rígida e os outros termos à transformação rígida.

Nota-se que o modelo consiste de **uma transformação rígida aplicada a uma transformação não rígida**. Parte da riqueza do modelo está na transformação não rígida, que é apresentada mais a fundo a seguir.

O vetor parâmetros não rígidos $\boldsymbol{q} \in \mathbb{R}^d$ indica o peso com o qual cada uma das d possíveis transformações não-rígidas influencia o posicionamento do ponto chave em questão no quadro considerado. O número d deve ser fixado e é referenciado na literatura como a dimensão do PDM. A matriz $\mathbf{\Phi}$ contém as d deformações possíveis para cada um dos K pontos de interesse e a matriz Φ_i é a submatriz que contém as deformações possíveis para o i-ésimo ponto de interesse.

As constantes $v_{i,0}$ e Φ_i devem ser aprendidas a partir de um conjunto de dados de treinamento. O conjunto de dados de treinamento consiste em uma série de imagens anotadas onde todos os K pontos do PDM foram **manualmente** marcados. A partir dessas imagens é possível conhecer as principais maneiras com que o modelo de pontos aparece deformado na imagem. A técnica com a qual aprende-se a matriz $\mathbf{\Phi}$ é a Análise de Componentes Principais.

Finalmente, nota-se que a Equação 2.1 define $\mathbf{v}_i$ como função de $\mathbf{p}$. A seguinte expansão em série de Taylor é utilizada para aproximar os valores $\mathbf{v}_i$ em torno de um ponto $\mathbf{v}_i^c$:

$$\mathbf{v}_i \approx \mathbf{v}_i^c + \mathbf{J}_i \Delta \mathbf{p} \tag{2.2}$$

2.2.3 Análise de Componentes Principais

A técnica de Análise de Componentes Principais, do inglês *Principal Components Analysis* (PCA), é comumente utilizada para reduzir a dimensionalidade de um conjunto de vetores de características. Um vetor de características é simplesmente um vetor cujas componentes representam características de um objeto. A entrada para a técnica consiste de um conjunto $\mathcal{D}$ de N vetores D-dimensionais e a saída é um outro conjunto D_{PCA} de N vetores L-dimensionais. A técnica reduz a dimensionalidade dos dados quando $L < Q$. A PCA permite isso ao reescrever cada elemento $\boldsymbol{v}$ de D como uma combinação linear de L vetores D dimensionais $\{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_L\}$. Os pesos $\{z_i\}_{i=1}^{L}$da combinação se tornam as L componentes do vetor $\mathbf{z} \in \mathcal{D}_{PCA}$. Se a transformação for perfeita, é possível recuperar unicamente o vetor $\mathbf{v}$ apartir de $\mathbf{z}$, mas esse não é o caso comum. Em geral a diminuição de dimensão implica em perda de dados e é possível medir o erro da aproximação.

Seja $\mathbf{z} = (z_1, z_2, \cdots, z_L)$ um elemento de D_{PCA}, uma estimativa $\hat{\mathbf{v}}$ do vetor $\mathbf{v}$ de D que gerou $\mathbf{z}$ por meio da transformação é

$$\hat{\mathbf{v}} = z_1\mathbf{w}_1 + z_2\mathbf{w}_2 + \cdots + v_L\mathbf{w}_L \tag{2.3}$$

na qual os vetores $w_i, i = 1 \cdots L$ são os vetores base aprendidos pela PCA. O erro quadrático da transformação é dada por

$$erro_i = ||(\hat{\mathbf{v}}_i - \mathbf{v}_i)||^2 \tag{2.4}$$

E o erro quadrático médio (a função custo a ser otimizada) para o conjunto de dados de aprendizagem $\mathcal{D}$ de N elementos $\mathbf{v_i}$ é

$$J(\mathbf{W}, \mathbf{Z}) = \frac{1}{N} \sum_{i=1}^{N} ||\hat{\mathbf{v}}_i - \mathbf{v}_i||^2 \tag{2.5}$$

Onde $\mathbf{W}$ é a matriz obtida concatenado-se os vetores $\mathbf{w}_i$ e $\mathbf{Z}$ a matriz obtida concatenando-se os $\mathbf{z}_i$. Se $L = D$ então $E_{total} = 0$ e, geralmente, se $L < D$, $E_{total} > 0$. Além disso, os vetores $\mathbf{w}_i$ devem ser ortonormais.

A solução ótima para o problema é conhecida e é encontrada fazendo-se $\mathbf{W} = \mathbf{V}_L$, onde $\mathbf{V}_L$ contém os L autovetores com maiores autovalores da matriz empírica de covariância dada por

$$\hat{\mathbf{\Sigma}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{v_i} \mathbf{v_i}^T \tag{2.6}$$

Além disso, os $\mathbf{z_i}$ podem ser escritos como $\mathbf{z}_i = \mathbf{W}^T \mathbf{v}_i$ [2]. Para referências futuras, nota-se o autovalor associado ao autovetor $\mathbf{w}_i$ por λ_i.

A redução de dimensionalidade obtida pela PCA é valiosa por si só, tornando alguns problemas tratáveis computacionalmente. No entanto, não é com esse objetivo que a PCA entra no algoritmo utilizado para rastreamento de pontos de interesse. Para RLMS, o interesse está nos vetores $\mathbf{w}_i$. Esses vetores representam as componentes fundamentais escondidas por trás do conjunto de dados de aprendizado. O conjunto de vetores $\mathbf{w}_i$ é utilizado para montar a matriz Φ das transformações permitidas em nosso PDM.

A Figura 2.2 mostra um exemplo em baixa dimensão da técnica de PCA. Os círculos azuis são os dados originais $\mathbf{v_i}$ e as cruzes vermelhas são as reconstruções $\mathbf{z_i}$. Observa-se que os pontos são projetados ortogonalmente sobre a linha que marca a direção principal dada por $\mathbf{w}_1$ [2].

2.2.4 Conceitos de Probabilidade e Estatística

Esta seção apresenta uma breve revisão de conceitos estatísticos necessários para se entender a formulação na Seção 2.2.5.

- Variável Aleatória

  Uma variável aleatória é uma variável cujos possíveis valores são resultados numéricos de um fenômeno aleatório. Uma variável aleatória discreta é aquela restrita a assumir valores dentre um conjunto contável de eventos. No texto que segue, uma variável aleatória $\varkappa$ assume valores em um conjunto $\Upsilon = \{\varphi_1, \varphi_2, \ldots\}$.

- Probabilidade

  Diz-se que a probabilidade da ocorrência de um evento é uma medida da confiança que ele ocorra. A probabilidade é um número real entre 0 (indicando impossibilidade) e 1 (indicando

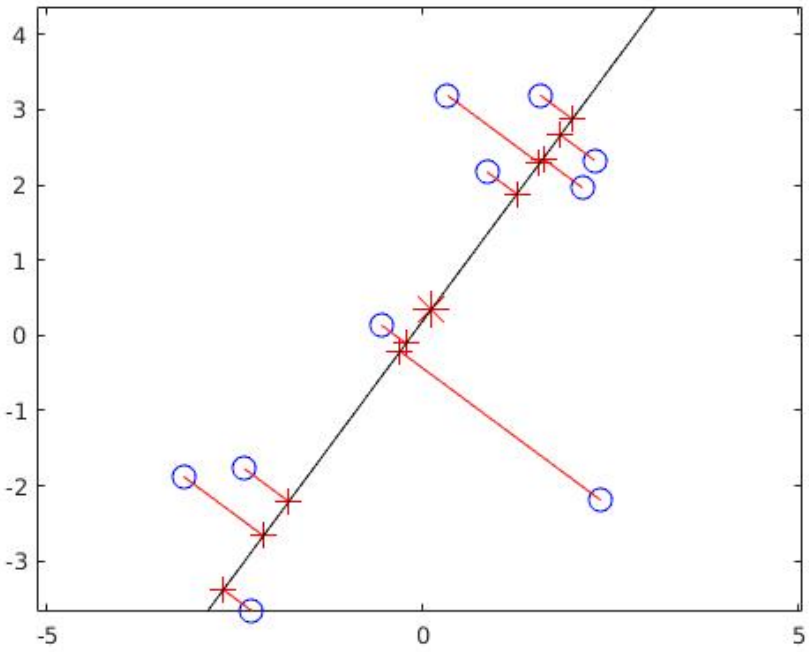


Figura 2.2: Imagem retirada de [2]. Exemplo de PCA com D = 2 e L = 1.

certeza). Quanto maior a probabilidade de um evento, mais certeza há sobre sua ocorrência. No texto que segue, $\varkappa_1$ e $\varkappa_2$ são duas variáveis aleatórias, cada uma tomando valores em $\Upsilon_1 = \{\varphi_1^1, \varphi_1^2, \ldots\}$ e $\Upsilon_2 = \{\varphi_2^1, \varphi_2^2, \ldots\}$. Denota-se a probabilidade que $\varkappa_1$ assuma valor $\varphi_1 \in \Upsilon_1$ como $p(\varkappa_1 = \varphi_1)$ ou simplesmente $p(\varphi_1)$.

- Probabilidade Conjunta

  A probabilidade conjunta para $\varkappa_1$ e $\varkappa_2$ é a distribuição de probabilidade que informa a probabilidade que $\varkappa_1 = \varphi_1$ e $\varkappa_2 = \varphi_2$, $\forall(\varphi_1, \varphi_2) \in \Upsilon_1 \times \Upsilon_2$, representada por $p(\varphi_1, \varphi_2)$. A definição pode ser expandida para k variáveis. Pode-se com isso definir a probabilidade de um vetor discreto aleatório k-dimensional $\varkappa = [\varkappa_1, \ldots, \varkappa_k]$ como:

$$p(\varkappa) = p(\varkappa_1, \varkappa_2, \ldots, \varkappa_k) \tag{2.7}$$

- Probabilidade Condicional

  Probabilidade condicional é a medida de probabilidade de um evento dado que um outro evento ocorreu. A probabilidade condicional de φ_1 dado φ_2 é escrita como $p(\varphi_1|\varphi_2)$ e pode ser calculada por:

$$p(\varphi_1|\varphi_2) = \frac{p(\varphi_1, \varphi_2)}{p(\varphi_2)} \tag{2.8}$$

- Marginalização

Suponha que um dado processo estocástico seja governado por uma função de probabilidade $p(\varphi_1, \varphi_2)$. Suponha ainda que em dada aplicação φ_1 seja uma variável de interesse e não se dê importância para φ_2. Pode-se calcular a probabilidade $p(\varphi_1)$ ao marginalizar-se φ_2:

$$p(\varphi_1) = \sum_{\varphi_2 \in \Upsilon_2} p(\varphi_1|\varphi_2)p(\varphi_2) \tag{2.9}$$

- Lei de Probabilidade Total

  O seguinte resultado combina marginalização e probabilidade condicional:

$$p(\varphi_1|\varphi_3) = \sum_{\varphi_2 \in \Upsilon_2} p(\varphi_1|\varphi_2, \varphi_3)p(\varphi_2|\varphi_3) \tag{2.10}$$

- Distribuição Normal Multivariada

  Uma função que comumente aparece em estatística é a distribuição normal. A distribuição normal, ou gaussiana, em várias dimensões é escrita como:

$$p(\varphi) = (2\pi)^{-\frac{k}{2}}|\mathbf{\Sigma}|^{-\frac{1}{2}}e^{-\frac{1}{2}(\varphi-\mu)'\mathbf{\Sigma}^{-1}(\varphi-\mu)} \tag{2.11}$$

  onde $\mathbf{\Sigma}$ é a matriz de co-variâncias e μ, um vetor k-dimensional, é a média da distribuição. Quando uma variável k-dimensional $\varkappa$ possui distribuição de probabilidade dada por uma gaussiana de média μ e covariância $\mathbf{\Sigma}$, a seguinte notação é utilizada:

$$\varkappa \sim \mathcal{N}(\varkappa; \mu, \mathbf{\Sigma}) \tag{2.12}$$

  A Figura 2.3 mostra a forma de sino da Gaussiana bidimensional.

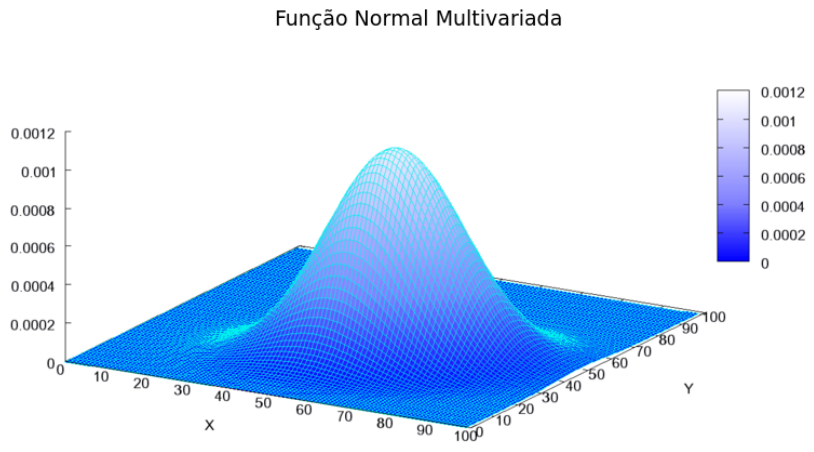


Figura 2.3: Distribuição Normal Bidimensional.

2.2.5 Formulação Estatística para o Ajuste de Parâmetros

Como visto anteriormente, ajustar o PDM a uma imagem significa calcular o conjunto $\mathbf{p} = \{s, \boldsymbol{R}, \boldsymbol{t}, \boldsymbol{q}\}$ que melhor ajusta o modelo à imagem que se quer ajudar. Isso é feito minimizando-se a função custo

$$Q(\mathbf{p}) = R(\mathbf{p}) + \sum_{i=1}^{n} D_i(\mathbf{v}_i; \mathcal{I}) \tag{2.13}$$

na qual R é um fator de regularização que penaliza deformações complexas e D_i denota uma medida de desalinhamento para o i-ésimo ponto de interesse na imagem I [17].

A Equação 2.13 pode ser interpretada probabilisticamente como a maximização da probabilidade dos parâmetros do modelo de forma que os pontos de interesse estejam alinhados com suas respectivas localizações no objeto em uma imagem [17]. A forma da função custo na Equação 2.13 assume implicitamente independência condicional entre a detecção de cada ponto chave. Probabilisticamente falando, essa independência condicional toma a forma:

$$p(\mathbf{p}|\{l_i = 1\}_{i=1}^{n}, \mathcal{I}) \propto p(\mathbf{p}) \prod_{\mathbf{i=1}}^{\mathbf{n}} p(\mathbf{l_i} = \mathbf{1}|\mathbf{v_i}, \mathcal{I}) \tag{2.14}$$

A variável discreta $l_i \in \{1, -1\}$ indica se o i-ésima ponto chave está alinhado ou desalinhado, sendo $l_i = 1$ no primeiro caso e $l_i = -1$ no segundo. Na Equação 2.14 o símbolo $\propto$ indica que existe uma relação de proporção direta entre os termos a esquerda e a direita do símbolo. Deixando mais claro, a notação $\boldsymbol{a} \propto \boldsymbol{b}$ significa que $\exists k \in \mathbb{R}$ tal que $\boldsymbol{a} = k\boldsymbol{b}$. Como se está falando de probabilidade, vale notar que a constante deve ser positiva e, portanto, maximizar o lado direito implica em maximizar o lado esquerdo da Equação 2.14. Para se obter a Equação 2.13, aplica-se o logaritmo em ambos os lados da Equação 2.14 para transformar os produtos em somatório. Além disso, algoritmos de otimização são comumente escritos para minimizar funções e não para as maximizar. Por esse motivo, inverte-se o sinal da equação obtida de forma que maximizar $f(\mathbf{x})$ é o mesmo que minimizar $-f(\mathbf{x})$. Obtém-se então uma forma para as componentes de Q na Equação 2.13:

$$P(\mathbf{p}) = -\ln(p(\mathbf{p})) \tag{2.15}$$

$$D(\mathbf{v}_i; I) = -\ln(p(l_i = 1|\mathbf{x}_1, \mathcal{I})) \tag{2.16}$$

Em [17], utiliza-se a seguinte função para a probabilidade de alinhamento de uma localização $\mathbf{x}$ de um ponto chave:

$$p(l_i = 1|\mathbf{v}, \mathcal{I}) = \frac{1}{1 + \exp\{l_i C_i(\mathbf{v}; \mathcal{I})\}} \tag{2.17}$$

Sendo C_i o classificador que distingue locações alinhadas de desalinhadas [16]. Ainda em [17], o classificador C_i utilizado é um de regressão logística aplicado em uma janela $\mathbf{\Omega_x}$ de pixeis ao redor de $\mathbf{x}$.

Para completar a abordagem é preciso especificar $p(\mathbf{p})$ em 2.15. Este termo indica um conhecimento prévio sobre a distribuição dos parâmetros. Quando se assume que todas as configurações são igualmente prováveis a formulação em 2.14 leva a uma estimação dos parâmetros $\mathbf{p}$ por maximização de probabilidade, ou, em inglês, *Maximum Likelihood*. Quando, por outro lado, supõe-se uma distribuição não uniforme dos parâmetros, tem-se uma estimação de Máximo a Priori (MAP) [16]. Lembrando que para o MDP $\mathbf{p} = \{s, \mathbf{R}, \mathbf{t}, \mathbf{q}\}$, $p(\mathbf{p})$ é especificado por partes. Em geral, assume-se um modelo gaussiano para os parâmetros não rígidos $\mathbf{q}$ e uma probabilidade uniforme sobre os parâmetros rígidos $\{s, \mathbf{R}, \mathbf{t}\}$ [17]. O que leva a distribuição a priori:

$$p(\mathbf{p}) \propto \mathcal{N}(\mathbf{q}; 0, \mathbf{\Lambda}) \tag{2.18}$$

com $\mathbf{\Lambda} = \text{diag}\{[\lambda_1; \ldots; \lambda_m]\}$, isto é, a matriz diagonal contendo os autovalores dos modos de deformação aprendidos pelo PCA. Nota-se que quanto maior o valor de λ_i mais lentamente decai a distribuição normal em 2.18. Colocando de outra forma, um maior λ_i indica que é mais provável uma componente p_i em $\mathbf{p}$ de alta magnitude.

2.2.6 Otimização de Probabilidade por Deslocamento Regularizado de Média

Devido ao erro de truncamento herdado do PCA, o modelo utilizado não pode reconstruir perfeitamente a localização perfeita das poses chaves [16]. O erro de truncamento é modelado nesta técnica por:

$$y_i = x_i + \epsilon_i \tag{2.19}$$

sendo $\epsilon_i \sim \mathcal{N}(\epsilon_i; 0, \rho I)$, y_i a variável aleatória representando a posição real da i-ésima pose chave e o parâmetro ρ pode ser aprendido durante o treinamento. A Equação 2.19 nos dá uma forma para $p(y_i|x_i)$:

$$p(y_i|v_i) = N(y_i; v_i, \rho I) \tag{2.20}$$

Para a próxima etapa assume-se que existe um conjunto de candidatos Φ_i para cada marcador i do modelo. Por exemplo Φ_i pode denotar os pontos em uma região de busca pelo i-ésimo ponto chave. Tratando a localização real dos marcadores y_i como uma variável escondida, pode-se marginalizá-la da probabilidade de alinhamento dos marcadores:

$$p(l_i = 1|v_i, I) = \sum_{y_i \in \Phi_i} p(l_i = 1|y_i, \mathcal{I}) p(y_i|v_i) \tag{2.21}$$

O termo $p(l_i = 1|y_i, \mathcal{I})$ é renomeado π_{y_i} e a equação é reescrita:

$$p(l_i = 1|\mathbf{v}_i, \mathcal{I}) = \sum_{\mathbf{y}_i \in \mathbf{\Phi}_i} \pi_{\mathbf{y}_i} N(\mathbf{v}_i; \mathbf{y}_i, \rho\mathcal{I}) \tag{2.22}$$

Substituindo 2.22 em 2.14 obtém-se:

$$p(p|\{l_i = 1\}_{i=1}^n, \mathcal{I}) \propto p(p) \prod_{i=1}^{n} \sum_{\mathbf{y}_i \in \mathbf{\Phi}_i} \pi_{y_i} N(\mathbf{v}_i; \mathbf{y}_i, \rho\mathbf{I}) \tag{2.23}$$

A Equação 2.23 pode ser minimizada por um processo iterativo que envolve calcular o vetor deslocamento de média $v = [v_1, v_2, \ldots, v_n]$ e a atualização dos parâmetros Δp através de:

$$\mathbf{v}_i = \left(\sum_{\mathbf{y}_i \in \Psi_i} \frac{\pi_{\mathbf{y}_i} \mathcal{N}(\mathbf{v}_i^c; \mathbf{y}_i, \rho\mathbf{I})}{\sum_{\mathbf{z}_i \in \Psi_i} \pi_{\mathbf{z}_i} \mathcal{N}(\mathbf{v}_i^c; \mathbf{z}_i, \rho\mathbf{I})} \mathbf{y}_i \right) - \mathbf{v}_i^c \tag{2.24}$$

e

$$\Delta\mathbf{p} = -(\rho\tilde{\mathbf{\Lambda}}^{-1} + \mathbf{J}^T\mathbf{J})^{-1}(\rho\tilde{\mathbf{\Lambda}}^{-1}\mathbf{p} - \mathbf{J}^T\mathbf{v}) \tag{2.25}$$

onde $\tilde{\mathbf{\Lambda}}$ é a matriz diagonal cujos elementos da diagonal são os elementos do vetor $[\mathbf{0}, \lambda_1, \ldots, \lambda_m]$, $\mathbf{J} = [\mathbf{J}_1; \ldots; \mathbf{J}_n]$ é o jacobiano do PDM(ver Equação 2.1) e $\mathbf{x}_c^i$ é a estimação atual do i-ésimo marcador.

2.3 Estimação de Profundidade

Um problema que aparece quando se trata de visão monocular é a falta de informação de profundidade, uma vez que o processo de formação de uma imagem consiste na captura apenas de duas das três dimensões espaciais. Com isso, uma infinidade de objetos tridimensionais podem ser relacionados a uma mesma imagem bidimensional. Ou seja, uma imagem não contém informações suficientes para reconstruir uma cena tridimensional [18]. No entanto, uma aplicação que necessite de informação de tridimensionalidade a partir de captura de imagens pode utilizar duas ou mais câmeras cujos parâmetros intrínsecos são conhecidos.

2.3.1 Modelo da Câmera

Nesta seção é definido o modelo de câmera necessário para a extração dos parâmetros intrínsecos utilizados neste trabalho.

Primeiramente, apresenta-se o modelo de projeção dos pontos do sistema de coordenadas do mundo para o plano da imagem. Considera-se que o centro da projeção é a origem do sistema de coordenadas e que $\{(X, Y, Z), Z = f\}$ corresponde ao plano da imagem. Usando um modelo de

câmera *pinhole*, um ponto no sistema de coordenadas do mundo $\mathcal{X} = (X, Y, Z)^T$ é mapeado para um ponto no plano da imagem como mostrado na Figura 2.4.

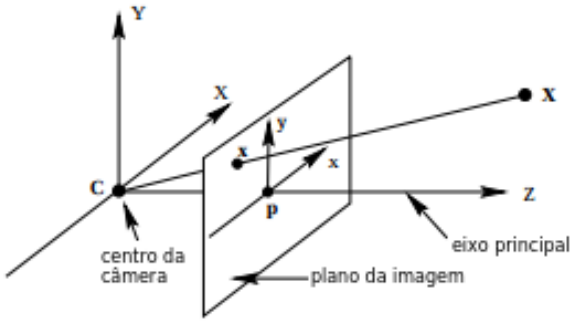


Figura 2.4: Projeção de um ponto no mundo para o plano da imagem (adaptado de [3]).

Por aplicação de uma semelhança de triângulos ilustrada na Figura 2.5 é possível mostrar que o ponto $(X, Y, Z)^T$ do espaço é mapeado para o ponto $(fX/Z, fY/Z, f)^T$ no domínio da imagem, ou seja:

$$(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T \tag{2.26}$$

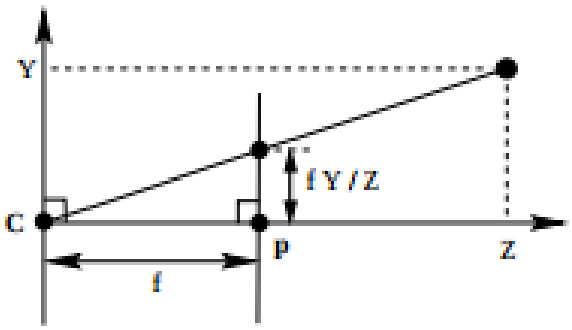


Figura 2.5: Mapeamento de um ponto no mundo para o plano da imagem (retirado de [3]).

Sendo os pontos no mundo e na imagem representados por vetores homogêneos, então a projeção central é expressa como um mapeamento linear de suas coordenadas homogêneas:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.27}$$

Fazendo $\mathcal{X}$ o vetor homogêneo $(X, Y, Z, 1)^T$ e $\mathtt{x}$ o ponto na imagem representado por um vetor de 3 dimensões, tem se que $\mathcal{P}$ representa a matriz de projeção da câmera, sendo que a relação entre esses termos dada por:

$$\mathtt{x} = \mathcal{P}\mathcal{X} \tag{2.28}$$

Este mapeamento assume que a origem do sistema de coordenadas no plano da imagem é a mesma do sistema de coordenadas da câmera; porém, isso nem sempre é verdade [3]. Na prática tem-se que:

$$(X, Y, Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T \tag{2.29}$$

sendo p_x e p_y são as coordenadas do **ponto principal**. Desta forma, ao expressar em coordenadas homogêneas a equação é:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.30}$$

Definindo $\mathbf{K}$ como:

$$\mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.31}$$

tem-se que $\mathtt{x}$ é dado por:

$$\mathtt{x} = \mathbf{K}[\mathbf{I}|\mathbf{0}]\mathcal{X} \tag{2.32}$$

A matriz $\mathbf{K}$ é chamada de matriz intrínseca da câmera.

No geral, pontos no mundo serão expressados como termos de diferentes sistemas de coordenadas. Isso implica que a origem do sistema de coordenadas do mundo geralmente não coincide com a origem do sistema de coordenadas da câmera [3], esses dois sistemas de coordenadas são então relacionados por rotação e translação, como visto na Figura 2.6 .

Se $\tilde{\mathcal{X}}$ é um vetor não homogêneo que representa um ponto no sistema de coordenadas do mundo e $\tilde{\mathcal{X}}_{cam}$ representa o mesmo ponto no sistema de coordenadas da câmera, então pode se dizer que $\tilde{\mathcal{X}}_{cam} = \mathbf{R}(\tilde{\mathcal{X}} - \tilde{\mathbf{C}})$, onde $\tilde{C}$ representa o centro da câmera no sistema de coordenadas do mundo e $\mathbf{R}$ é a matriz de rotação que representa a orientação do sistema de coordenadas da câmera. A equação em termos de coordenadas homogêneas pode ser descrita por:

$$\mathcal{X}_{cam} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathcal{X} \tag{2.33}$$

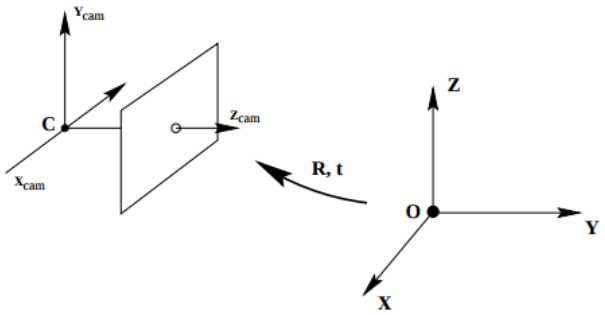


Figura 2.6: Transformação entre as coordenadas globais e as coordenadas da câmera por rotação e translação (retirado de [3]).

na qual $\mathtt{x}$ é descrito por:

$$\mathtt{x} = \mathbf{KR}[\mathbf{I}|-\tilde{\mathbf{C}}]\mathcal{X} \tag{2.34}$$

Usando a matriz de projeção da câmera definida na Equação 2.28, $\boldsymbol{\mathcal{P}}$ pode ser escrito como:

$$\boldsymbol{\mathcal{P}} = \mathbf{KR}[\mathbf{I}|-\tilde{\mathbf{C}}] \tag{2.35}$$

Este é o modelo básico dado por uma câmera do tipo *pinhole*. É muitas vezes conveniente não explicitar o centro da câmera [3], mas representar a transformação de um ponto no mundo para um ponto na imagem como $\tilde{\mathcal{X}}_{cam} = \mathbf{R}\tilde{\mathcal{X}} + \mathbf{t}$, sendo $\mathbf{t} = -\mathbf{R}\tilde{\mathbf{C}}$. Neste caso, a matriz dos parâmetros intrínsecos da câmera $\boldsymbol{\mathcal{P}}$ é escrita como

$$\boldsymbol{\mathcal{P}} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \tag{2.36}$$

2.3.2 Triangulação

Para a reconstrução tridimensional da cena, deve se resolver o problema da triangulação. Supondo que um ponto $\mathcal{X}$ em R^3 é visível em duas imagens e que as distâncias focais f_1 e f_2 correspondentes a essas duas imagens são conhecidas, sendo u e u' as projeções do ponto $\mathcal{X}$ nas imagens, as linhas no espaço correspondentes aos dois pontos da imagem podem ser, a partir desses dados, facilmente computadas. O problema da triangulação é achar a interseção destas duas linhas no espaço [19]. Este modelo pode ser visualizado na Figura 2.7.

Com a Figura 2.7 como referência, é possível, por semelhança de triângulos, derivar as seguintes equações para o valor X do ponto no sistema de coordenadas do mundo.

$$X = (u_x/f_1)Z \tag{2.37}$$

ou

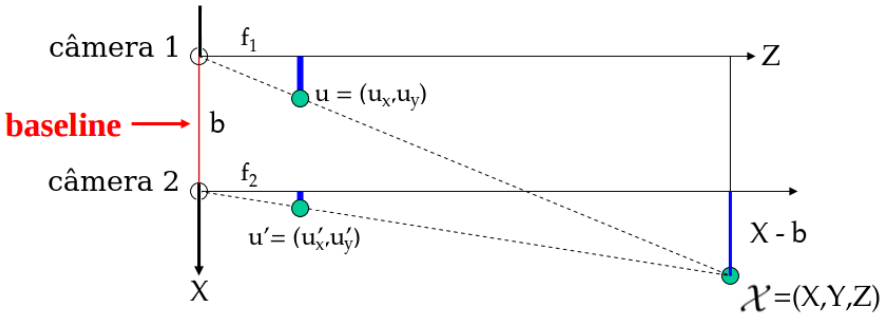


Figura 2.7: Disposição das câmeras (adaptado de [4]).

$$X = (u'_x/f_2)Z + b \tag{2.38}$$

A relação do valor Y do ponto no sistema de coordenadas do mundo pode também ser relacionado com sua projeção no plano da imagem por uma semelhança de triângulos, como mostrado na Figura 2.5. As equações de Y são mostradas a seguir.

$$Y = (u_y/f_1)Z \tag{2.39}$$

ou

$$Y = (u'_y/f_2)Z \tag{2.40}$$

A partir das Equações 2.37 e 2.38 é possível expressar o valor da profundidade Z do ponto como

$$Z = f_1 f_2 b/(u_x f_2 - u'_x f_1) \tag{2.41}$$

Em que b se refere a *baseline*, ou seja, a distância fixa entre as câmeras, u'_x e u_x são os valores em x da projeção do ponto no plano da imagem das câmeras 1 e 2, respectivamente, e f_1 e f_2 são as distâncias focais obtidas a partir da matriz intrínseca de cada câmera, esta que pode ser derivada através da matriz da câmera, também chamada de matriz de calibração. A matriz de calibração pode ser obtida usando a *Calibration Toolbox for Matlab*, onde a partir de uma série de imagens de um tabuleiro de xadrez (*chessboard*) capturadas pela mesma câmera é possível relacionar pontos iguais em várias imagens e então derivar a matriz de calibração [9].

2.4 Renderização de Objetos Tridimensionais

A renderização de objetos tridimensionais neste trabalho foi feita utilizando a Interface de Programação de Aplicações (API) *Open Graphics Library* (OpenGL). Para a compreensão de como

esta renderização é feita, deve-se entender como é formado um objeto e como ele é posicionado no sistema de coordenadas.

Um triângulo é a estrutura mais básica em computação tridimensional [20] e todos os objetos neste trabalho são representados computacionalmente por um conjunto de pequenos triângulos texturizados. Um triângulo é definido por três vértices e cada vértice é composto por três coordenadas: X_p, Y_p e Z_p.

A informação de um objeto tridimensional completamente renderizado é armazenada em um *Vertex Array Object* (VAO), uma estrutura que contém um ou mais *Vertex Buffer Objects* (VBOs). Um VBO é um *buffer* de memória contido na memória de alta velocidade da placa de vídeo, e contém informações sobre os vértices [21]. Podem, por exemplo, conter as coordenadas dos vértices ou talvez a cor associada a cada vértice.

Desta maneira é possível criar a forma de um objeto construindo um VBO que contém as coordenadas de todos os triângulos que compõem o objeto; porém, assim sempre existirão duplicatas quando vértices de dois triângulos compartilharem uma aresta [5]. Para evitar esta replicação desnecessária, os pontos em si são guardados separadamente do objeto e a malha é construída com índices para estes pontos. O efeito desta indexação é ilustrado na Figura 2.8.

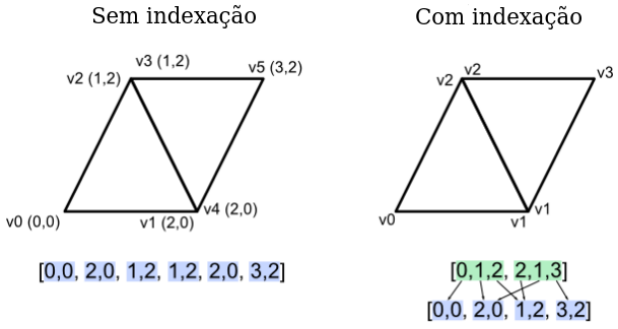


Figura 2.8: Indexação do VBO (adaptado de [5]).

Além disso, esta forma de representação tem a vantagem de apresentar uma estrutura bem semelhante a forma como a geometria tridimensional é armazenada em um arquivo no formato OBJ. Esta semelhança faz com que o carregamento destes arquivos para a renderização utilizando OpenGL fique bem mais simples.

Para definir um objeto completamente e não apenas sua forma, se faz necessário o uso de outros VBOs como, por exemplo, um VBO contendo informações das coordenadas de textura e um VBO contendo informações das direções normais a superfície de cada triângulo. A direção normal é importante para que se aplique iluminação sobre o objeto renderizado e sem iluminação adequada não seria possível ver os detalhes da superfície, apenas seu contorno.

Para o posicionamento do objeto no sistema de coordenadas ser definido, primeiramente são definidas as seguintes matrizes de transformação: translação, rotação e escala.

A matriz de translação é dada por:

$$\begin{bmatrix} 1 & 0 & 0 & D_y \\ 0 & 1 & 0 & D_x \\ 0 & 0 & 1 & D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} X_p + D_x \\ Y_p + D_y \\ Z_p + D_z \\ 1 \end{pmatrix} \tag{2.42}$$

A matriz de escala por:

$$\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} S_x \cdot X_p \\ S_y \cdot Y_p \\ S_z \cdot Z_p \\ 1 \end{pmatrix} \tag{2.43}$$

E as matrizes de rotação:

Rotação no eixo X:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sen\theta & 0 \\ 0 & sen\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} X_p \\ cos\theta \cdot Y_p - sen\theta \cdot Z_p \\ sen\theta \cdot Y_p + cos\theta \cdot Z_p \\ 1 \end{pmatrix} \tag{2.44}$$

Rotação no eixo Y:

$$\begin{bmatrix} cos\theta & 0 & sen\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sen\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} cos\theta \cdot X_p + sen\theta \cdot Z_p \\ Y_p \\ -sen\theta \cdot X_p + cos\theta \cdot Z_p \\ 1 \end{pmatrix} \tag{2.45}$$

Rotação no eixo Z:

$$\begin{bmatrix} cos\theta & -sen\theta & 0 & 0 \\ sen\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = \begin{pmatrix} cos\theta \cdot X_p - sen\theta \cdot Y_p \\ sen\theta \cdot X_p + cos\theta \cdot Y_p \\ Z_p \\ 1 \end{pmatrix} \tag{2.46}$$

Os vértices são aqui definidos como vetores de quatro dimensões pois em OpenGL este quarto parâmetro, chamado de W_p, define se o vetor é uma posição no espaço, caso $W_p = 1$, ou se o vetor é uma direção, caso $W_p = 0$.

Com essas matrizes definidas pode-se derivar outras três matrizes: a matriz do modelo, a matriz de visualização e a matriz de projeção. Essas matrizes são úteis para separar as transformações de forma eficiente.

Um objeto é definido por um conjunto de vértices e as coordenadas X_p, Y_p e Z_p destes vértices são definidos relativamente ao centro do objeto, ou seja, se um vértice está localizado em $(0, 0, 0)$, então ele está localizado no centro do objeto [5]. Para mover o objeto no espaço virtual utiliza-se a matriz do modelo, que é um produto das matrizes de translação, rotação e escala aplicadas no objeto. A aplicação desta transformação coloca vértices do objeto no sistema de coordenadas do espaço. A Figura 2.9 ilustra a transformação.

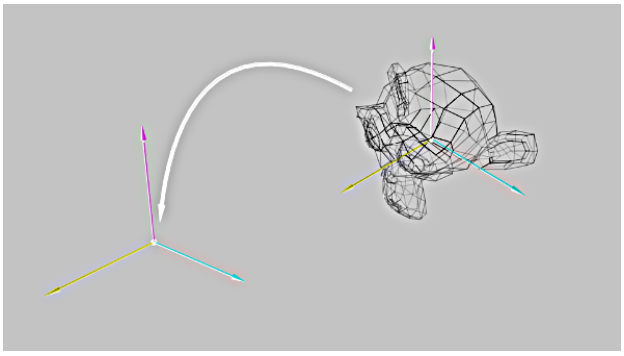


Figura 2.9: Efeito da Matriz do Modelo (adaptado de [5]).

A matriz de visualização pode ser interpretada como o posicionamento e angulação de uma câmera que irá apontar para o cenário observado, de forma que é possível mover esta câmera para observar o objeto a partir de outras posições e direções arbitrárias. Apesar da interpretação acima, o que realmente acontece é que não é a câmera que move e sim todo o cenário, incluindo o objeto. A aplicação desta transformação por meio do produto a esquerda com a matriz de visualização coloca as coordenadas dos vértices no sistema de coordenadas da câmera. A Figura 2.10 ilustra o que acontece.

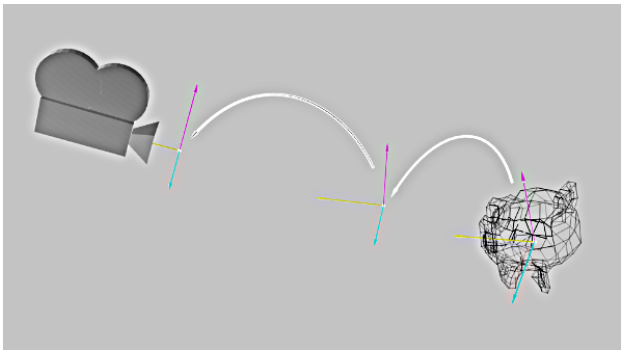


Figura 2.10: Efeito da Matriz de Visualização (adaptado de [5]).

A função da matriz de projeção é basicamente fazer com que objetos mais distantes da câmera pareçam menores, como ocorre no mundo real. No sistema de coordenadas da câmera dois vértices com um mesmo par de coordenadas (X_p, Y_p) serão renderizados no mesmo local, o que não é sempre verdade. Isto acontece pois a coordenada Z_p não está sendo levada em conta. Para a profundidade

ser levada em conta e a perspectiva corrigida a matriz de projeção transforma o espaço de visão da câmera, que inicialmente é um hexaedro irregular, em um cubo [5], com isso os objetos são deformados de modo que o que está mais próximo da câmera pareça maior e o que está mais distante da câmera pareça menor, tendo-se, por fim, um sistema de coordenadas homogêneas. As Figura 2.11 ilustra o que acontece.

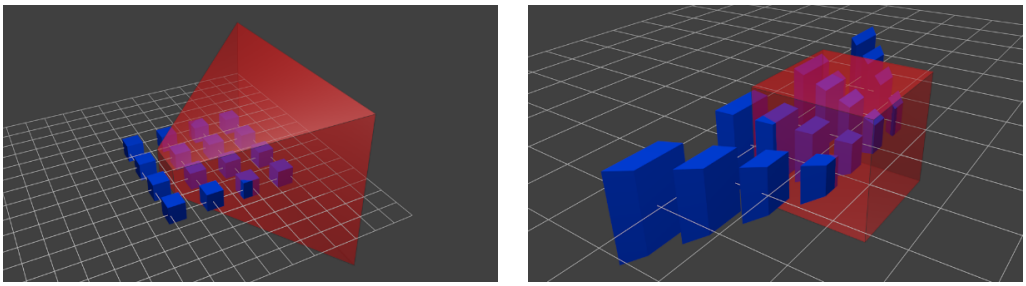


Figura 2.11: Efeito da Matriz de Projeção (retirado de [5])

Um diagrama do efeito dessas três matrizes no sistemas de coordenadas onde se encontra o objeto pode ser visualizado na Figura 2.12.

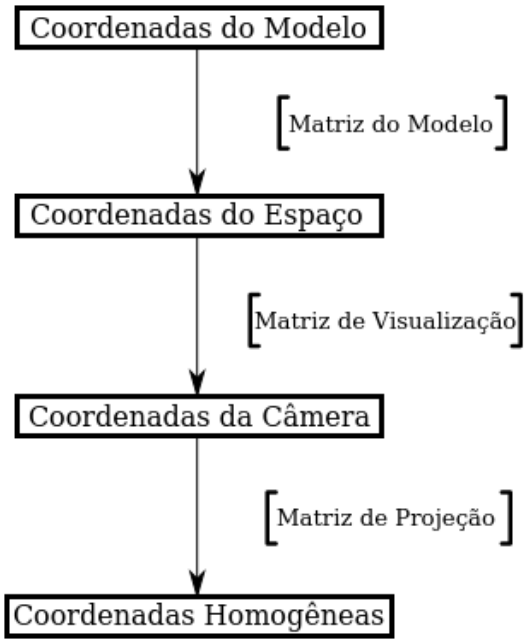


Figura 2.12: Diagrama do efeito das matrizes (adaptado de [5]).

2.5 Mistura de Poses

Mesmo nas mais simples animações comerciais, um modelo tridimensional utilizado costuma ser composto por milhares de vértices e polígonos. Renderizar um rosto expressivo requer definir valores para cada um dos vértices envolvidos. Adicione ao problema o fato de que um vídeo de

poucos segundos pode exigir centenas de expressões ligeiramente diferente uma da outra para que o animação seja percebida de forma suave.

Uma das técnicas utilizadas pelos animadores digitais é a Animação Esqueletal (*Skeletal Animation*) na qual se define um esqueleto sobre o modelo tridimensional, de forma que movimento de cada ponto sobre o modelo está atrelado ao movimento de rotação e translação das juntas. O artista pode então posicionar as juntas em poses chaves do movimento e pedir para que o computador interpole os pontos intermediários. Comumente, mesmo depois da interpolação é necessário que o artista retoque manualmente alguns pontos para que o resultado se torne ainda mais atrativo.

A Figura 2.13, retirada do tutorial em [6], mostra a tela do problema *Blender* enquanto o usuário define os *ossos* ou a *armadura* sobre o modelo. Mais tarde, pode-se movimentar os ossos individualmente e a malha do modelo se moverá juntamente.

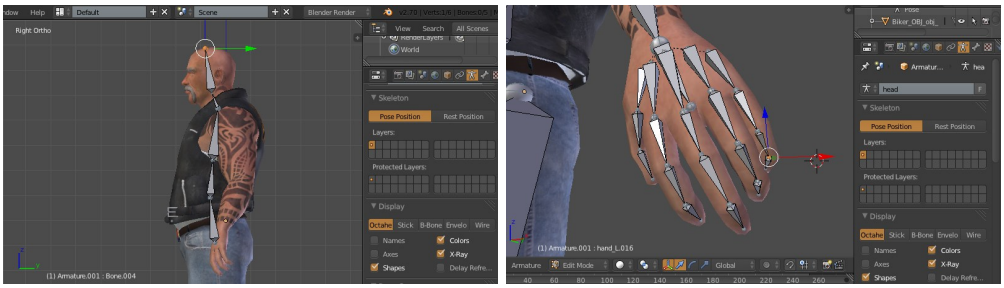


Figura 2.13: Tela do programa gratuito de código aberto *Blender* (retirado de [6]). Cada um dos elementos em cinza é um osso do modelo. O termo utilizado no meio artístico é que se trata de um modelo *rigado*.

O uso de *rigged models* é recomendado para animar cenas onde o corpo inteiro do personagem se mexe. No entanto, é ineficiente para animar expressões faciais. O problema é que as transformações permitidas pelo modelo de juntas se limita a transformações rígidas e a forma como os músculos do rosto se comportam é melhor representada por transformações não rígidas, isto é, deformações na geometria do corpo.

A técnica de Mistura de Poses - MP, do inglês *Blend Shapes* ou *Morph Target*, é uma das opções comumente empregadas para animar objetos deformáveis como a face humana. Outros objetos comumente animados com essa técnica são peças de roupa e pele, uma vez que esses objetos são dificilmente modelados por um modelo esqueletal [22]. A técnica consiste em gerar poses intermediárias como uma combinação linear de poses pré-definidas. A Figura 2.14 mostra três poses utilizadas para a MP mostrada na Figura 2.15.

Em sua versão mais simples, a MP requer que cada um dos modelos representando as poses chaves tenham a mesma quantidade de vértices. Mais do que isso, é necessário que se seja capaz de mapear todos os pontos de um modelo nos pontos de outro. Colocando de outra forma, dá-se um identificador para cada vértice que compõe a malha do modelo [7].

Um modelo M(t) que se deforma no tempo é dado pela tupla de N pontos tridimensionais $(v_1(t), v_2(t), \ldots, v_N(t))$. Considere o espaço de poses ao longo do tempo $\Omega = \{M(t), t \in R\}$ e amostre L poses distintas $M_i = M(t_i), i = 1 \ldots L$. A MP consiste em construir o espaço das poses

Figura 2.14: Três modelos a serem misturados pela técnica de *Blend Shapes* (retirado de [7]). Respectivamente da esquerda para a direita, os modelos representam a pose neutra, a expressão de felicidade e de raiva. Nota-se a diferença na forma dos lábios e nas sobrancelhas.

Figura 2.15: Resultado obtido ao combinar 0.5 da pose alegre, 0.3 da pose com raiva e 0.2 da pose neutra (retirado de [7]).

que podem ser obtidas como uma combinação linear das poses amostradas, limitado a fato de que a soma dos coeficientes da combinação seja unitária. Além disso, é comum que se defina uma das amostras como a pose neutra M_1. O peso dessa amostra neutra é escolhido como o negativo da soma das outras amostras mais um.

Uma pose J pertencente ao espaço de poses criado deve ser obtido por:

$$J = \sum_{i=1}^{L} w_i M_i \tag{2.47}$$

Um pouco de manipulação pode deixar mais clara a forma como a mistura de deformações não rígidas acontece.

$$J = w_1 M_1 + w_2 M_2 + \ldots + w_L M_L \tag{2.48}$$

$$J = (1 - (w_2 + w_3 + \ldots w_L))M_1 + w_2 M_2 + \ldots + M_\theta + \ldots + w_L M_L \tag{2.49}$$

$$J = M_1 + w_1(M_2 - M_1) + \ldots + w_L(M_L - M_1) \tag{2.50}$$

$$J = M_1 + \sum_{i=2}^{L} w_i \Delta M_i \tag{2.51}$$

sendo $\Delta M_i = M_i - M_1$ é a deformação relativa entre M_i e M_1. A última equação deixa claro que as poses obtidas podem ser vistas como a soma de deformações conhecidas superpostas com a pose neutra.

Programas de animação utilizam a Equação 2.51 da seguinte maneira:

- modela-se a pose neutra do seu personagem;
- geram-se expressões chaves a partir da pose neutra apenas pela movimentação dos seus vértices;
- carrega-se a pose neutra e as demais no framework de mistura de poses chaves;
- mexendo-se em cursores associados ao peso de cada pose na mistura produzem-se poses intermediárias que são marcadas a pontos no tempo do vídeo;
- pede-se então que o programa interpole as posições marcadas para que o resultado seja suave.

A técnica de Animação Esqueletal e a MP são semelhantes no fato em que o artista configura poses chaves e então o computador completa as lacunas por meio de interpolação. No entanto, as técnicas produzem resultados de natureza diferente. Enquanto a primeira pode produzir um braço que tem a junta do cotovelo em 45°, a MP produz um rosto que está meio bravo, meio alegre.

Colocando-se em termos bem simples, depois que os modelos para as poses chaves estão prontos, a técnica de MP recebe um vetor de pesos e produz uma nova pose como uma média ponderada das poses chaves.

Programas que realizam a técnica de mistura de poses chaves não são novidade. O nosso objetivo é substituir a mão do artista mexendo os cursores de uma interface gráfica, por um programa que associa posições do cursor a propriedades do vídeo em execução.

2.6 Estabilização do movimento

Devido a ruídos oriundos do processo de captura de imagem e mesmo a características estatísticas de alguns algoritmos de rastreamento, é comum que o resultado da detecção oscile em torno do valor correto, mesmo quando a detecção é feita adequadamente. Em muitas aplicações é desejável que se suavize o resultado da detecção antes que este seja utilizado para a tomada de decisões. É com esse objetivo que são utilizadas técnicas de filtragem digital.

2.6.1 Sequências e Sistemas Lineares Invariantes ao Deslocamento

Para a discussão que segue, é dito que uma sequência é uma função $N_0 \rightarrow R$. Isto é, para cada número $n \in \{0, 1, 2, , \cdots\}$ a sequência χ associa o número $\chi(n) \in R$.

Alguns sistemas podem ser descritos por meio de suas respostas a sequências de entrada. Por exemplo, o sistema H produz a sequência de saída $h(n)$ quando é submetido a sequência de entrada $\chi(n)$.

Um sistema é dito linear quando vale a seguinte relação:

$$\gamma(n)_{\alpha\chi_1+\beta\chi_2} = \alpha\gamma_{\chi_1}(n) + \beta\gamma_{\chi_2}(n) \tag{2.52}$$

na qual $\gamma(n)_{\alpha\chi_1+\beta\chi_2}$ é a resposta do sistema quando submetido à sequência de entrada $\alpha\chi_1(n)+\beta\chi_2(n)$ e $\gamma_{\chi_1}(n)$ e $\gamma_{\chi_2}(n)$ são as saídas do sistema quando submetido a sequências de entrada $\chi_1(n)$ e $\chi_2(x)$ respectivamente. Em palavras, o sistema é linear quando a sua resposta à combinação linear de dois sinais é igual à combinação das suas respostas aos dois sinais quando aplicados independentemente.

Um sistema é invariante ao deslocamento quando:

$$\gamma_{\chi(n-k)} = \gamma_{\chi(n)}(n-k) \tag{2.53}$$

significando que o comportamento da resposta ao sinal de entrada depende unicamente do comportamento do sinal de entrada e não do instante em que ele foi aplicado.

Sistemas Lineares e Invariantes ao Deslocamento (LTI) são úteis pois aproximam adequadamente o comportamento de muitos sistemas de interesse. Para tais sistemas vale a seguinte relação:

seja $h(n)$ a resposta finita ($h(n) = 0 \ \forall \ n \geq \mathcal{M}$) de um sistema LTI ao sinal de entrada $\delta(n)$ dado por:

$$\delta(n) = \begin{cases} 1, & \text{se } n = 0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.54}$$

o sinal $\gamma(n)$ da resposta do mesmo sistema a um sinal de entrada $\chi(n)$ qualquer será dada por:

$$\gamma(n) = \sum_{k=0}^{\mathcal{M}-1} h(k)\chi(n-k) = h(n) * \chi(n) \tag{2.55}$$

Com isso, um sistema LTI é muitas vezes representado pela sua resposta $h(n)$ ao impulso $\delta(n)$.

2.6.2 A Transformada Z

Existem várias maneiras de se representar uma sequência, e a transformada Z pode ser vista como uma delas. A transformada Z quando aplicada a uma sequência $\chi : N_t \rightarrow R$ retorna uma função $X_z : C_z \rightarrow X_z$. A transformada é dada pela Equação 2.56:

$$\mathcal{Z}\{\chi(n)\} = X_z(z) = \sum_{i=0}^{\infty} \chi(i) z^{-i} \tag{2.56}$$

A informação contida na transformada z é, a princípio, a mesma contida na sequência inicial, mas ao mudar a representação da sequência algumas de suas características podem ser melhor entendidas. Por exemplo, se $H_z(z) = \mathcal{Z}\{h(n)\}$ é a transformada da resposta ao impulso do sistema, então é possível mostrar que a transformada $Y_z(z)$ da resposta do sistema a um sinal de transformada $X_z(z)$ é dada por:

$$Y_z(z) = H_z(z)X_z(z) \tag{2.57}$$

Outro resultado importante permite recuperar a sequência $\chi(n)$ a partir de $X_z(z)$ através da transformada Z inversa:

$$\mathcal{Z}^{-1}\{X_z(z)\} = \frac{1}{2\pi} \oint X_z(z) z^{n-1} dz = \chi(n) \tag{2.58}$$

2.6.3 Resposta em Frequência

A função $X_z(z)$ pode ser avaliada em qualquer ponto $z \in C_z$. Em particular ela pode ser avaliada nos pontos $z = e^{j\omega}$ do círculo unitário. Quando isso é feito, tem-se a transformada de Fourrier em Tempo Discreto:

$$X_d(\omega) = X_z(z)|_{z=e^{j\omega n}} = \sum_{n=0}^{\infty} \chi(n) e^{-j\omega} \tag{2.59}$$

com transformada inversa:

$$\chi(n) = \frac{1}{2\pi j} \int_{-\pi}^{\pi} X_d(\omega) e^{j\omega} d\omega \tag{2.60}$$

Este último resultado tem uma interpretação interessante: ele nos diz que uma sequência x(n) qualquer pode ser obtida como uma combinação de um número infinito de sequências de uma coleção de sequências exponenciais $\{e^{j\omega}, -\pi \leq \omega \leq \pi\}$. Conhecendo a equação de Euler:

$$e^{wj} = cos(\omega) + jsen(\omega) \tag{2.61}$$

e com $\omega = 2\pi f_q$, f_q a frequência do sinal, diz-se que uma **sequência qualquer $\boldsymbol{\chi(n)}$ pode ser escrita como uma combinação de sequências senoidais primitivas**. A vantagem desta análise é que em muitas aplicações as componentes senoidais primitivas representam características físicas do sistema em que se deseja trabalhar.

Por exemplo, uma sequência obtida por amostragem de um sensor ruidoso é composta pela justaposição da variável real medida e de diversas interferências. Se em uma dada aplicação sabe-se que todas as interferências estão associadas a sinais de alta frequência, é possível se blindar do ruído e recuperar o sinal verdadeiro apenas filtrando as componentes de alta frequência.

2.6.4 Equação de Diferenças

Uma equação de diferença calcula uma amostra de saída no tempo n baseado em amostras de entradas passadas e presentes e amostras de saída passadas no domínio do tempo [23]. A equação geral para um sistema causal, linear e invariante no tempo pode ser escrita como:

$$\gamma(n) = \sum_{i=0}^{M_l} b_i \chi(n-i) - \sum_{j=1}^{N_l} a_j \gamma(n-j) \tag{2.62}$$

Quando a equação de diferenças utiliza amostras de saída passadas (como $\gamma(n-1)$) no cálculo da amostra de saída presente $\gamma(n)$ dizemos que há realimentação no sistema. Os coeficientes b_i são chamados de coeficientes diretos e os a_j são ditos os coeficientes de realimentação [23].

Equações de diferença podem ser utilizadas para implementar filtros digitais.

2.6.5 Filtros Digitas

Um filtro digital é uma forma de ir de um sinal digital para um outro [24], podendo ser implementado em *software*, por meio de uma sub-rotina de computador, como em *hardware*, por

meio de um projeto de circuito integrado. Em aplicações, filtros digitais são comumente empregados para realçar características do sinal de interesse ou remover características indesejadas.

Uma forma comum de especificar um filtro digital é por meio da resposta em frequência. A Figura 2.16 mostra a especificação de um filtro passa baixas ideal. Nesse tipo de filtro, é desejado remover do sinal de entrada qualquer componente de frequência $\omega > \omega_c$, na qual ω_c é a frequência de corte do filtro (em radianos/segundo).

2.6.6 Filtros Digitais de Resposta Finita ao Impulso

Um filtro digital é dito de Resposta Finita ao Impulso, do inglês *Finite Impulse Response* (FIR), é aquele cuja resposta ao impulso tem comprimento finito. Seja $h(n)$ a resposta do filtro ao impulso $\delta(n)$, o filtro é FIR se $\exists \mathcal{M} \geq 0$ tal que $h(n) = 0$ para todo $n \geq \mathcal{M}$.

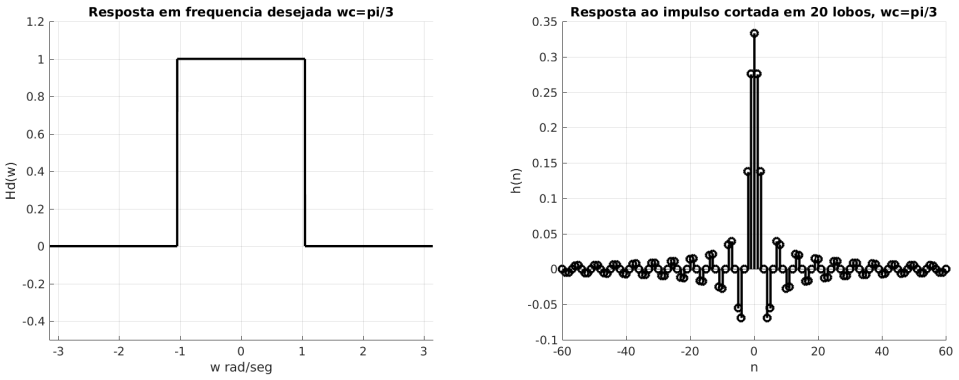
Um resultado útil da teoria de Processamento de Sinais Digitais é que um filtro FIR pode ser implementado por uma equação de diferenças sem realimentação, ou seja, uma equação da forma:

$$\gamma(n) = \sum_{i=0}^{\mathcal{M}} b_i \chi(n-i) \tag{2.63}$$

Filtros com a forma acima são desejáveis pois são mais facilmente implementáveis.

2.6.7 Projeto de Filtro Passa-Baixa pela Técnica de Janela

O princípio por trás dessa técnica é se aproximar do filtro passa-baixas ideal, mostrado na Figura 2.16. A Equação 2.60 permite calcular a resposta ao impulso $h(n)$ que produziria tal filtro ideal.



(a) (b)

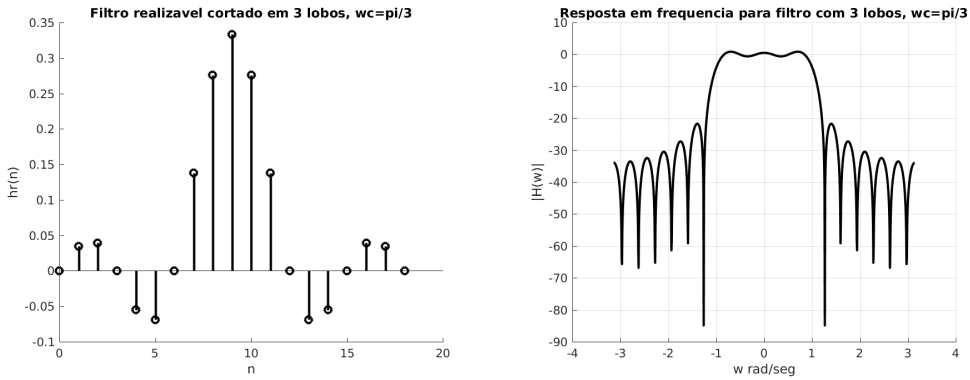
Figura 2.16: Em (a) é observada a resposta ao impulso em frequência ideal de um filtro passa-baixas. A frequência de corte projetada é $\omega_c = \frac{\pi}{3}$ rad/segundos. Em (b) são observados alguns pontos de $h(n)$. Na realidade a sequência é infinita para ambos os lados do eixo.

Pode ser observado que além da resposta ser infinita, ela é não causal pois possui termos não

nulos para $n < 0$. Ou seja, o sistema precisaria começar a responder antes mesmo que a entrada impulso fosse aplicada. Fica claro que a realização de tal filtro é impossível. Notando que a forma de onda de $h(n)$ é composta por lobos positivos e negativos, a técnica de janela busca aproximar o filtro ideal ao limitar $h(n)$ a um número finito de pontos não nulos, de forma a manter um número inteiro de lobos simétricos. Além disso, $h(n)$ precisa ser deslocado para a direita caso o objetivo seja um filtro fisicamente realizável.

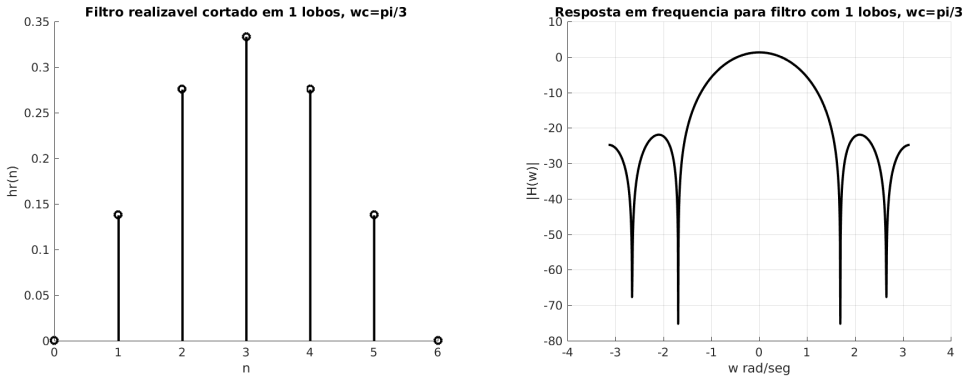
A Figura 2.17 mostra o filtro e a resposta em frequência obtidos caso a resposta ideal seja cortada com o lobo central e mais dois lobos simétricos. A Figura 2.18 mostra o filtro e a resposta em frequência obtidos caso mantenha-se somente o lobo central. Nota-se que no primeiro caso são necessários 18 valores, enquanto no segundo apenas 6. Por outro lado, a resposta em frequência no primeiro caso se aproxima melhor do filtro passa-baixas ideal. Nota-se que há um compromisso que precisa ser feito entre a qualidade do filtro e a quantidade de memória e processamento necessários para implementar o filtro.

Para efeito de comparação, a Figura 2.19 mostra o mesmo projeto aplicado para a frequência de corte $\omega_c = \frac{\pi}{10}$. Nota-se que uma frequência de corte mais baixa requer uma quantidade de pontos não nulos no filtro maior que aquela vista na Figura 2.17.



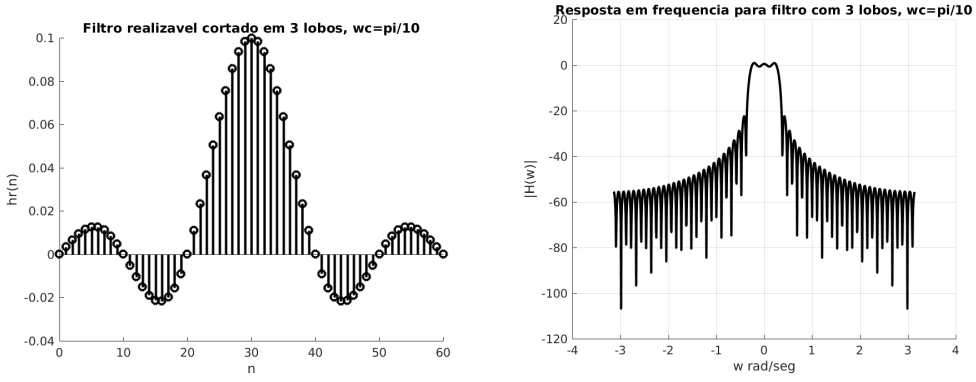
(a) (b)

Figura 2.17: Em (a) é observada a resposta ao impulso do filtro realizável obtido cortando-se o lobo principal mais dois lobos para cada lado da resposta ideal. Note que o eixo horizontal foi transladado para permitir uma resposta realizável. Em (b) é observada a resposta em frequência obtida com esse filtro. Nota-se que o eixo vertical é mostrado em decibéis.



(a) (b)

Figura 2.18: Em (a) é observada a resposta ao impulso do filtro realizável obtido cortando-se o lobo principal somente. Em (b) é observada a resposta em frequência obtida com esse filtro.



(a) (b)

Figura 2.19: Filtro projetado com 3 lobos e frequência de corte $w_c = \frac{\pi}{10}$.

Capítulo 3

Metodologia

Este capítulo descreve a sequência de passos utilizada para extrair movimento facial a partir de um par de imagens e transferi-lo para um modelo tridimensional. O objetivo do sistema de animação computacional de baixo custo proposto é a transferência de movimento da imagem para o modelo. Portanto a qualidade do resultado final deve ser analisada através da comparação entre o movimento esperado e aquele efetivamente gerado. Este trabalho não propõe uma metodologia formal para a avaliação da qualidade final, essa avaliação é feita de forma qualitativa pela observação dos resultados. No entanto, métricas consideradas mais quantitativas podem ser definidas para as etapas intermediárias da aplicação e alguns experimentos são propostos com o objetivo de avaliá-las. Este capítulo é dividido em duas partes: a primeira descreve as etapas da aplicação proposta e a segunda descreve os experimentos de validação dessas etapas.

3.1 Etapas de Desenvolvimento

A Figura 3.1 apresenta um diagrama que ilustra as etapas da implementação do método bem como relaciona todas as técnicas utilizadas. As etapas apresentadas representam o caminho dos dados dentro de cada iteração do programa em execução. Como sugerido pelo diagrama, as etapas são as seguintes: Posicionamento das Câmeras para Captura das Imagens de Entrada, Rastreamento de Pontos do Rosto, Estimação da Profundidade, Atualização dos Pesos de Mistura - Razão de Distância, Filtros e Mistura de Poses.

3.1.1 Posicionamento das Câmeras para Captura das Imagens de Entrada

A estrutura utilizada neste trabalho para captura de vídeo consiste na fixação de duas *webcams* comuns posicionadas paralelamente sobre uma plataforma de madeira. As câmeras utilizadas são ambas do modelo *Phillips Webcam Easy SPC530*, com uma resolução de 1.3MP. Vale notar que essa é uma estrutura extremamente simples e barata para um sistema de animação computacional, custando, na data atual de publicação deste trabalho, em volta de cem reais. Esse é um dos fatores que torna o sistema proposto neste trabalho um sistema de baixo custo. A Figura 3.2 mostra a

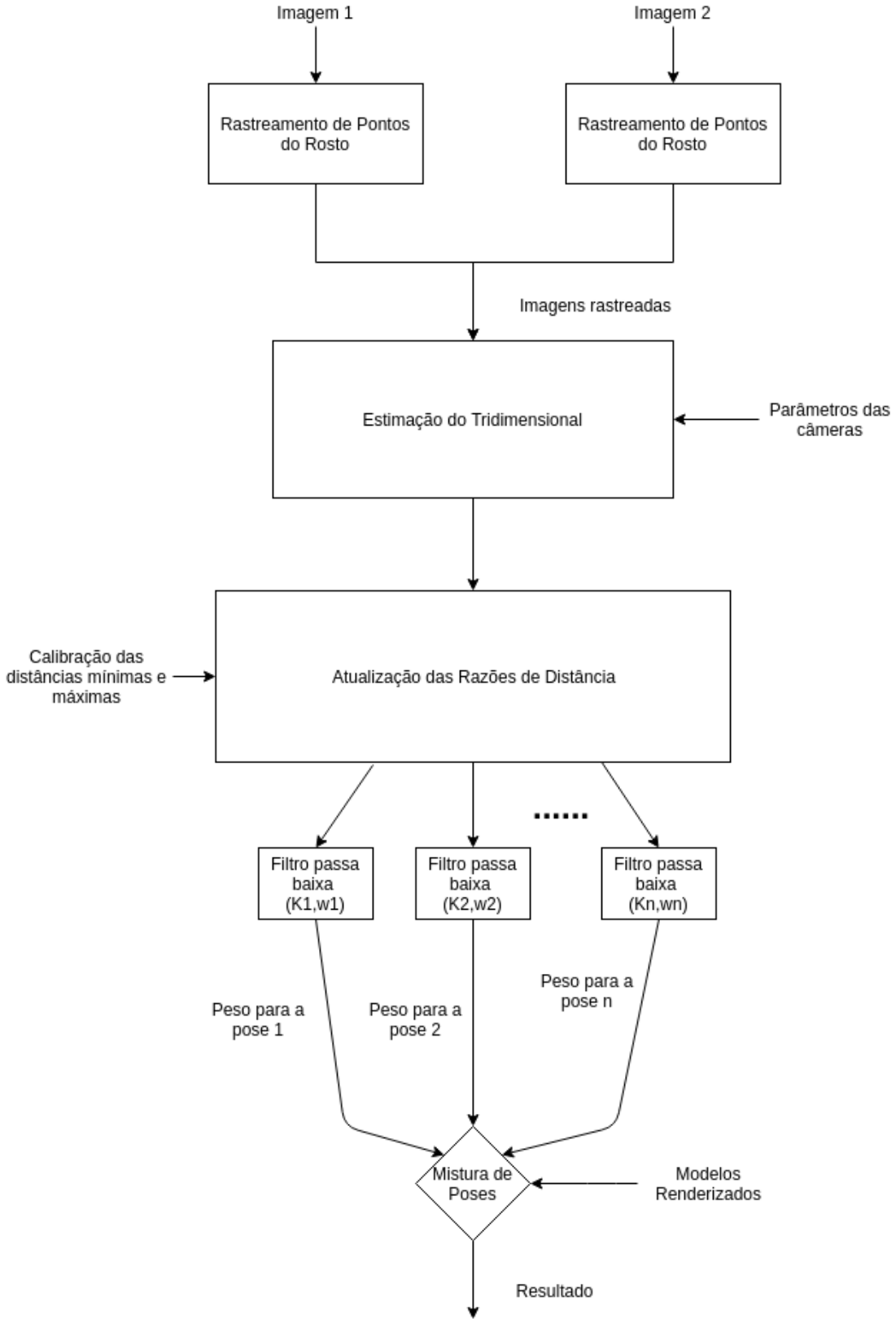


Figura 3.1: Diagrama completo para a animação de cada *frame*. As setas verticais representam o fluxo entre as técnicas dentro de cada iteração do programa. As setas horizontais indicam dados provenientes de uma etapa de calibração do sistema.

fixação do par de câmeras utilizado, sendo '$b = 18cm$' a distância entre os centros das câmeras e '$a = 2cm$' a distância entre o centro de captura da câmera e a aresta do suporte. Este parâmetro a deve ser conhecido pois durante os experimentos é necessário medir a distância do alvo ao centro das câmeras e isso é mais facilmente feito tomando-se a distância do alvo ao suporte e utilizando o parâmetro a para recuperar a medida necessária. O parâmetro b é utilizado durante a etapa de estimação de profundidade e é equivalente ao que foi chamado de *baseline* na Figura 2.7. A câmera de índice 1 é aquela que fica à direita do usuário quando este encontra-se de frente para a montagem na Figura 3.2 e a de índice 2 é aquela que se encontra à esquerda. Ao relacionar os pontos da câmera aos termos da Equação 2.41, chama-se o ponto alvo projetado no plano da imagem da câmera 1 de u e o ponto alvo projetado no plano da imagem da câmera 2 de u'. Além disso, ao conectar as câmeras às entradas USB do computador que roda a aplicação, deve-se conectar primeiramente a câmera 1 e em seguida a câmera 2. Isso deve ser feito para que o programa localize corretamente as câmeras, definindo qual imagem de entrada é proveniente da câmera da direita e qual é proveniente da câmera da esquerda.

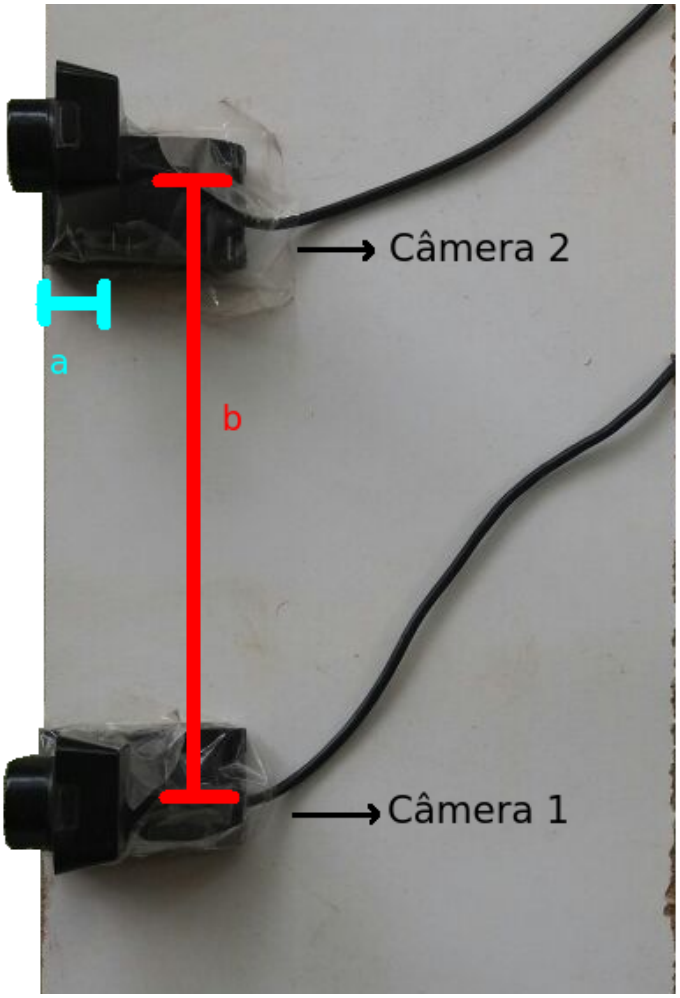


Figura 3.2: Fixação das câmeras. Estrutura utilizada neste trabalho para a captura das imagens de entrada.

3.1.2 Rastreamento de Pontos do Rosto

O rastreamento de pontos do rosto é a segunda etapa do processo iterativo de animação implementado neste trabalho. Como pode ser visto na Figura 3.1, em cada iteração do programa em execução as duas imagens de entrada passam por esse processo separadamente.

Para fins de rastreamento de pontos do rosto, utiliza-se a *Software Development Kit* (SDK) *CSIRO Face Analysis* [25] desenvolvida utilizando a biblioteca *Open Source Computer Vision* (OpenCV) para representação e manipulação de imagens e vídeos. Para que ocorra o rastreamento de pontos exige-se que a imagem de entrada esteja no formato *grayscale* (níveis de cinza). Caso não esteja, deve-se primeiramente transformar a imagem de entrada para este formato.

Para rastrear pontos da face a SDK implementa a técnica RMLS introduzida na fundamentação. O Algoritmo 1 descreve de forma simplificada os passos tomados para se obter o rastreamento [16]. A variável $\mathbf{p}$ no algoritmo representa os parâmetros do PDM e com ela calculada pode-se obter a posição dos pontos de interesse através da Equação 2.1. Um detalhamento da técnica pode ser encontrado em [16].

Algoritmo 1 RLMS (*Regularized landmark mean-shift*)

Require: $\mathcal{I}$ e $\mathbf{p}$ ▷ $\mathcal{I}$ sendo a imagem e $\mathbf{p}$ como definido na Equação 2.17

1: Computar respostas [Equações 2.17]
2: **while** nao_convergiu($\mathbf{p}$) **do**
3: Linearizar o modelo de forma [Equação 2.2]
4: Computar os vetores do deslocamento da média [Equação 2.24]
5: Computar a atualização dos parâmetros PDM [Equação 2.25]
6: Atualizar parâmetros: $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$
7: **end while**
8: **return p**

O resultado do rastreamento é um arranjo de 66 pontos contendo os valores das coordenadas (x, y) na imagem de cada ponto chave. A SDK ordena os pontos retornados de forma que cada ponto chave esteja sempre em uma mesma posição do arranjo. Os pontos rastreados e a numeração associada a cada um deles são mostrados na Figura 3.3.

Além dos pontos retornados, um valor entre zero e dez representando a qualidade da detecção é fornecido ao final de cada rastreamento. Esse valor serve para avaliar se a detecção do quadro atual deve ser utilizada, para assim atualizar a malha tridimensional: caso o valor da qualidade de detecção seja zero em uma iteração, o modelo não é atualizado. Além disso, uma qualidade de zero indica que o rastreamento perdeu completamente os pontos de interesse e é necessário reinicializar o rastreador para que a detecção tenha chance de ser realizada com sucesso na próxima iteração.

Além disso, como o rastreamento é aplicado independentemente a cada uma das imagens de entrada, é importante que se mantenham dois rastreadores inicializados e atualizados independentemente. Isso deve ser feito pois a SDK reutiliza informações de estados anteriores para acelerar o rastreamento no quadro atual e não se deve misturar o estado do rastreamento na imagem esquerda

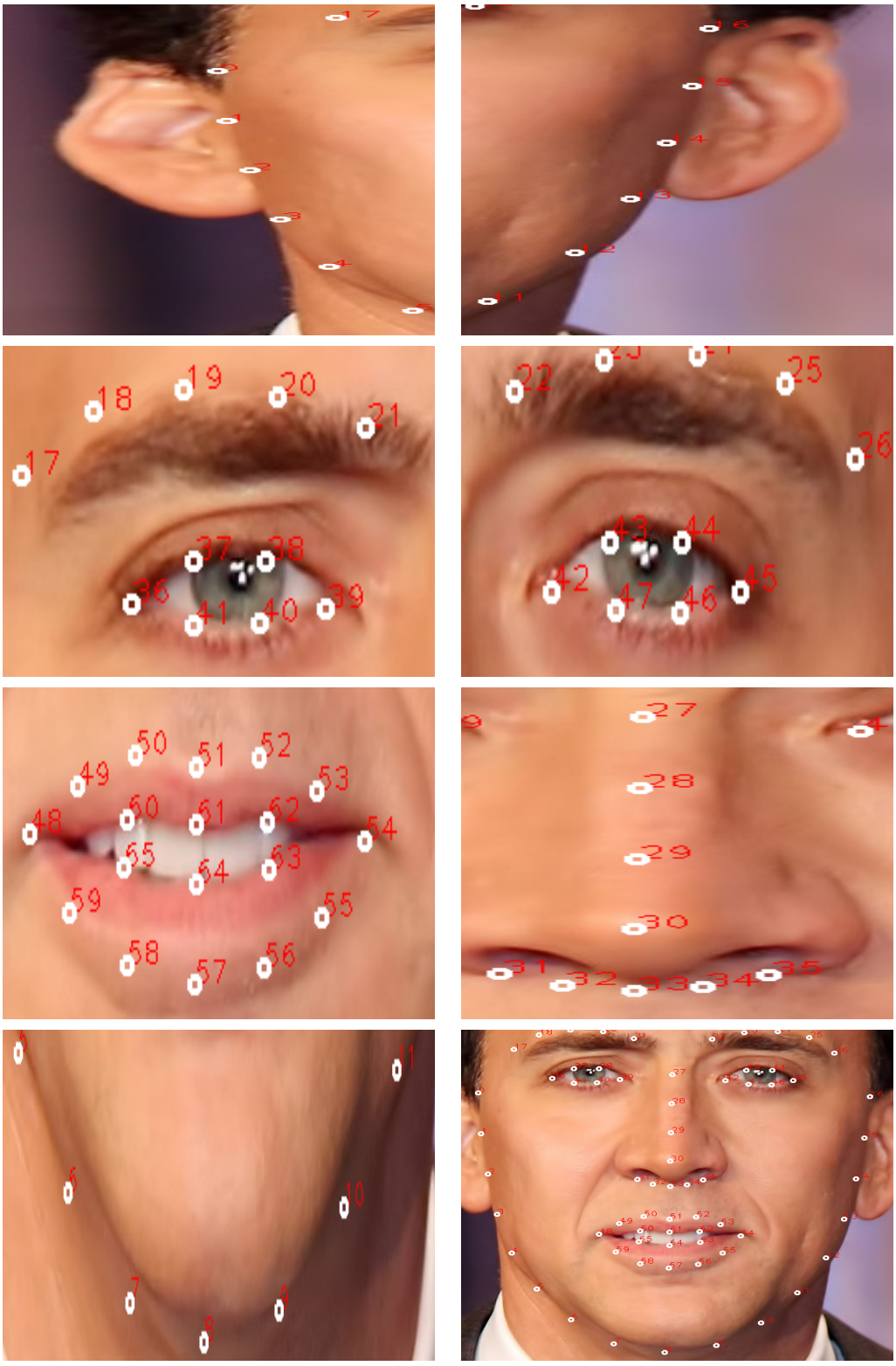


Figura 3.3: Pontos retornados pelo rastreamento facial e a numeração associada a cada ponto. A imagem sobre a qual se realiza o rastreamento foi retirada de [8].

com o estado do rastreamento na imagem direita.

A saída desta etapa como um todo consiste de dois conjuntos de 66 pontos bidimensionais enumerados, uma para cada imagem, bem como dois parâmetros que representam a qualidade do rastreamento feito em cada imagem.

3.1.3 Estimação da Profundidade

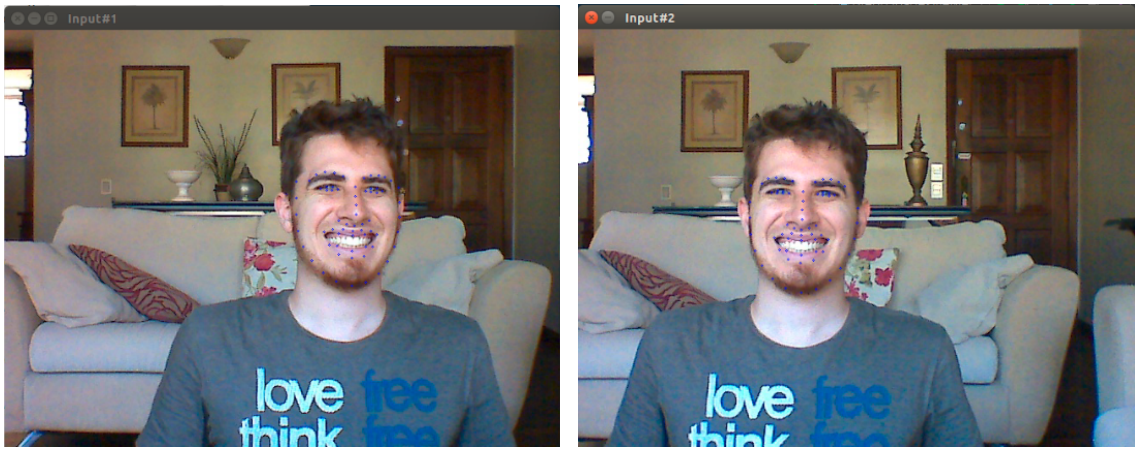
Como ficará colocado em mais detalhes nas próxima seções, esta aplicação utiliza a distância entre pontos do rosto para interpretar a posição que deve ser transferida para o modelo. Sem a estimação de profundidade, a distância entre dois pontos do rosto diminuiria a medida que o usuário se afastasse do par de câmeras e colocaria sobre o usuário a responsabilidade de realizar gravações sempre a uma mesma distância. Além disso, uma leve movimentação despercebida durante as gravações poderia ser erroneamente interpretada pelo programa, já que decisões são tomadas a partir do movimento entre os pontos e um afastamento do usuário causaria uma redução em todas as distâncias. A estimação da profundidade permite que medidas retiradas da imagem sejam independentes da distância em que o usuário se encontra das câmeras, o que garante maior flexibilidade de uso e uma maior estabilidade na animação. Com a estimação da profundidade é possível aproximar os valores X, Y e Z dos pontos de interesse no sistema de coordenadas do mundo. Por exemplo, a distância entre os dois olhos do usuário deve ser sempre a mesma e pode ser medida em centímetros e não mais em pixeis.

A estimação da profundidade é feita utilizando-se os dois conjuntos de pontos fornecidos pela etapa de rastreamento. Um exemplo de um par de imagens de entrada onde o rastreamento foi aplicado e os pontos marcados pode ser observado na Figura 3.4. As imagens tiradas em uma mesma iteração do programa por câmeras diferentes mostram o usuário em uma posição ligeiramente deslocada em relação a outra, e é justamente essa diferença horizontal que é utilizada para estimar a profundidade. Além disso, uma vez estimada a profundidade, pode-se corrigir a posição horizontal dos pontos de interesse[1].

Como o cojunto de pontos rastreados é sempre ordenado da mesma maneira, é fácil localizar o mesmo ponto do rosto em cada uma das imagens: basta tomar os mesmos índices em cada um dos vetores retornados pela etapa de rastreamento. Esse mapeamento de pontos é utilizado para estimar a posição no mundo de cada um dos pontos rastreados, isso é feito através de uma semelhança de triângulos, como foi resumido na Equação 2.41 apresentada na fundamentação teórica.

Vale notar que para a aplicação da Equação 2.41 deve-se anteriormente conhecer a distância focal de cada uma das câmeras, o que pode ser obtido pela utilização da ferramenta *Calibration Toolbox*, uma extensão disponível para Matlab. O processo de calibração de uma câmera por meio dessa ferramenta consiste na captura de vinte imagens de um tabuleiro de xadrez em diferentes posições. Após a marcação manual das extremidades desse tabuleiro, o algoritmo de calibração

[1] Vale lembrar que assume-se que os centros de captura das câmeras estão alinhados verticalmente e, portanto, não espera-se diferença entre as coordenadas Y de um mesmo ponto do rosto nas duas imagens.



(a) (b)

Figura 3.4: Imagens extraídas da câmera 1 (a) e da câmera 2 (b) em uma mesma iteração do programa em execução. Ao comparar as duas imagens, nota-se que o usuário aparece deslocado horizontalmente em uma imagem quando comparado a outra. Os pontos de interesse foram rastreados e marcados com círculos azuis.

realiza um processo de detecção em cada uma das imagens de um conjunto de pontos conhecidos - as quinas de cada um dos quadrados do tabuleiro de xadrez . O mapeamento desses pontos em cada uma das imagens em conjunto com o conhecimento do tamanho real de cada uma das casas do tabuleiro permite determinar a matriz de calibração da câmera, que inclui, dentre outros valores, os parâmetros intrínsecos da câmera. Um exemplo do conjunto de imagens utilizado durante a etapa de calibração pode ser observado na Figura 3.5. A calibração[2] é feita após o processo de fixação das câmeras e os resultados guardados para utilização enquanto os resultados da estimação de profundidade se mostrarem satisfatórios. Vale notar que choques ocasionais aplicados a montagem podem alterar os parâmetros das câmeras e requerer uma recalibração destes.

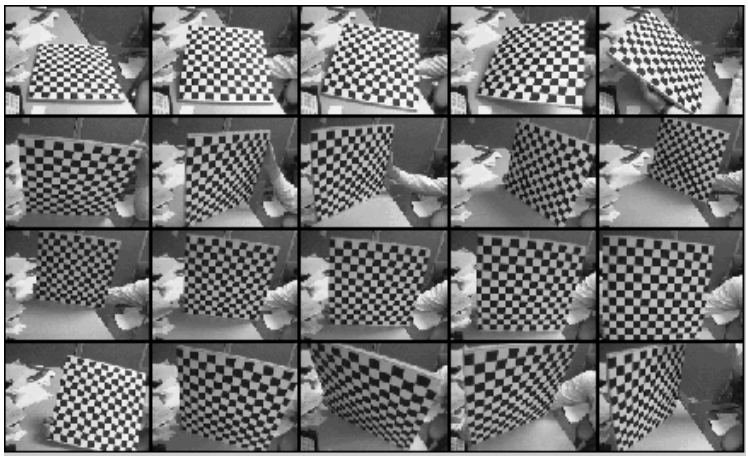


Figura 3.5: Exemplo de imagens utilizadas em uma calibração (retirado de [9]).

[2]O termo 'calibração' aqui significa a realização de um processo que estima os parâmetros de interesse. Nenhuma modificação é aplicada sobre os sensores em si.

A matriz KK dos parâmetros intrínsecos da câmera obtida por uma calibração feita utilizando a *Calibration Toolbox for Matlab* é mostrada na Equação 3.1.

$$KK = \begin{bmatrix} f_c(1) & alpha_c * f_c(1) & cc(1) \\ 0 & f_c(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Onde $f_c(1)$ e $f_c(2)$ são as distâncias focais descritas em unidades de pixeis horizontais e verticais, o coeficiente $alpha_c$ é o ângulo entre os eixos de coordenadas x e y e os valores de $cc(1)$ e $cc(2)$ representam as coordenadas x e y do ponto principal da câmera.

Caso seja considerado um pixel quadrado, ou seja, $alpha_c = 0$ e $f_c(1) = f_c(2)$, essa matriz intrínseca será igual a matriz definida na Equação 2.31.

De posse de todas as variáveis é então possível estimar o valor Z de um ponto no mundo e com isso estimar também os valores X e Y deste mesmo ponto. O cálculo dos valores X e Y é feito por meio das Equações 2.37 e 2.38 e das Equações 2.39 e 2.40, respectivamente.

Sumarizando, a etapa de estimação de profundidade utiliza os **dois** conjuntos de pontos **bi-dimensionais** provenientes da etapa de rastreamento para produzir **um** conjunto de sessenta e seis pontos **tridimensionais** contendo cada um os valores (X, Y, Z) referentes às coordenadas de cada ponto rastreado no sistema de coordenadas do mundo. Além de adicionar a componente de profundidade, esta etapa retorna os valores em centímetros das componentes dos elementos do conjunto produzido.

3.1.4 Atualização dos Pesos de Mistura - Razão de Distância

Até este ponto a aplicação obteve uma estimação para as coordenadas espaciais de um conjunto de pontos de interesse da face humana. Passa-se agora para a etapa de extração de informação de pose a partir desse conjunto de pontos. Como será detalhado na próxima seção, isso é feito por meio da técnica de mistura de poses que requer como entrada um vetor de pesos. O objetivo desta etapa pode ser posto como inferir um vetor de pesos que adequadamente produzirá uma mistura de poses cujo resultado se assemelha com a expressão facial na imagem de entrada.

Define-se cada uma das componentes do vetor de peso independentemente como uma **razão de distâncias** entre dois pontos do conjunto de pontos fornecidos pela etapa de rastreamento. A razão é tomada entre a distância entre os pontos medida no quadro considerado e a distância entre esses mesmos pontos em quadros de uma etapa de calibração. Como o interesse é a variação percentual, o que é medido é de fato a fração que a distância atual se encontra entre a distância máxima e a mínima medida para o par de pontos.

Seja o peso w_i^j associado com o par de pontos $(\boldsymbol{p_1^i}, \boldsymbol{p_2^i})$ no quadro j e seja $d_i^j = |\boldsymbol{p_1^i} - \boldsymbol{p_2^i}|$ a distância medida entre esse par de pontos nesse quadro, o peso da i-ésima pose é dado por:

$$w_i^j = \frac{|d_i^j - d_i^{\min}|}{|d_i^{\max} - d_i^{\min}|} \tag{3.2}$$

onde $d_i^{\min}$ e $d_i^{\max}$ são as distâncias mínimas e máximas respectivamente para o par de pontos associado ao peso w_i obtidas em uma etapa de calibração.

A Equação 3.2 descreve poses onde um aumento na razão de distância significa um aumento no peso da pose associada. No caso de poses onde o oposto deve acontecer, ou seja, uma diminuição da razão de distância significa um aumento do peso da pose associada, a equação de peso é dada por:

$$w_i^j = 1 - \frac{|d_i^j - d_i^{\min}|}{|d_i^{\max} - d_i^{\min}|} \tag{3.3}$$

A Equação 3.3 é utilizada, por exemplo, para o peso associado a pose que fecha um dos olhos. Nesta pose, quanto maior o peso associado, mais fechado o olho se encontra. Por outro lado, quando mais fechado o olho, menor a distância vertical entre pontos ao longo das pálpebras de cima e de baixo desse olho.

Nota-se que as equações apresentadas descrevem cada peso como um valor entre 0 e 1. No entanto, o modelo de poses utilizado requer que a soma dos pesos resulte em 1. Para garantir essa condição, o peso da pose neutra é escolhido como:

$$w_{\text{neutra}}^j = 1 - \sum_i w_i^j \tag{3.4}$$

onde i varia no intervalo de índices das poses de ação (as poses excluindo a pose neutra).

Resumindo, para cada pose disponível na mistura de poses escolhe-se dois pontos do conjunto de pontos da etapa de estimação de profundidade e define-se o peso da pose associada a partir da comparação entre a distância entre esses dois pontos no quadro atual e a a distância entre esses dois pontos em quadros de uma etapa de calibração. Os quadros da etapa de calibração são dois: um correspondente a 0% da pose e outro correspondendo a 100% da aplicação da pose.

O par de pontos utilizados para cada uma das poses disponíveis para mistura são listados na Tabela 3.1. Para cada ponto apresenta-se o índice correspondente no vetor de 66 pontos da etapa de rastreamento, bem como uma descrição de sua localização no rosto. O mapeamento entre o índice de pose e a sua descrição é apresentado na Tabela 3.2.

3.1.5 Filtros

Os pesos de mistura calculados anteriormente não são utilizados diretamente no processo de mistura de poses, antes disso eles são filtrados. O objetivo dos filtros digitais é remover componentes de altas frequências associadas a ruídos presentes no processo de captura de imagem e na própria detecção.

k	i_k	descrição de p_{i_k}	j_k	descrição de p_{j_k}
1	54	canto esquerdo da boca	48	canto direito da boca
2	24	centro da sobrancelha esquerda	42	canto direito do olho esquerdo
3	19	centro da sobrancelha direita	39	canto esquerdo do olho direito
4	43	canto superior do olho esquerdo	47	canto inferior do olho esquerdo
5	37	canto superior do olho direito	41	canto inferior do olho direito
6	61	centro inferior do lábio superior	64	centro superior do lábio inferior

Tabela 3.1: Pares de pontos utilizados para razão de distâncias. O peso w_k está associado com o par (p_{i_k}, p_{j_k}). Os índices dos pontos podem ser comparados com os índices da Figura 3.3

k	pose k
1	sorrir
2	levantar da sobrancelha esquerda
3	levantar da sobrancelha direita
4	fechar do olho esquerdo
5	fechar do olho direito
6	abrir da boca

Tabela 3.2: Mapeamento do índice da pose no vetor de pesos com ação realizada pela aplicação da pose.

Os filtros digitais projetados para suavizar a detecção foram filtros FIR passa-baixas, implementados no domínio do tempo por meio de equações de diferenças. Para a implementação dos filtros é necessário que se mantenha em memória os valores das entradas em instantes de tempo anteriores. Uma primeira implementação consiste em mover cada uma das entradas para a próxima posição do vetor de entradas anteriores em cada atualização do filtro. Para evitar esse laço de deslocamento, utiliza-se nesta aplicação a técnica de buffer circular que mantém um índice para a posição mais recente e atualiza esse índice ao invés de atualizar a posição dos dados em si.

Para o projeto dos coeficientes do filtro, duas técnicas foram utilizadas: filtros de média móvel e projeto por janela. Para este último, a janela de Hamming foi utilizada para realizar cortes sobre o filtro de resposta infinita que geraria a resposta ideal. O motivo de se projetar vários filtros é permitir que diferentes bandas sejam filtradas em cada um dos pesos de mistura. Isso se faz necessário pois, por exemplo, enquanto altas frequências representam ruídos na detecção da ação de sorrir, elas podem ser necessárias para capturar movimentos rápidos como o piscar de olhos. Durante a execução do programa, cursores de seleção permitem selecionar o filtro utilizado para cada um dos pesos de mistura.

Os gráficos na Figura 3.6 mostram a resposta em frequência de cada filtro. O Filtro de Média Móvel de Hanning é descrito na Equação 3.5.

$$y(n) = \frac{1}{4}(1x(n) + 2x(n-1) + x(n-2)) \tag{3.5}$$

k w_c	0.1π	0.1π	0.1π
0	0,0025	-0,0000	0,0030
1	0,0057	-0,0052	-0,0050
2	0,0147	0,0000	0,0067
3	0,0315	0,0232	-0,0000
4	0,0555	-0,0000	-0,0252
5	0,0834	-0,0761	0,0721
6	0,1099	0,0000	-0,1306
7	0,1289	0,3077	0,1801
8	0,1358	0,5009	0,7979
9	0,1289	0,3077	0,1801
10	0,1099	0,0000	-0,1306
11	0,0834	-0,0761	0,0721
12	0,0555	-0,0000	-0,0252
13	0,0315	0,0232	-0,0000
14	0,0147	0,0000	0,0067
15	0,0057	-0,0052	-0,0050
16	0,0025	-0,0000	0,0030

Tabela 3.3: Coeficientes para os filtros projetados com técnica de janela, utilizando a janela de Hamming.

Os filtros projetados com a técnica da janela seguem a seguinte equação:

$$y(n) = \sum_{k=0}^{L} b_k x(n-k) \tag{3.6}$$

Para L=16, a Tabela 3.3 mostra os coeficientes para alguns dos filtros projetados pela técnica de corte com janela de Hamming.

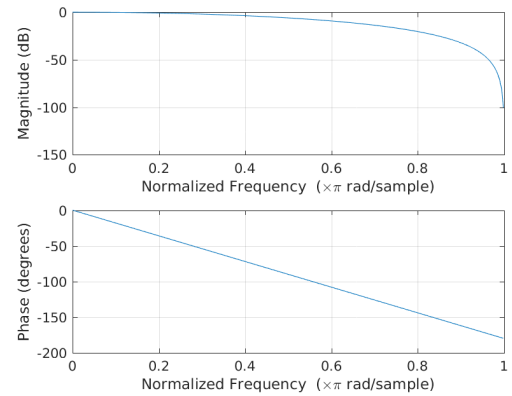
3.1.6 Mistura de Poses

O conceito básico por trás da Mistura de Poses está em criar poses intermediárias a partir da interpolação linear de poses pré-definidas.

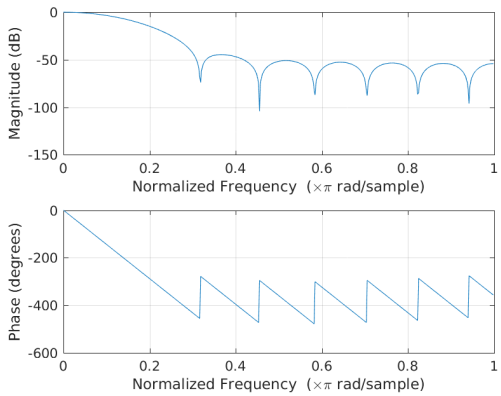
A pose base utilizada para a mistura de poses deve ser uma onde todas as razões de distância estão no mínimo, ou seja, deve ser uma pose onde a face esteja em repouso. O modelo utilizado para a pose neutra é mostrada na Figura 3.7.

Aos demais modelos renderizados é atribuído um peso de mistura definido como a saída de um filtro que recebe como entrada uma sequência de valores de uma razão de distância.

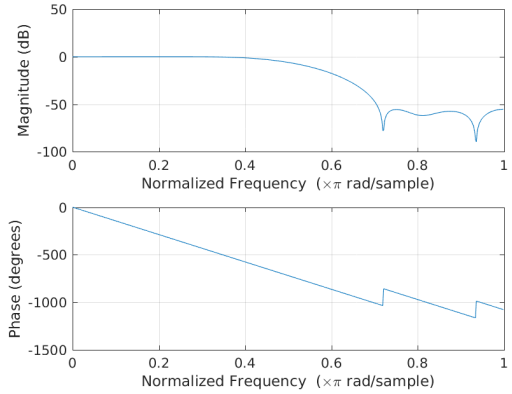
Alguns detalhes são necessários para que a mistura de poses ocorra com sucesso. Dentre eles, é necessário que os pontos na malha do modelo apareçam enumerados na mesma ordem em todas



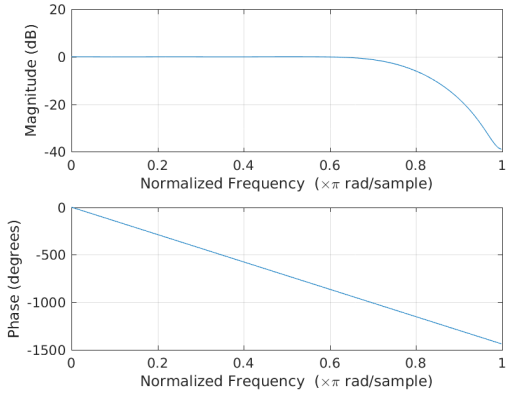
(a) Filtro de Média móvel de Hanning



(b) Projeto com janela de Hamming $w_c = \frac{\pi}{10}$



(c) Projeto com janela de Hamming $w_c = \frac{\pi}{2}$



(d) Projeto com janela de Hamming $w_c = \frac{4\pi}{5}$

Figura 3.6: Magnitude e fase da resposta em frequência para alguns dos filtros utilizados na aplicação.

as poses utilizadas para a mistura. Isso pode ser garantido requerendo-se do software que exporta os modelos para formato de arquivo OBJ que mantenha os pontos na mesma ordem. Essa é uma opção comum nos softwares de modelagem, mas que nem sempre é habilitada por padrão e o requisito de manutenção da ordem dos pontos deve ser informado ao artista responsável pela produção dos modelos para que ele configure o software adequadamente. Outro requisito é que os modelos utilizem um mesmo sistema de coordenas e que os triângulos da malha utilizem uma mesma indexação de VBOs. O primeiro requisito não apresentou problemas na realização deste trabalho, sendo suficiente centrar o objeto em (0,0,0), porém o segundo requisito apresentou problemas. Enquanto o número e a ordem na listagem de triângulos se mantém o mesmo, verificou-se que os softwares de modelagem nem sempre mantêm a mesma ordem da listagem dos pontos de um dado triângulo. Ou seja, os pontos de um triângulo são sempre os mesmos em dois arquivos OBJ, mas não necessariamente aparecem listados na mesma ordem, o que causa problemas durante o processo de mistura. Para resolver este problema, a construção da malha por meio de triângulos para todos os modelos utilizados é feita com a listagem de triângulos da malha da pose neutra. Ou seja, os triângulos são definidos pela pose neutra e carrega-se dos OBJ de cada uma das outras poses apenas o posicionamento dos pontos da malha.

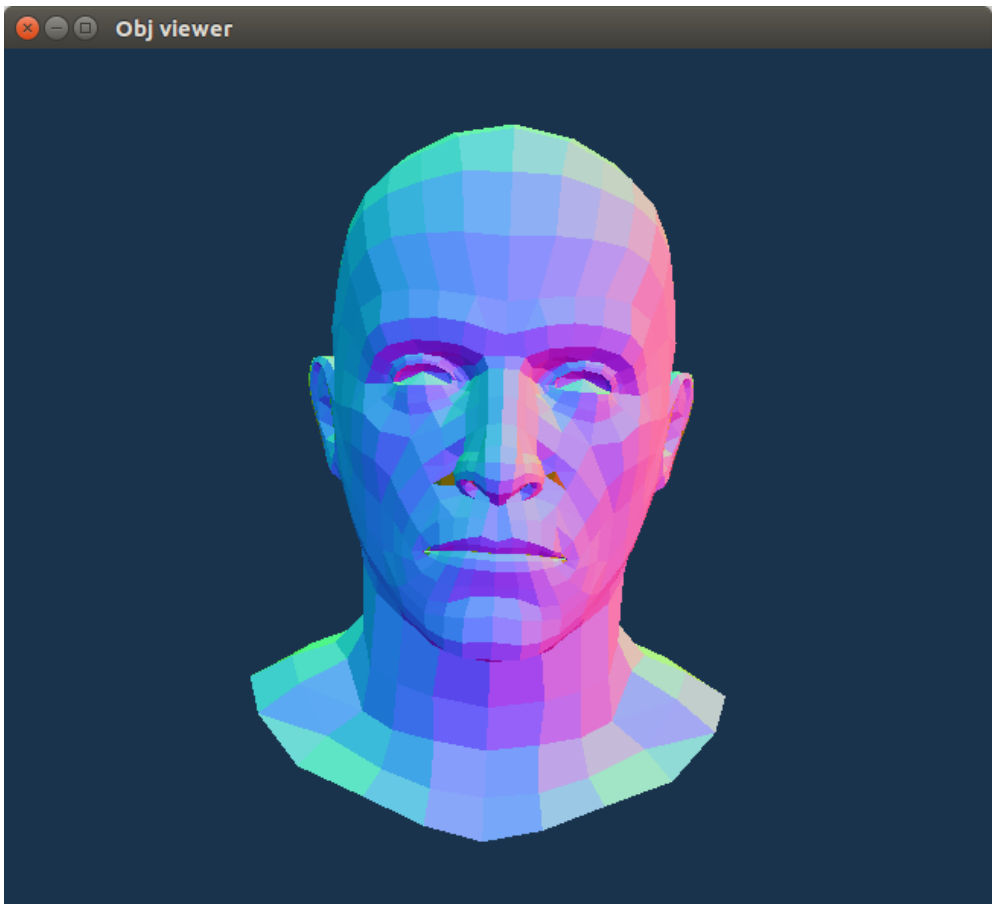


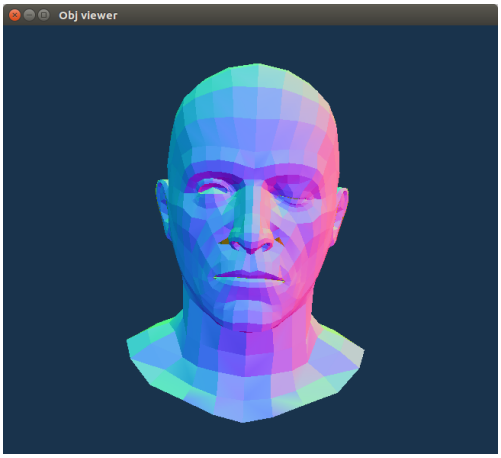
Figura 3.7: Pose base (neutra).

Caso os arquivos sejam carregados adequadamente e a mistura de poses seja realizada com sucesso é possível verificar uma transição natural entre o modelo da Pose Neutra para qualquer outro modelo pré-definido ao fazer a mistura de poses utilizando-se somente a pose neutra e uma das outras poses.

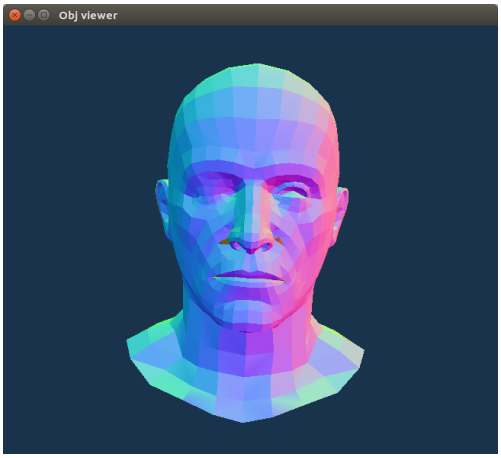
Como o modelo final renderizado é uma interpolação dos pontos de todos os modelos após a aplicação dos pesos calculados, esse modelo será igual a uma pose pré-definida apenas se a razão de distância referente a ela for máxima, enquanto todas as outras forem mínimas. Caso contrário, se o rosto não está em repouso, o modelo final sempre será igual a uma pose intermediária. As poses pré-definidas utilizadas neste trabalho são mostradas na Figura 3.8 e na Figura 3.9.

A Equação 2.51 descreve como as poses pré-definidas podem ser combinadas com a pose neutra de forma a obter uma quantidade imensa de poses intermediárias.

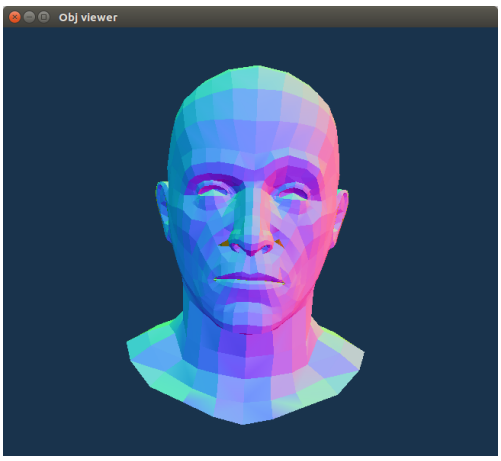
Conectando as etapas apresentas até agora dentro de um laço de execução, os pesos de mistura são atualizados iterativamente de acordo com o movimento dos pontos rastreados, produzindo mudanças suaves na forma do modelo resultante renderizado. A sequência de configurações pelas quais passa o modelo renderizado a medida que as imagens são capturadas e processadas resultam na animação computacional que é o objetivo da aplicação.



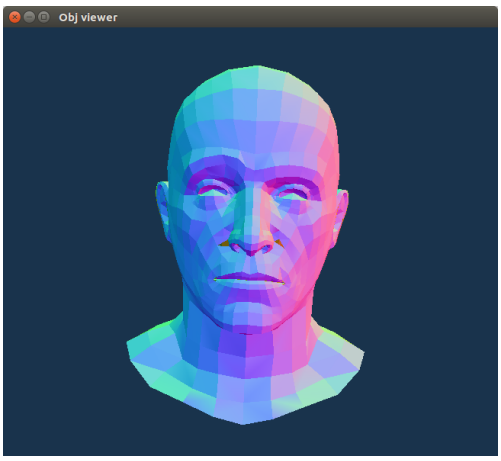
(a) Pose Olho Esquerdo



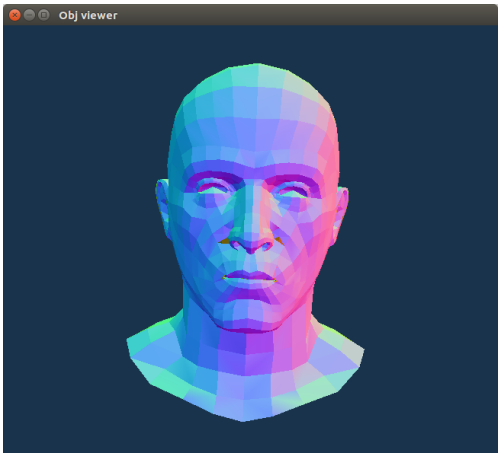
(b) Pose Olho Direito



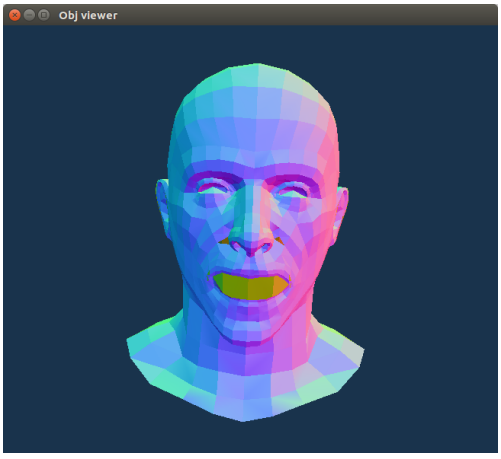
(c) Pose Sobrancelha Esquerda



(d) Pose Sobrancelha Direita

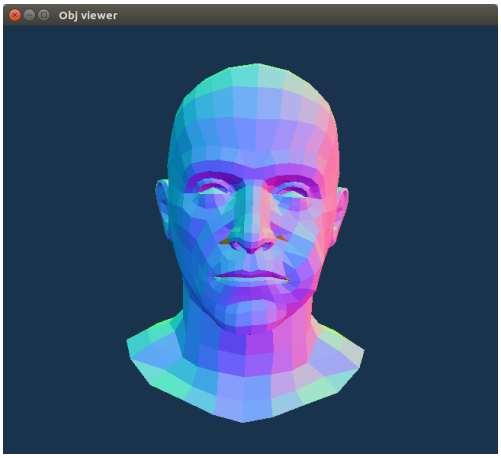


(e) Pose Boca Fechada

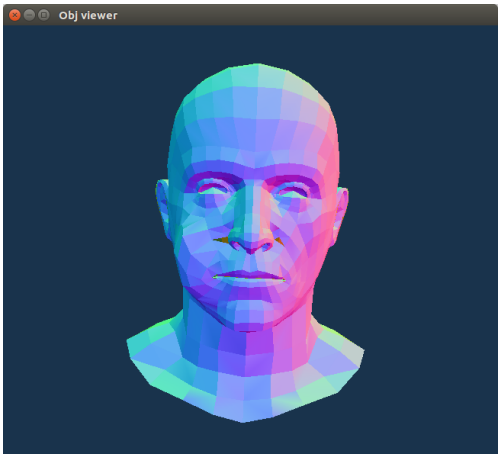


(f) Pose Boca Aberta

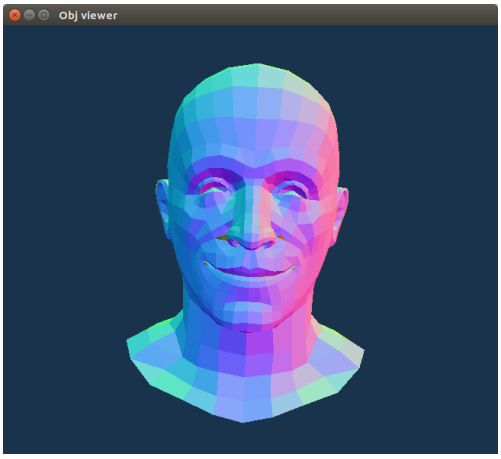
Figura 3.8: Exemplos de modelos usados como poses pré-definidas.



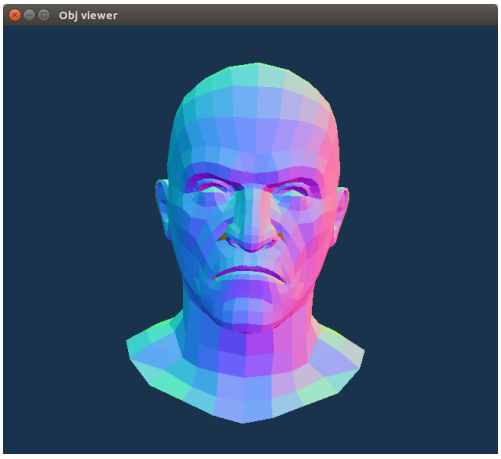
(a) Pose Bochecha Esquerda



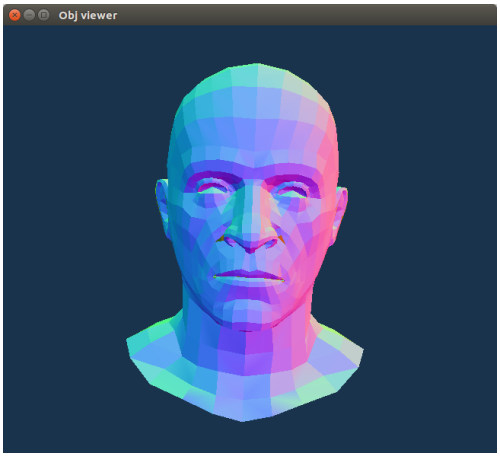
(b) Pose Bochecha Direita



(c) Pose Sorrindo



(d) Pose Bravo



(e) Pose Nariz Aberto

Figura 3.9: Exemplos de modelos usados como poses pré-definidas.

3.2 Experimentos de Validação dos Métodos Utilizados

Como não se dispõe de um método para medir a qualidade final dos resultados de modo quantitativo, propõem-se alguns experimentos com o objetivo de validar as etapas do processo. Os experimentos visam revelar as limitações atuais da técnica ao medir a precisão de cada etapa.

Uma das dificuldades enfrentadas no experimento é que a aplicação envolve o rastreamento de um rosto humano e não é encontrada uma boa maneira de substituir o alvo humano por um objeto que possa ser facilmente fixado e ter a posição controlada. Com isso, pessoas são utilizadas como parte do experimento e certamente há erros introduzidos nas medidas. Para exemplificar, em um dos experimentos propostos o objetivo é tirar várias fotos do usuário há uma distância fixa, no entanto, é difícil garantir que o alvo não tenha se movido entre uma foto e outra. Há ainda o problema da iluminação que pode variar ligeiramente a medida que as horas de experimentos passam. Tendo isso em mente, o objetivo dos experimentos é que eles forneçam uma ideia das limitações da técnica e não que sejam uma metodologia precisa de estimação dos erros envolvidos.

Tendo em vista que o controle do ambiente e da iluminação não são necessários para o funcionamento adequado do sistema de animação proposto, não especificamos restrições em relação a estes fatores para os experimentos. Em relação ao hardware utilizado, o sistema foi instalado e testado em dois computadores. As configurações das máquinas utilizadas são apresentadas a seguir:

- Computador 1: Intel® Core™ i7-2630QM CPU @ 2.00GHz × 8; 6GB RAM; GeForce GT 540M/PCIe/SSE2; Ubuntu 14.04; OpenGL 3.3.
- Computador 2: Intel® Core™ i7-4510U CPU @ 2.00GHz × 4; 16GB RAM; Sem placa de vídeo dedicada; Ubuntu 16.04; OpenGL 3.3.

Vale notar que as máquinas utilizadas apresentam configurações comumente encontradas na data atual de publicação deste trabalho, mostrando que o sistema não necessita de máquinas com um hardware muito sofisticado para ter um bom desempenho. Esta última observação está alinhada com o princípio de projeto de desenvolver um sistema de baixo custo.

A Figura 3.10 mostra a montagem utilizada nos experimentos que se seguem. Antes de cada medida, os passos a seguir são tomados como uma configuração inicial:

1. O par de câmeras é posicionado na altura do rosto alvo. Seja a direção da linha ligando as câmeras chamada de $\vec{A}$.
2. A ponta da trena horizontal é posta no ponto médio entre as câmeras e é puxada em direção $\vec{B}$ perpendicular a $\vec{A}$ e no sentido apontando para o usuário.
3. O usuário segura a trena e se afasta do par de câmeras se movendo na direção $\vec{B}$ até atingir a distância desejada.
4. Nesse ponto uma segunda régua (segmento vermelho na Figura 3.10) é posta verticalmente e ortogonalmente sobre a trena e o usuário encosta o nariz na ponta da régua vertical.

5. Esta é a posição que o ator deve manter durante o resto do experimento o mais fielmente possível.

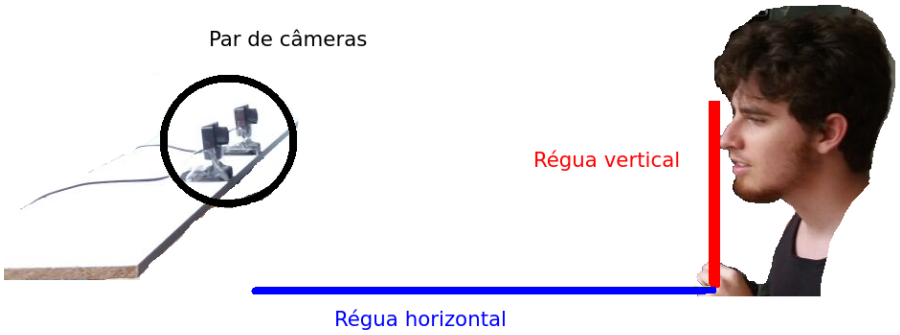


Figura 3.10: Montagem experimental utilizada para captura repetitiva de fotos do mesmo alvo em várias posições controladas.

3.2.1 Rastreamento de Pontos da Face e Estimação do Tridimensional

O Rastreamento de Pontos da Face retorna o valor das coordenadas dos pontos em pixeis, fazendo com que as distâncias entre pares de pontos associados a uma pose, quando avaliadas diretamente na imagem, mudem à medida que o usuário se afasta ou se aproxima do sistema de captura. Por outro lado, a técnica de estimação de profundidade almeja garantir que essas distâncias, agora medidas em centímetros, se mantenham as mesmas se as distâncias assim se mantiverem no objeto real. Para mostrar que a recuperação da profundidade está sendo feita de forma adequada, propõe-se um experimento que irá medir a distância em centímetros de um par de pontos do rosto, à medida que o usuário se afasta do par de câmeras. Se o processo for feito como esperado, a distância entre os pontos do rosto será sempre a mesma, independentemente das distância do alvo às câmeras. Outro ponto a ser notado é que para o rastreamento ser considerado preciso, as distâncias medidas não podem variar no caso de o face alvo estar imóvel. Para verificar isso, propõe-se também medir o desvio padrão na sequência de medidas de distância entre dois pontos de uma face imóvel em uma série de capturas.

Para ambos os experimentos o par de pontos escolhidos para se medir em cada quadro é composto pelos pontos laterais extremos dos lábios (pontos 48 e 54 na Figura 3.3). As mesmas imagens capturadas foram utilizadas em ambos os experimentos.

Para a realização dos experimentos, os seguintes passos foram tomados:

1. A montagem inicial é executada posicionando o nariz do alvo a 40 centímetros de distância.

2. Capturam-se 30 quadros de cada câmera enquanto o alvo se mantém o mais estático possível.

3. Repete-se o processo afastando-se o alvo do par de câmeras 5 centímetros em cada etapa, até que se atinja a distância de 80 cm do par de câmeras.

Após a captura, as imagens são entradas aos pares em um programa que:

1. Aplica o algoritmo de rastreamento dos 66 pontos do rosto no par de imagens.

2. Mede a distância entre os pontos 48 e 54 em pixeis para cada câmera.

3. Aplica a técnica de estimação de profundidade para corrigir as coordenadas (x,y) de cada ponto detectado

4. Mede a distância entre os pontos 48 e 54 em centímetros.

Para cada par de quadros, os seguintes valores são salvos em arquivo: a distância entre os pontos 48 e 54 em pixeis da imagem capturada com a câmera 1, a distância entre os pontos 48 e 54 em pixeis da imagem capturada com a câmera 2 e a distância entre os pontos 48 e 54 estimada em centímetros.

3.2.2 Filtragem Digital

Neste experimento o objetivo é comparar a sequência de coeficientes de mistura filtrada e não-filtrada. Para cada um dos pesos de mistura grava-se um par de vídeos, um para cada câmera, onde o ator centrado na tela movimenta intencionalmente os músculos do rosto associados com o peso de mistura que se quer examinar.

Apenas um vídeo é gravado para cada pose examinada. Todos os vídeos são gravados a uma mesma distância da câmera e sobre condição de iluminação semelhante. No caso de poses simétricas, como o fechar do olho esquerdo e do direito, apenas o movimento do lado esquerdo do rosto é examinado. Para cada movimento avaliado, os vídeos gravados são colocados em um programa que aplica o rastreamento visual dos pontos do rosto e mede apenas a razão de distância considerada (por exemplo, no vídeo que examina o olho, mede-se somente a abertura dos olhos e no vídeo que examina o sorrir, apenas o comprimento horizontal dos lábios é medido).

A razão obtida é entrada independentemente em cada um dos filtros projetados. A razão medida e a saída de cada um dos filtros é guardada em arquivo para análise posterior.

O próximo capítulo apresenta os resultados obtidos neste trabalho para cada etapa intermediária e para o sistema de animação completo em funcionamento.

Capítulo 4

Resultados

Este capítulo apresenta os resultados obtidos nos experimentos feitos para validação das técnicas utilizadas neste trabalho, bem como o resultado final da aplicação. Buscou-se repetir os experimentos várias vezes (mais do que 30 vezes) a fim de garantir valor estatístico para os resultados apresentados.

4.1 Calibração das Câmeras

A Tabela 4.1 mostra os parâmetros intrínsecos das câmeras utilizadas, obtidos a partir da ferramenta *Camera Calibration Toolbox for Matlab*.

Parâmetros intrínsecos	Câmera 1 (pixel)	Câmera 2 (pixel)
$f_c(1)$	854 ± 4	834 ± 6
$f_c(2)$	858 ± 4	836 ± 6
$cc(1)$	298 ± 6	300 ± 7
$cc(2)$	232 ± 6	217 ± 7
$alpha_c$	0	0

Tabela 4.1: Parâmetros intrínsecos da câmera medidos em pixeis.

Como pode ser observado os valores de $f_c(1)$ e $f_c(2)$, referentes a distância focal medida em pixeis horizontais e verticais, são bem próximos para ambas as câmeras. Essa verificação valida a aproximação assumida, durante a modelagem, de que câmera possui um pixel quadrado. Portanto, a distância focal utilizada, para efeitos de cálculo, é dada por $f = (f_c(1) + f_c(2))/2$. Os valores são dados em pixeis.

Os resultados similares obtidos para cada uma das câmeras não é ao acaso, visto que as duas câmeras utilizadas são da mesma marca e modelo. Além disso, ambas possuem foco ajustável, o ajuste em cada uma foi feito manualmente e por avaliação visual, de modo a capturar imagens nítidas.

Os parâmetros calculados fornecem os elementos necessários para tornar possível a estimação

da profundidade de um ponto alvo, a partir da Equação 2.41. Assim, pode-se estimar o valor de X em centímetros, pelas Equações 2.37 e 2.38, bem como estimar o valor de Y em centímetros, pelas Equações 2.39 e 2.40.

4.2 Rastreamento de Pontos da Face

Os experimentos realizados nesta seção foram feitos com o objetivo de avaliar isoladamente a precisão da detecção de pontos da face utilizando a SDK *CSIRO Face Analysis*, ou seja, as medidas são tomadas com os valores puros (em pixeis e não filtrados) obtidos pelo rastreamento. O objetivo desta análise é permitir a comparação com os resultados obtidos após filtragem e estimação de tridimensionalidade, bem como justificar o uso destas técnicas.

Escolheu-se medir a razão de distância para o **comprimento dos lábios** em repetidas tomadas, para duas das poses disponíveis: a Pose Neutra e a Pose Sorrindo. Essa razão de distância, após filtragem, compõe o peso de mistura para a Pose Sorrindo na aplicação final e foi escolhida por ser a razão que apresenta maior amplitude de variação entre os extremos máximos e mínimos.

Para ambas câmeras 1 e 2 obteve-se a média e o desvio padrão, em pixeis, para essa razão de distância. Esses dados foram adquiridos a medida que o usuário se posicionava a diferentes distâncias em relação à montagem de captura, variando, assim, os valores de profundidade. Os dados de média e desvio padrão foram obtidos a partir de medições em trinta imagens de uma face imóvel, capturadas sucessivamente com intervalos de tempo menores que um segundo.

A Tabela 4.2 exibe os valores obtidos através da câmera 1 enquanto a Figura 4.1 mostra um gráfico da variação do desvio padrão do comprimento dos lábios, para a Pose Neutra e a para Sorrindo, em relação a distância entre a câmera 1 e o alvo.

	Câmera 1 - Pose Neutra (pixel)		Câmera 1 - Pose Sorrindo (pixel)	
Distância da câmera (cm)	Média	Desvio padrão	Média	Desvio padrão
40	93,4096	0,5575	92,1488	1,3473
45	83,7242	0,6406	84,694	0,8848
50	75,6548	0,7163	78,1044	1,0881
55	67,9294	0,6772	69,1071	0,9671
60	65,1105	0,3884	65,8582	0,8114
65	58,8054	0,3392	59,7229	0,5219
70	54,7137	0,5925	55,7597	0,6367
75	53,4555	0,4781	53,7684	0,6277
80	49,1185	0,3156	49,7219	0,2373

Tabela 4.2: Comprimento horizontal dos lábios medido em pixeis a partir de imagens capturadas do rosto alvo pela câmera 1 em distâncias variáveis.

A Tabela 4.3 exibe os valores obtidos através da câmera 2 e a Figura 4.2 mostra um gráfico da

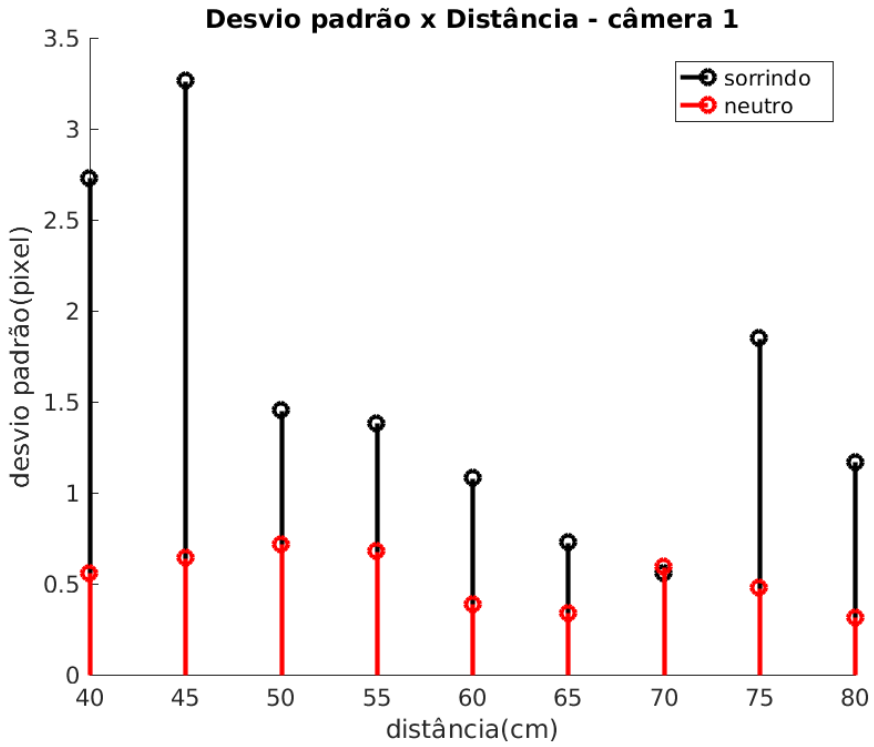


Figura 4.1: Gráfico da variação do desvio padrão, em pixeis, em relação a distância, em centímetros, de pontos rastreados nas imagens capturadas pela câmera 1.

variação do desvio padrão do comprimento dos lábios, para a Pose Neutra e a para Sorrindo, em relação a distância entre a câmera 2 e o alvo.

	Câmera 2 - Pose Neutra (pixel)		Câmera 2 - Pose Sorrindo (pixel)	
Distância da câmera (cm)	Média	Desvio padrão	Média	Desvio padrão
40	92,1488	1,3473	119,6076	2,2723
45	84,694	0,8848	106,7452	2,23
50	78,1044	1,0881	108,4763	1,2881
55	69,1071	0,9671	94,7001	1,2003
60	65,8582	0,8114	87,5216	0,5491
65	59,7229	0,5219	82,719	0,9558
70	55,7597	0,6367	75,3994	0,4099
75	53,7684	0,6277	67,9172	1,0836
80	49,7219	0,2373	68,2629	1,0946

Tabela 4.3: Comprimento horizontal dos lábios medido em pixeis a partir de imagens capturadas do rosto alvo pela câmera 2 em distâncias variáveis.

O desvio padrão é importante pois funciona como um indicador da precisão do processo de medida. Fosse o processo regido por uma distribuição puramente Gaussiana, o desvio padrão

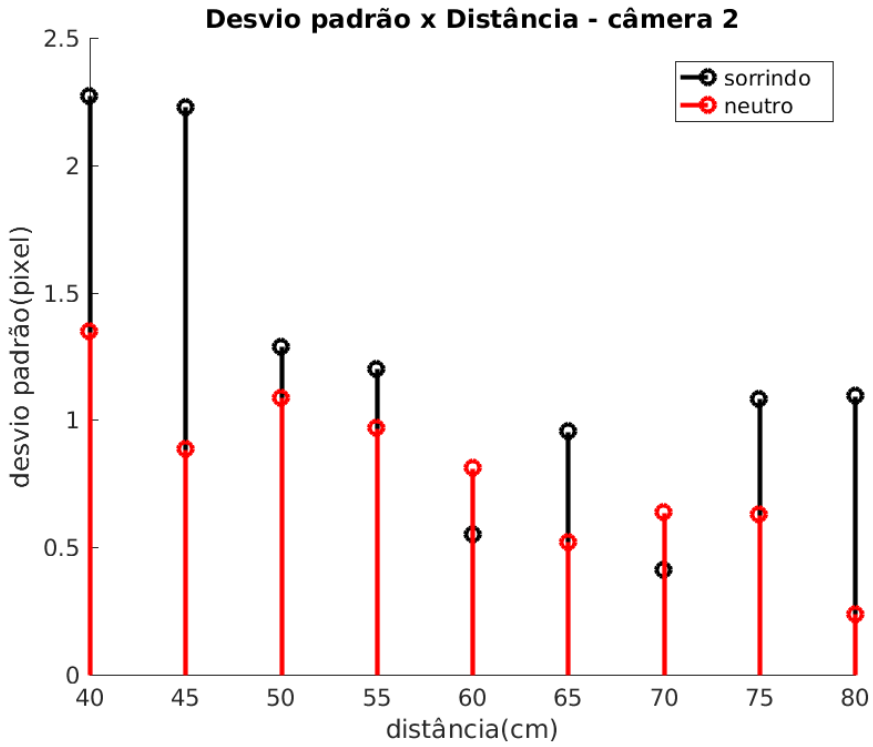


Figura 4.2: Gráfico da variação do desvio padrão, em pixeis, em relação a distância, em centímetros, de pontos rastreados nas imagens capturadas pela câmera 2.

do processo indicaria que aproximadamente 66% das medidas serão observadas dentro de uma distância igual ao desvio padrão da média do processo, que é aceito como o valor mais provável para o valor verdadeiro sendo medido. Os gráficos exibidos na Figura 4.1 e na Figura 4.2 mostram que existe incerteza no rastreamento de pontos da face e que a incerteza apresenta um comportamento que varia com a distância. Apesar de o desvio padrão ter sido pequeno quando comparado às dimensões de cada uma das das imagens capturadas, a existência dessa imprecisão pode afetar a qualidade da aplicação final. Além disso, outra análise interessante é que, como as imagens onde o rastreamento foi aplicado foram tomadas em intervalos de tempo muito curtos, e, ainda assim, houve variação nas medidas realizadas, diz-se que essas variações constituem um sinal de erro que deve possuir componentes de elevada frequência. Isto é, o ruído presente no processo de medida deve possuir componentes rápidas o suficiente para que seja capaz de influenciar medidas tomadas em instantes de tempo muito próximos uns dos outros. Essa observação motiva o uso de filtragem digital com o objetivo de cortar tais altas frequências.

Outro comportamento exibido pelos gráficos é o de que o desvio padrão é quase sempre maior quando a razão de distância é medida na Pose Sorrindo, do que quando ela é medida na Pose Neutra, ou seja, o rastreamento geralmente é mais preciso quando tomado em uma face sem expressões. Este resultado pode ser um reflexo do erro de truncamento introduzido pela escolha de apenas algumas das configurações encontradas pela análise PCA e que, aparentemente, as componentes escolhidas privilegiam configurações neutras.

Outra observação que pode ser tirada é que o desvio padrão em geral diminui com o aumento da distância. Esse acontecimento pode ser atribuído ao fato de que, como neste experimento as medidas são avaliadas em pixeis, quanto maior a distância do usuário a câmera menor o valor da razão de distância.

Finalmente, observando as tabelas é possível reparar a diminuição drástica da média das medidas conforme o valor de profundidade aumenta: vê-se que o valor obtido para o comprimento dos lábios a uma distância de 80cm é quase a metade do valor obtido a 40cm. Essa mudança brusca impossibilita uma boa determinação das distâncias mínimas e máximas, necessárias para a boa calibração de uma razão de distância, uma vez que estes parâmetros mudam com uma variação de profundidade, mesmo se o alvo estiver na mesma pose.

Estas duas últimas observações motivam o uso de uma técnica de medida que produza valores invariantes a distância que o usuário se encontra do sistema de captura. Esta técnica deve produzir sempre uma mesma razão de distância para uma mesma pose, independentemente da proximidade do usuário nas imagens de entrada.

4.3 Estimação de Tridimensionalidade

O objetivo dos experimentos nessa seção é avaliar a precisão da técnica de estimação tridimensional. Deve-se notar que uma precisão na estimação do X e do Y no sistema de coordenadas do mundo só é possível caso haja precisão na estimação do Z, como pode ser observado nas Equações 2.37, 2.38, 2.39 e 2.40.

Como nos experimentos da seção anterior, a razão de distância analisada foi a referente ao comprimento dos lábios. Foram utilizadas as mesmas imagens capturadas no experimento anterior, porém desta vez as medidas foram tomadas após o par de imagens ter sido combinado com os parâmetros intrínsecos da câmera para produzir medidas em centímetros, ou seja, os pontos foram estimados no sistema de coordenadas do mundo.

A Tabela 4.4 exibe os valores estimados em centímetros para a média e o desvio padrão do comprimento dos lábios para as poses Neutra e Sorrindo. Já a Figura 4.3 e a Figura 4.4 mostram gráficos para estes valores a medida que o usuário se afasta do sistema de captura.

O gráfico exibido na Figura 4.3 mostra como se comporta o desvio padrão a medida que o usuário se distancia do sistema de captura. O comportamento é similar ao observado nas Figuras 4.1 e 4.2, revelando a característica estocástica do processo de medida adotado. A diferença entre as Figuras 4.3, 4.1 e 4.2 é que a primeira apresenta os resultado quando as medidas são tomadas no sistema de coordenadas estimadas do mundo enquanto as últimas apresentam resultados quando medidas são tomadas diretamente nos planos das câmeras. Vê-se que a estimação de profundidade não altera este comportamento do sistema de medida.

A precisão da estimação tridimensional pode ser verificada ao se avaliar a variação da média em relação a distância. Esses dados são exibidos na Figura 4.4, nota-se que a média se mantém aproximadamente constante, principalmente nos intervalos entre 50cm e 65cm. Com isso pode-se

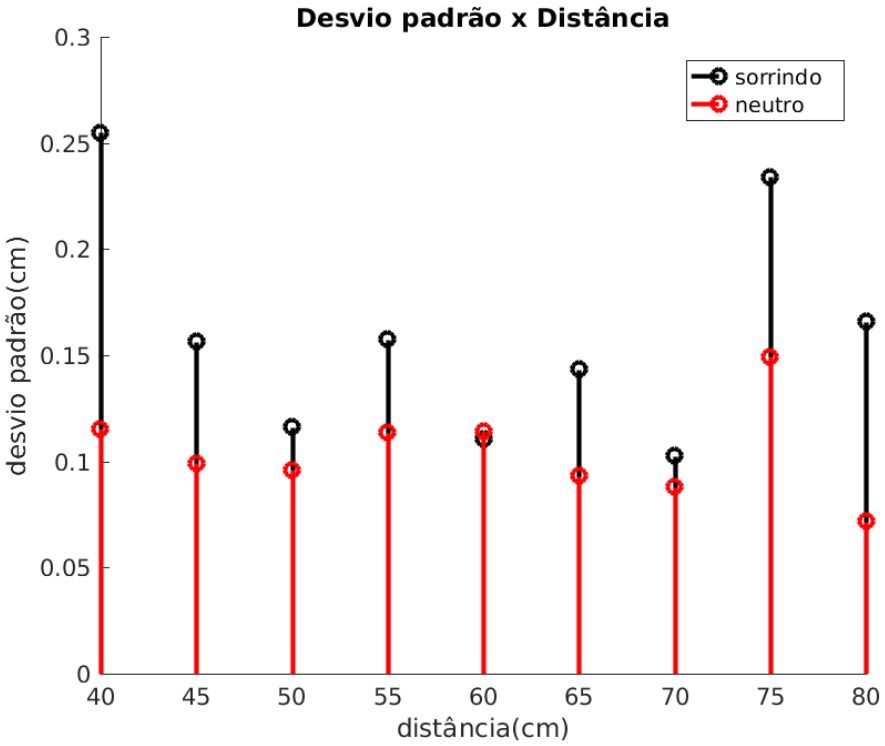


Figura 4.3: Gráfico da variação do desvio padrão, em centímetros, em relação a distância, em centímetros, de pontos estimados no sistema de coordenadas do mundo.

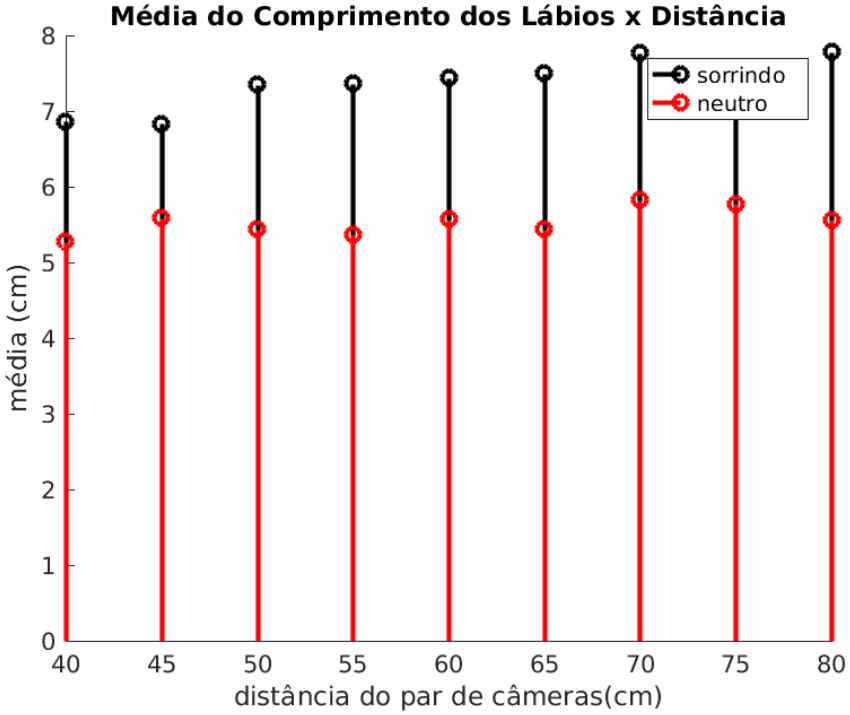


Figura 4.4: Gráfico da variação média, em centímetros, em relação a distância, em centímetros, de pontos estimados no sistema de coordenadas do mundo.

	Pose Neutra (cm)		Pose Sorrindo (cm)	
Distância da câmera (cm)	Média	Desvio padrão	Média	Desvio padrão
40	5,2755	0,1155	6,8652	0,2549
45	5,5857	0,0991	6,8311	0,1564
50	5,4356	0,0959	7,3474	0,116
55	5,3663	0,1135	7,3599	0,1574
60	5,5722	0,1144	7,4396	0,1107
65	5,4381	0,0932	7,4996	0,1433
70	5,8243	0,0879	7,7648	0,1024
75	5,7759	0,1494	7,1711	0,2339
80	5,5645	0,0716	7,7803	0,1659

Tabela 4.4: Comprimento horizontal dos lábios medido em centímetros a partir de imagens capturadas do rosto alvo em distâncias variáveis.

concluir que mesmo que aconteça uma variação da distância do alvo em relação as câmeras, a razão de distância se manterá estável, garantindo estabilidade para o modelo final.

4.4 Filtros

Nestes experimentos foram gerados gráficos para as sequências dos pesos de mistura com e sem filtragem de três das poses utilizadas. Cada gráfico mostra medidas de pesos de mistura para uma das poses, com um curva para os valor não-filtrado e curvas para saídas de alguns dos filtros projetados. As sequências foram obtidas em um vídeo gravado com o objetivo de movimentar especificamente a pose sendo testada. Os gráficos foram gerados selecionando-se manualmente parte das sequências gravadas que produziram comportamento interessante para comparação.

A Figuras 4.5, 4.6 e 4.7 mostram a atuação dos filtros sobre a variação do peso de mistura para as poses Olho Esquerdo, Boca Aberta e Sorriso respectivamente. Nestes gráficos, mostra-se sempre a saída não filtrada, a saída filtrada com filtro de média móvel de Hanning e a saída filtrada com filtro de projeto com janela de Hamming para algumas frequências de corte. Os filtros com projeto em janela tem todos comprimento 16.

Como visto nas Figuras 4.1 e 4.2 dos experimentos anteriores, o rastreamento em si apresenta instabilidades mesmo quando o rosto está imóvel. Portanto, pode-se esperar que em um vídeo longo onde ocorre movimentação e variação de luminosidade o sinal sem filtragem seja ainda mais instável. Esse comportamento é visto nas oscilações rápidas observadas nos gráficos em 4.5, 4.6 e 4.7.

A filtragem suaviza o sinal e consequentemente remove oscilações indesejáveis na animação final. Vários tipos de filtro foram analisados e seus desempenhos foram variados. Como pode ser observado, os filtros projetados pela técnica de janela foram compridos de mais e introduzem

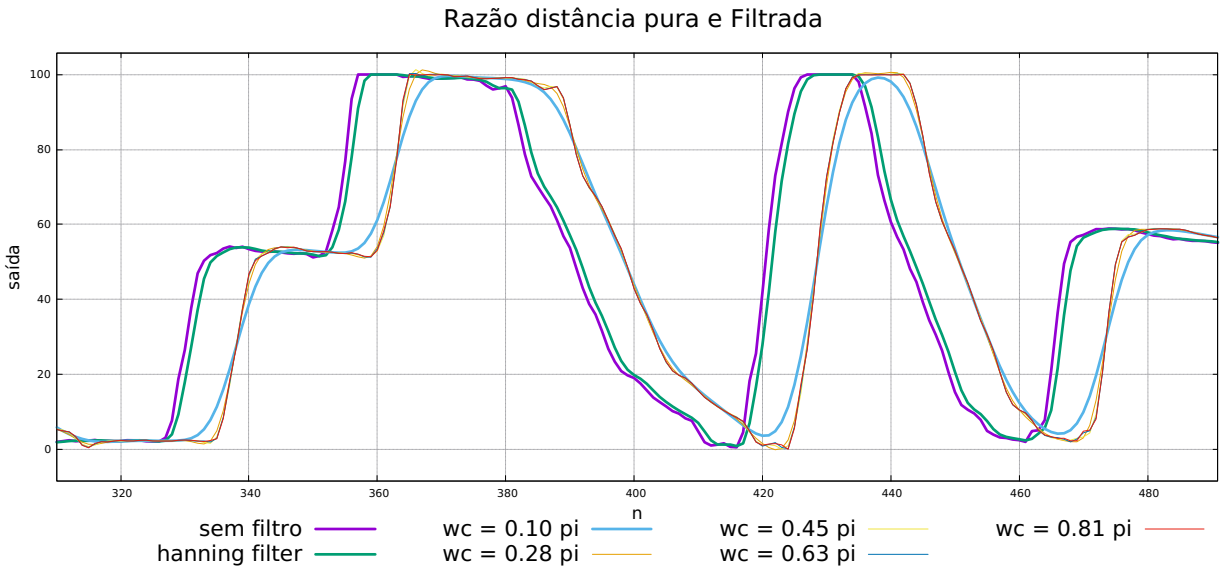


Figura 4.5: Peso de mistura para a Pose Olho Esquerdo.

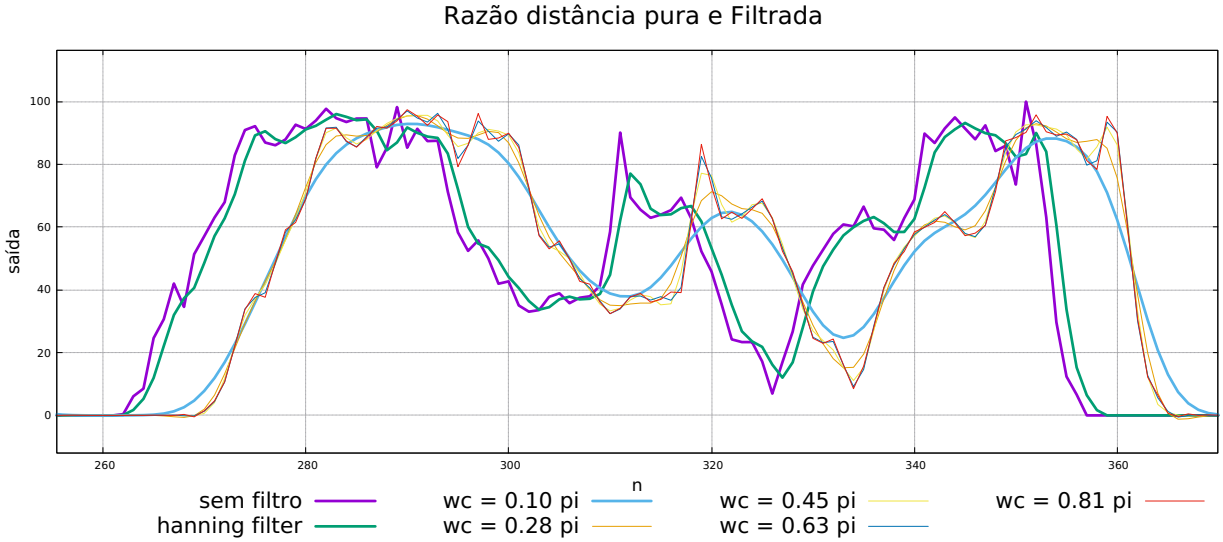


Figura 4.6: Peso de mistura para a Pose Boca Aberta.

atrasos no sinal de saída em relação ao de entrada sinal e, apesar de eles serem mais poderosos em filtrar altas frequências, esse tipo de filtro acaba prejudicando a performance da animação em tempo real.

Por outro lado o filtro de média móvel de Hanning segue adequadamente o sinal original e mantem uma boa atenuação das altas frequências. Esse filtro introduz um menor atraso em seguir o sinal de entrada, pois possui muito menos coeficientes que os outros filtros projetados. Para este trabalho, onde um dos objetivos é a animação em tempo real, o filtro de Hanning apresentou o melhor resultado. Poderia ter sido experimentado um projeto em janela utilizando-se menos coeficientes, mas como o filtro de média móvel mostrou bons resultados, decidiu-se por não prosseguir com a experimentação de mais filtros.

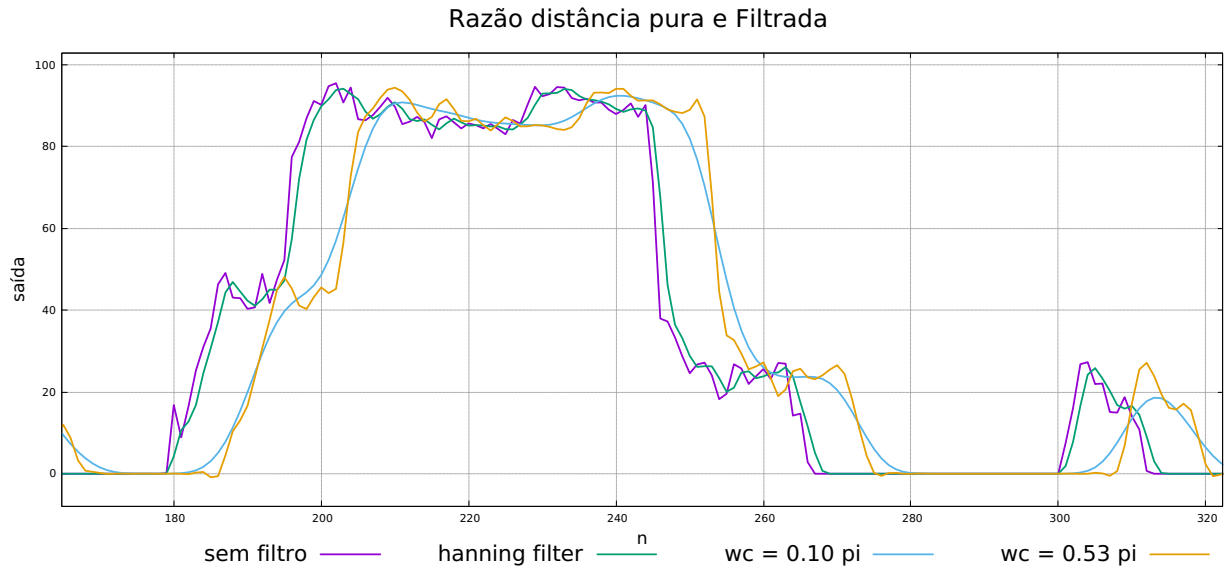


Figura 4.7: Peso de mistura para a Pose Sorriso.

4.5 Mistura de Poses

Neste experimento trabalhou-se apenas com a técnica de Mistura de Poses para compor um modelo final sem a influência do rastreamento de pontos da face. Isto foi feito para ser possível avaliar esta técnica isoladamente. Para isso foram atribuídos manualmente os valores dos pesos de mistura.

Alguns exemplos de modelos finais mostrando poses intermediárias renderizadas com sucesso podem ser vistos na Figura 4.8.

Como pode ser observado nos modelos exibidos, a técnica de mistura de poses teve ótimos resultados. Os resultados mostram que a implementação utilizando indexação de VBOs e que a ordenação dos vetores foi adequada. Não fosse esse o caso, seria possível observar falhas no modelo, como buracos ou presença de deformações irregulares.

Pode-se notar que a presença de cada pose chave no modelo final está bem relacionada com os pesos aplicados. Verifica-se que a aplicação da Equação 2.51 permite criar, a partir de poucas poses pré-definidas, uma quantidade enorme de poses intermediárias significativamente diferentes.

Com esses resultados nota-se que se o programa for capaz de gerar pesos de mistura que adequadamente correspondam às expressões apresentadas pelo usuário, a técnica de mistura de poses será capaz de compor um modelo final com expressões variadas.

4.6 Sistema em Funcionamento

Para que o sistema funcione adequadamente é necessária uma etapa de calibração das distâncias mínimas e máximas observadas em cada uma das razões de distância. Isso é feito de forma manual ao executar o programa, pois toma-se nota de valores impressos em tela sobre as distâncias entre

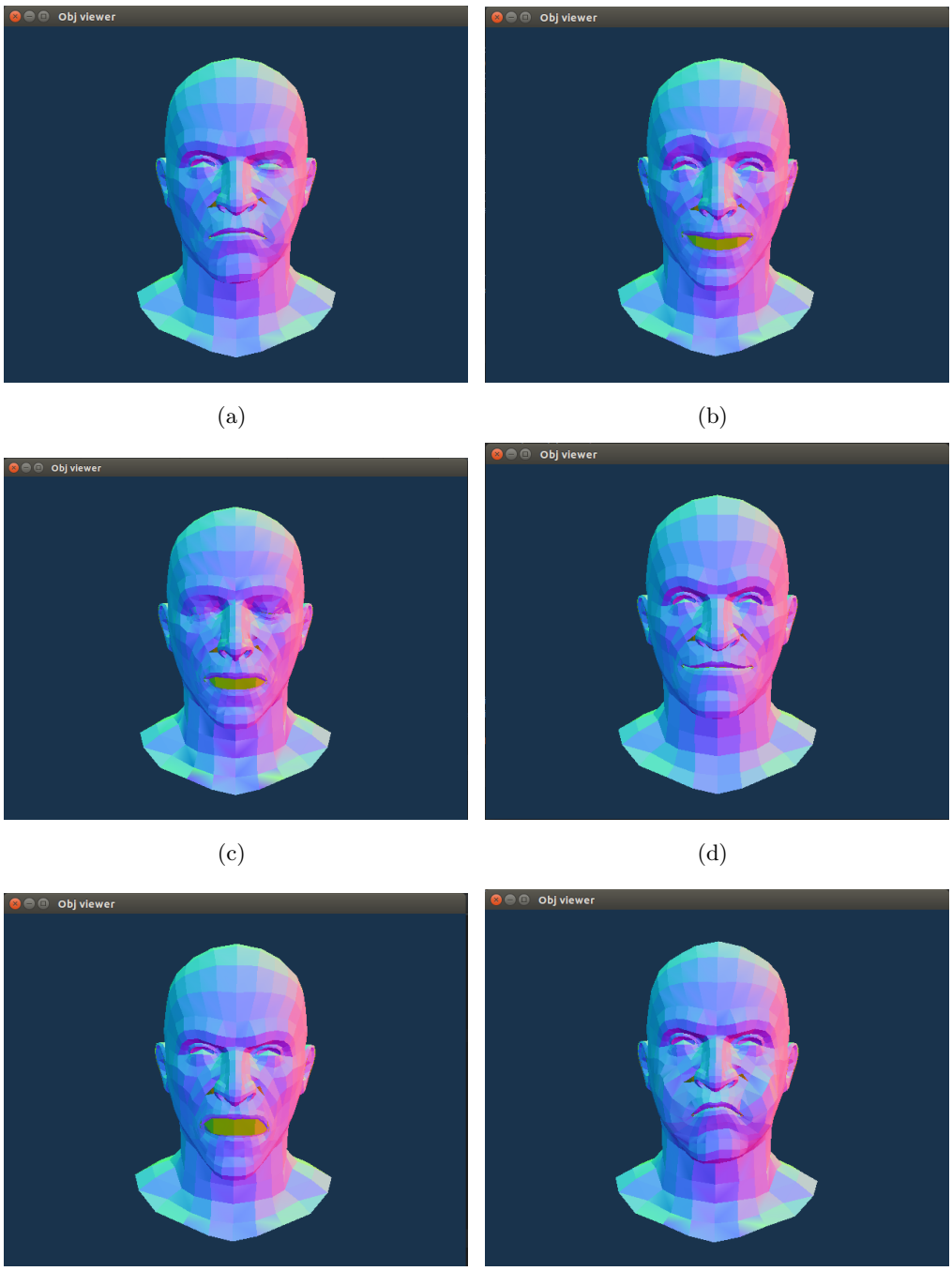


Figura 4.8: Exemplo de misturas geradas configurando os parâmetros de mistura manualmente. Pesos da Figura 4.12(a): Olho Esquerdo - 0.55, Boca Fechada - 0.35, Bochecha Esquerda - 0.75 e Bravo - 0.6. Pesos da Figura 4.12(b): Sobrancelha Direita - 0.95, Boca Aberta - 0.6 e Sorrindo - 0.4. Pesos da Figura 4.12(c): Olho Esquerdo - 1.0, Olho Direito - 1.0 e Boca Aberta - 0.6. Pesos da Figura 4.12(f): Sorrindo - 0.7 e Bravo - 0.6. Pesos da Figura 4.8(e): Boca Aberta - 1.0 e Bravo - 0.9. Pesos da Figura 4.8(f): Boca Fechada - 1.0, Bochecha Esquerda - 1.0, Bochecha Direita - 0.65, e Bravo - 1.0.

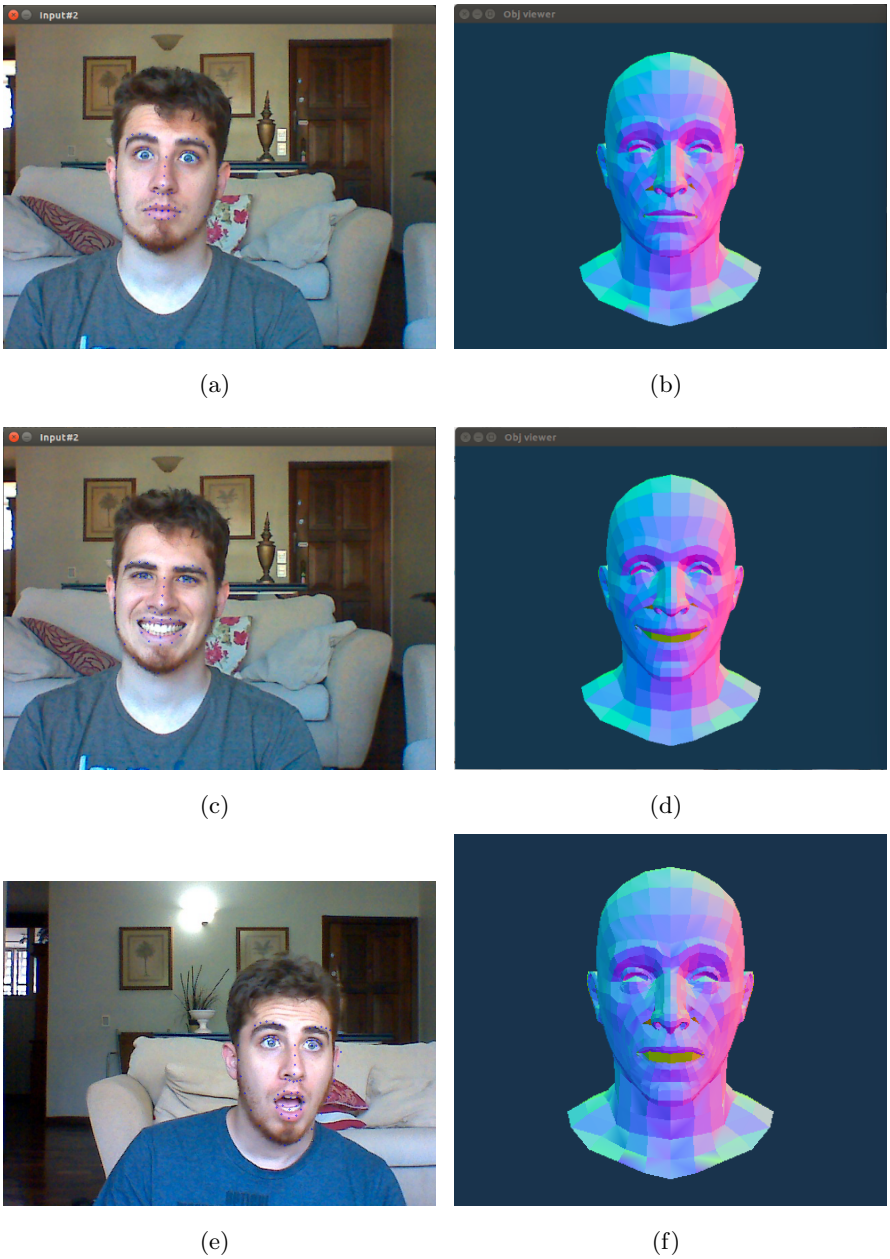
pose	distância mínima (cm)	distância mínima (cm)
sorriso	5,9	7,9
sobrancelha esquerda	2,4	3,7
sobrancelha direita	2,4	3,6
olho esquerdo	0,3	0,6
olho direito	0,3	0,6
boca aberta	0,3	4,3

Tabela 4.5: Valores de distância mínima e distância máxima definidos, para cada pose, após calibração.

os pontos tomadas no quadro atual. Por exemplo, pode-se sorrir e, então, anotar o valor medido para o sorriso que é impresso na tela. Os valores anotados são, posteriormente, inseridos no código para serem utilizados para cálculo das razões de distância. Os valores utilizados para a sequência de resultados obtidos a seguir são apresentados na Tabela 4.5.

Utiliza-se os coeficientes acima para rodar a aplicação sobre vídeos de entrada e guarda-se alguns quadros dos vídeos e da saída. As Figuras 4.9, 4.10, 4.11 e 4.12 mostram lado a lado quadros de entrada e os resultados da transferência de expressão para o avatar. Nas Figuras de 4.9 a 4.11 os quadros de entrada foram escolhidos de forma a mostrar expressões variadas que podem ser capturadas pela técnica proposta. A figura 4.12 mostra uma sequência de três quadros consecutivos tirada enquanto o usuário conversava com a câmera. Notar que a boca do avatar abre seguindo o movimento do usuário.

Os resultados mostram que o acoplamento entre o rastreamento de pontos do rosto com os pesos de mistura por meio de razões de distâncias se deu de forma satisfatória. Há poses para mistura que não foram utilizadas, mas todas as poses para as quais se definiu uma razão de distância foram adequadamente transferidas do usuário para o avatar. O avatar é capaz de sorrir, abrir a boca, movimentar as sobrancelhas e piscar segundo o movimento do usuário.

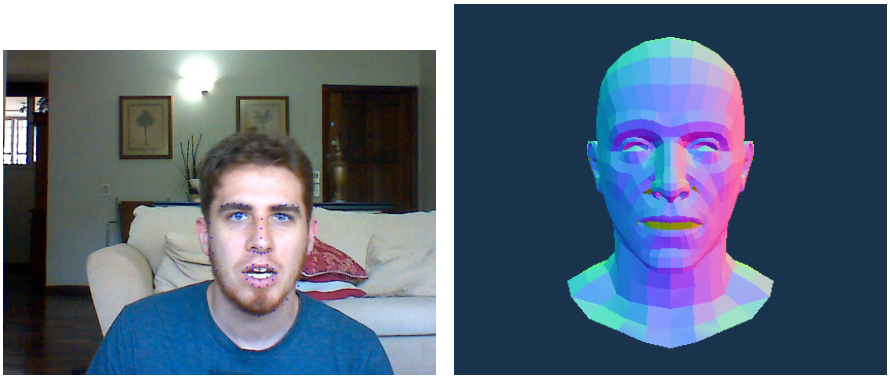


(a) (b)

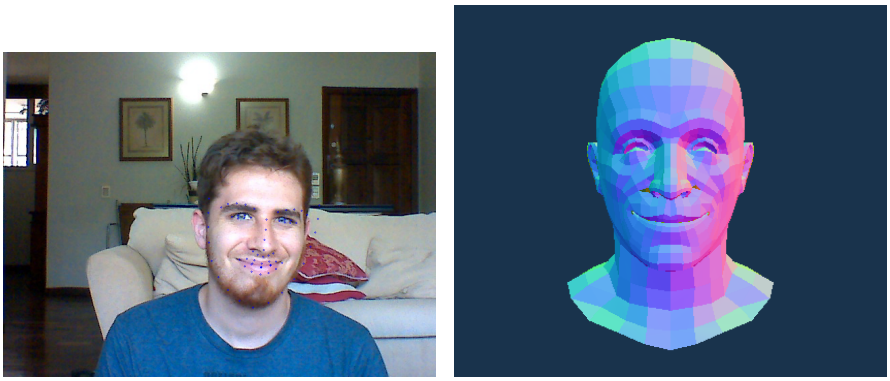
(c) (d)

(e) (f)

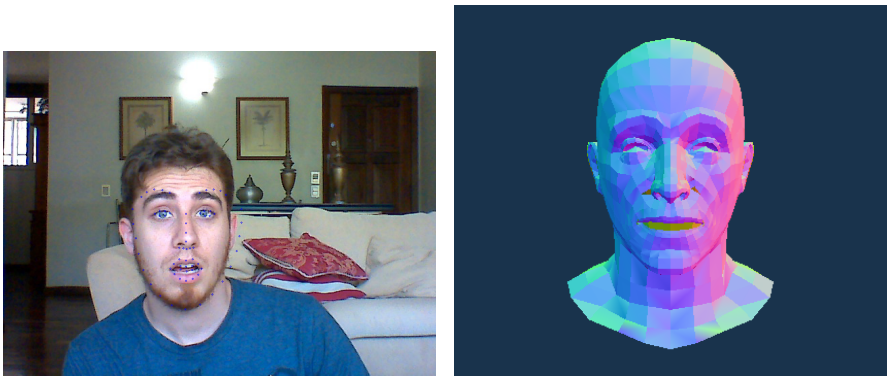
Figura 4.9: Imagens que demonstram o sistema em funcionamento. Em relação ao movimento da boca, as Figuras 4.9(d) e 4.9(f) deixam claro que há dois movimentos sendo rastreados independentemente: abrir a boca verticalmente como no ato de bocejar e abri-la horizontalmente como no ato de sorrir.



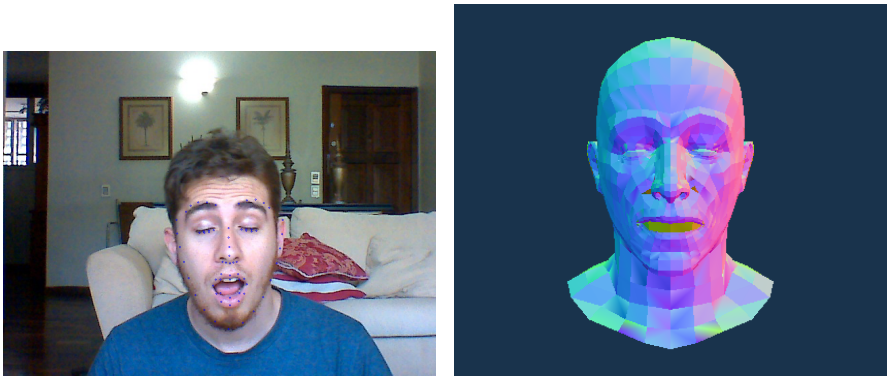
(a) (b)



(c) (d)

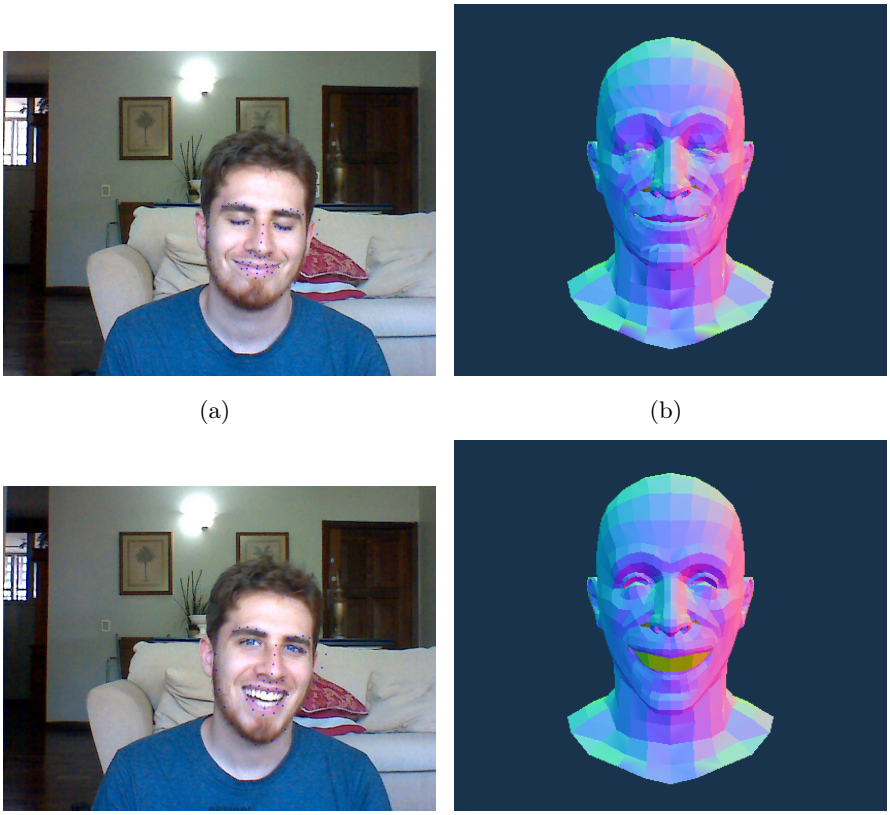


(e) (f)



(g) (h)

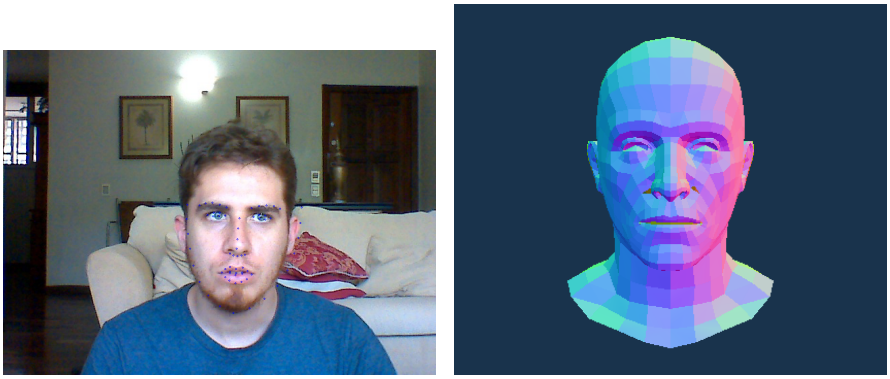
Figura 4.10: Imagens que demonstram o sistema em funcionamento. Compare a Figura 4.9(d) com a 4.10(d) para notar que o avatar está sorrindo com a boca fechada nesta e aberta naquela.



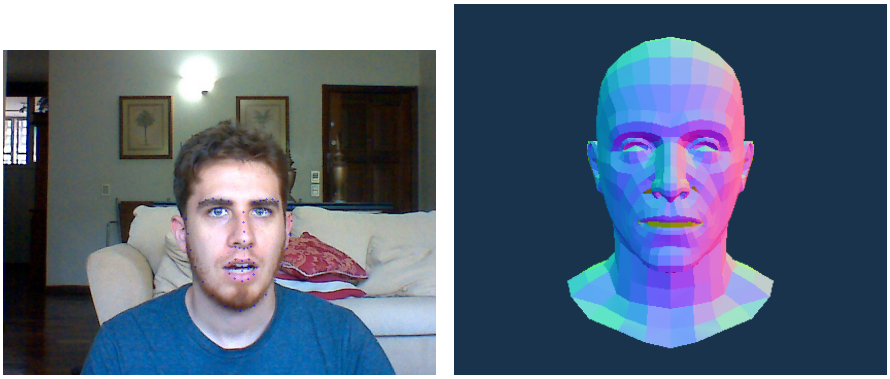
(a) (b)

(c) (d)

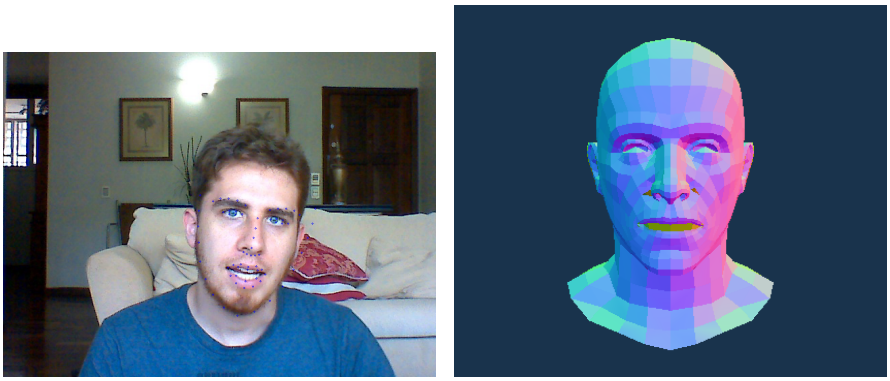
Figura 4.11: Imagens que demonstram o sistema em funcionamento. Novamente, o avatar pode sorrir com a boca fechada ou com a boca aberta, podendo o olho estar fechado e aberto. As poses são combinadas independentemente.



(a) (b)



(c) (d)



(e) (f)

Figura 4.12: Imagens que demonstram o sistema em funcionamento *frame* a *frame*. A sequência mostra a abertura gradual da boca enquanto o personagem conversa com as câmeras.

Capítulo 5

Conclusões

Desenvolveu-se um sistema que permite transferir um conjunto de movimentos faciais para um avatar computacional a partir de uma sequência de imagens capturadas por um par de câmeras. O sistema realiza a tarefa utilizando *webcams* simples e não exige um ambiente controlado, marcadores no usuário ou mesmo uso de iluminação especial. Para se ter uma ideia da robustez dos resultados, o sistema foi testado com sucesso em ambientes como uma sala de escritório e até mesmo a sala de estar de um dos autores.

Os movimentos independentemente transferidos incluem: sorrir, abrir a boca, fechar os olhos e levantar as sobrancelhas. A técnica chave utilizada foi conectar características de um conjunto de pontos obtidos de um processo de rastreamento visual com os pesos de mistura da técnica de Mistura de Poses, permitindo gerar expressões variadas para o avatar a partir da combinação de um conjunto reduzido de expressões chave.

Para o rastreamento visual, utilizou-se a biblioteca FaceAnalysis SDK para capturar 66 pontos chaves do rosto humano em uma sequência de imagens. Já para gerar os pesos de mistura, foi utilizada uma regra simples baseada na razão entre distâncias de pontos nomeados. Com o objetivo de tornar o processo robusto à distância do usuário ao par de câmeras, se utilizou estimação de tridimensionalidade para tornar os pesos de mistura uma função das distâncias reais entre os pontos e não das distâncias em pixeis das câmeras. Para isso, foi preciso uma etapa prévia de calibração das câmeras utilizadas.

Além disso, com o intuito de suavizar os resultados ao eliminar ruídos, os pesos de mistura passam por um filtro digital. Dois tipos de filtros foram utilizados: um filtro de de média móvel de Hanning de comprimento três e um conjunto de filtros projetados pela técnica de corte de janela. Apesar de muito menos elaborado, o primeiro filtro apresentou melhores resultados que os últimos. Entende-se que o ruído presente no sinal não possui comportamento espectral que justifique um projeto cuidadoso de um filtro passa-baixas. Um filtro curto, que ainda assim atenue frequências altas, performa melhor por impor um atraso menor no sinal de saída.

Uma série de experimentos foi realizada com o objetivo de medir a confiança em cada etapa do processo. A partir desses experimentos pôde-se inferir a distância que o usuário deve se posicionar do par de câmeras para que a técnica transfira os movimentos para o avatar de forma mais efetiva.

Outro resultado foi que apesar de adequada para capturar o movimento da boca e das sobrancelhas, a regra associativa entre pontos e pesos de mistura não foi capaz de capturar outros movimentos faciais, como o movimento de bochechas e o piscar dos olhos. Isso se deve, em parte, à certa rigidez do algoritmo de rastreamento utilizado quanto ao formato do rosto, o que impede que este seja aplicado, pelo menos diretamente, para capturar nuâncias de certas expressões. Com isso, algumas das poses chave disponíveis não foram utilizadas na aplicação final.

Os resultados obtidos mostram a validade da técnica. Por exemplo, o programa desenvolvido foi capaz de sincronizar a abertura e o fechamento da boca do avatar enquanto o usuário conversava com a câmera em um dos vídeos de teste. Além disso, quando executando em máquinas comuns e ligado diretamente nas câmeras, o programa mostra os resultados em tempo de execução sem atraso perceptível entre entrada e saída. Por rodar em máquinas ordinárias, não exigir longas horas de processamento e não exigir controle especial do ambiente de filmagem a metodologia proposta se confirma como uma de baixo custo.

5.1 Trabalhos Futuros

A técnica de mistura de poses chaves se mostrou uma maneira simples de gerar expressões complexas a partir de medidas simples obtidas em uma imagem. No entanto, na implementação atual os pontos na malha das poses chaves estão limitados a serem os mesmos dos pontos da pose base a menos de uma translação. Com isso não pode-se animar uma pose chave onde, por exemplo, o avatar vira o pescoço. A geometria vetorial na mistura de poses não funcionaria adequadamente. É possível aplicar a técnica de mistura de poses com rotação dos pontos do modelo, desde que se conheça a transformação completa que deve ocorrer em cada ponto, não apenas sua translação. Uma possível direção para o trabalho seria implementar estas transformações e as misturas delas. Nota-se que isso implicaria, possivelmente, em mudar a representação dos modelos e acarretaria maior custo na renderização.

Ainda sobre as poses chaves, durante o projeto o artista gráfico foi capaz de produzir mais poses bases do que criou-se regras associativas entre os pontos do rastreamento visual e os pesos de mistura. Por exemplo, o programa não possui regra capaz de reconhecer uma face com bochecha inchada. O problema principal é que o algoritmo de rastreamento visual não segue bem o contorno da bochecha nessa situação em particular, possivelmente devido à falta desta expressão nos dados onde foi treinado. Uma direção natural para a continuação do trabalho é o desenvolvimento de mais regras de associação entre imagem e pesos de mistura. O processo de treinamento descrito em [17] poderia ser repetido, sendo fornecido, dessa vez, um conjunto maior de dados de treinamento, incluindo as expressões que se deseja capturar. Uma abordagem mais simples seria construir rastreadores especializados que trabalhariam com dicas dos dados já capturados pela FaceAnalysis SDK. Apesar do algoritmo de rastreamento desta SDK não ser perfeito, ele fornece uma excelente pista de onde localizar as partes do rosto e essa informação poderia ser utilizada para construir rastreadores específicos para o que se quer medir. Por exemplo, é possível utilizar os pontos fornecidos para delimitar uma pequena janela de busca ao redor da bochecha onde se aplicaria

algum truque para detectar bochechas cheias.

Tais melhorias poderiam tornar a execução do programa mais pesada e justificaria o desenvolvimento de um código que melhor aproveitasse ferramentas de aceleração gráfica. O código atual utiliza aceleração gráfica somente para renderizar o resultado final, de forma que todas as outras operações são realizadas por código de máquina executado no processador principal e em apenas uma linha de execução. Outra justificativa para o uso de aceleração gráfica em outras etapas da aplicação seria permitir que o programa rodasse suavemente com malhas mais pesadas. Na implementação atual deste trabalho obtém-se atraso significativo quando os modelos envolvidos apresentam dezenas de milhares de pontos.

Finalmente, em relação à captura de dados, a aplicação desenvolvida neste trabalho utiliza apenas um par de câmeras baratas. Apesar de resultados interessantes terem sido atingidos, certamente há limitações, ou no mínimo dificuldades, impostas pelo método de captura utilizado. Uma direção interessante em que se poderia evoluir seria a adição de um sensor de nuvem de pontos, como o Kinect, ao pipeline das técnicas. Apesar deste tipo de sensor ser mais caro que um par de câmeras, a adição de um sensor de nuvem de pontos não desclassificaria a aplicação como uma de baixo custo, ao se considerar o custo total do desenvolvimento de animação. Porém, como essa adição mudaria drasticamente o formato dos dados, seria necessário também desenvolver novas regras que associem pesos de mistura à nuvem de pontos.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] SHKULEV, H. Disponível em: <http://www.maximonline.ru/guide/cinema/_article/the-hobbit-the-desolation-of-smaug-2/>.

[2] MURPHY, K. P. *Machine Learning - A Probabilistic Perspective*. [S.l.: s.n.], 2012.

[3] HARTLEY, R.; ZISSERMAN, A. *Multiple view geometry in computer vision*. [S.l.]: Cambridge university press, 2003.

[4] WASHINGTON, U. of. *Stereo and 3D Vision - Lecture 16*. Disponível em: <https://courses.cs.washington.edu/courses/cse455/09wi/Lects/lect16.pdf>.

[5] MABIN, D. *Tutorial de OpenGl*. Disponível em: <http://www.opengl-tutorial.org/.>.

[6] MASTERS, M. *How to Create Your First Character Rig in Blender: Part 1 - Setting up the Armature*. Disponível em: <http://blog.digitaltutors.com/how-to-create-your-first-character-rig-in-blender-part-1/>.

[7] GERDELAN, A. *Animating Blend Shapes*. Disponível em: <https://capnramses.github.io/opengl/blend_shapes.html>.

[8] LIST, W. Disponível em: <http://pcwallart.com/nicolas-cage-face-wallpaper-3.html>.

[9] BOUGUET, J. Y. *Camera Calibration Toolbox for Matlab*. Disponível em: <https://www.vision.caltech.edu/bouguetj/calib_doc/.>.

[10] MOBILE, PC & Console Gaming & Animation (by Entertainment, 2D, 3D, Visual Effects, TV, Direct-to-DVD and Content) Market - Global Advancements, Business Models, Market Forecasts & Analysis (2012 - 2016). [S.l.], 2011. Disponível em: <http://www.marketsandmarkets.com/Market-Reports/animation-gaming-market-514.html>.

[11] ROOS, D. *Why do movies cost so much to make?* [S.l.], 9 December 2009. Disponível em: <HowStuffWorks.com. http://entertainment.howstuffworks.com/movie-cost.html>.

[12] VISWANATHAN, B. *Why Do Animated Films Cost So Much to Make?* [S.l.], JUNE 20 2013. Disponível em: <http://www.slate.com/blogs/quora/2013/06/20/pixar_and_monsters_university_why_do_animated_movies_cost_so_much.html>.

[13] GONZALEZ, R. C.; WOODS, R. E. *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008.

[14] ROCHA, T. da. *UMA PROPOSTA PARA A CLASSIFICAÇÃO DE AÇÕESHUMANAS BASEADA NAS CARACTERÍSTICAS DO MOVIMENTOE EM REDES NEURAIS ARTIFICIAIS*. Dissertação (Mestrado) — Universidade de Brasíilia, 2012.

[15] FORSYTH, J. P. D. A. *Computer Vision - A Modern Approach*. [S.l.]: Prentice Hall, 2002.

[16] SARAGIH, J. M.; LUCEY, S.; COHN, J. F. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, Springer, v. 91, n. 2, p. 200–215, 2011.

[17] SARAGIH, J. M. *CSIRO Face Analysis SDK*. Disponível em: <http://face.ci2cv.net/>.

[18] BOLLES, R. C.; BAKER, H. H.; MARIMONT, D. H. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, Springer, v. 1, n. 1, p. 7–55, 1987.

[19] HARTLEY, R. I.; STURM, P. Triangulation. *Computer vision and image understanding*, Elsevier, v. 68, n. 2, p. 146–157, 1997.

[20] OPENGL. *Wikibooks OpenGl*. Disponível em: <https://en.wikibooks.org/wiki/OpenGL_Programming/Modern_OpenGL_Introduction>.

[21] GROUP, K. *OpenGl Wiki*. Disponível em: <https://www.opengl.org/wiki/Tutorial2:_VAOs,_VBOs,_Vertex_and_Fragment_Shaders_(C_/_SDL)>.

[22] LIU, C. *AN ANALYSIS OF THE CURRENT AND FUTURE STATE OF 3D FACIAL ANIMATION TECHNIQUES AND SYSTEMS*. Dissertação (Mestrado) — Communication University of China 2006, 2009.

[23] III, J. O. S. *class notes on INTRODUCTION TO DIGITAL FILTERS - Difference Equation*. Disponível em: <https://ccrma.stanford.edu/ jos/fp/Difference_Equation_I.html>.

[24] III, J. O. S. *class notes on INTRODUCTION TO DIGITAL FILTERS - What is a Filter?* Disponível em: <https://ccrma.stanford.edu/ jos/filters/What_Filter.html>.

[25] COX, M. et al. Csiro face analysis sdk. *Brisbane, Australia*, 2013.