UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

PROTÓTIPO PARA TRATAMENTO DE IMAGENS FACIAIS RASTER 2D, PARA GERAR A IMAGEM 3D.

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDO À UNIVERSIDADE REGIONAL DE BLUMENAU PARA A OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA COM NOME EQUIVALENTE NO CURSO DE CIÊNCIAS DA COMPUTAÇÃO — BACHARELADO

TARCÍSIO DE SOUZA NETO

BLUMENAU, JUNHO/2001

2001/1-48

PROTÓTIPO PARA TRATAMENTO DE IMAGENS FACIAIS RASTER 2D, PARA GERAR A IMAGEM 3D.

TARCÍSIO DE SOUZA NETO

ESTE TRABALHO DE CONCLUSÃO DE CURSO, FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO

_____

Prof. Dalton Solano dos Reis — Orientador na FURB

_____

Prof. José Roque Voltolini da Silva — Coordenador do TCC

BANCA EXAMINADORA

_____

Prof. Dalton Solano dos Reis

_____

Prof. Paulo César Rodacki Gomes

_____

Prof. Antonio Carlos Tavares

DEDICATÓRIA

Dedico este trabalho, com carinho, àqueles que
o tornaram possível, mediante o apoio direto e
o constante estímulo que me motivaram a realizá-lo

AGRADECIMENTOS

Primeiramente ao professor e orientador Dalton Solano dos Reis, que com muita dedicação e experiência conduziu-me na realização deste trabalho de conclusão.

A todos os professores do Curso de Computação, construtores do conhecimento, tão importante na formação profissional do acadêmico.

Aos amigos acadêmicos, companheiros de caminhada, incentivadores indispensáveis no cumprimento de mais uma importante etapa em nossas vidas.

A minha namorada Bianca, que vem sendo nos últimos anos uma das pessoas mais importantes para mim.

A meus pais, que sempre me incentivaram e ajudaram, dando-me condições necessárias para que eu pudesse concluir este curso.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

RESUMO

Este trabalho consiste num estudo de métodos utilizados para se fazer à construção de modelos faciais humanos 3D, usando um modelo facial genérico 3D e várias imagens faciais em 2D. Um algoritmo é desenvolvido para extrair manualmente todas as características faciais necessárias, tanto das imagens faciais do perfil frontal quanto dos perfis laterais. Tem-se também um modelo facial genérico 3D, o qual é deformado por estes pontos característicos através de transformação geométrica. Finalmente, é executado o mapeamento da textura para alcançar resultados realísticos.

ABSTRACT

This work consists of a study of methods used to do to the construction of human facial models 3D, using a generic facial model 3D and several facial images in 2D. An algorithm is developed to extract all the necessary facial characteristics manually, as much of the facial images of the front profile as of the lateral profiles. It is also had a generic facial model 3D, which it is deformed for these characteristic points through geometric transformation. Finally, the mapping of the texture is executed to reach realistic results.

1 INTRODUÇÃO

Quando foram criados, os computadores eram utilizados para calcular e imprimir números, desde então a utilização se diversificou muito. Nos últimos anos, com o surgimento de novas tecnologias, a utilização das funções gráficas se tornou indispensável para profissionais de diversas áreas (Brown, 1995).

Com a necessidade da utilização de funções gráficas, cresceu muito o estudo da Computação gráfica que é a área da ciência da computação que estuda a aquisição (síntese), manipulação (processamento) e interpretação (análise) de imagens por meio de computadores. Para ser possível manipular ou interpretar estas imagens é necessária a utilização de alguma técnica de armazenamento e especificação das figuras, que torne possível sua visualização em algum dispositivo computacional (Persiano, 1989) e (Banon, 1989).

Recentemente, entre as técnicas de tratamento de imagens pode-se citar as técnicas voltadas a tratamento de imagens faciais, que receberam atenção considerável, particularmente nos campos de visualização computacional e comunidades de processamento de sinal. Um problema importante é como criar um modelo 3D de um indivíduo específico (Ho, 2000).

Segundo Huang (1996), em telecomunicações visuais e teleconferência as faces humanas são a parte principal nas cenas. Tem-se então a idéia principal, que é construir um modelo para faces humanas em 3D. Um dos problemas básicos é como criar um modelo facial 3D de uma pessoa em particular usando várias imagens faciais 2D. Foram desenvolvidos vários algoritmos de modelagem automática, sendo que estes conseguiram resolver algumas limitações bastante difíceis. Outros algoritmos tentam achar características faciais usando minimização de esforço em *splines*, como *snakes* e modelos deformados. Eles normalmente assumem posições iniciais conhecidas das características faciais.

Existe também um algoritmo genético para modelagem de faces chamado FGPFM, que modela a partir de uma imagem de face descalibrada, e também toma como parâmetro um modelo facial genérico flexível (Ho, 2000).

Uma das partes que merece uma atenção considerável no desenvolvimento de técnicas de modelagem facial refere-se ao processo de aquisição de imagens, que consiste em transformar as imagens reais em imagens digitais. Este processo é feito através de dispositivos de entrada

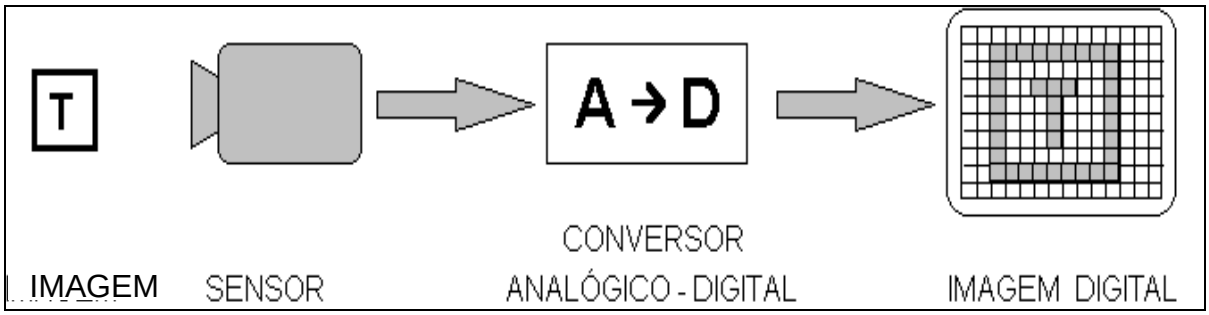
gráfica. De acordo com o tipo e destinação da imagem, os dispositivos de entrada gráfica podem ser classificados como vetoriais ou *raster* (matriciais) (França Neto, 1998).

Segundo Celani (1997) e Lindley (1991) existem duas grandes categorias de imagens: as de formato vetorial, que pode ser entendido como uma representação esquemática da realidade, formada a partir do desenho sucessivo de inúmeras linhas; e as de formato *raster* que pode ser entendido como uma matriz retangular de pontos coloridos chamados *pixel* em 2D e *voxel* em 3D, que quando observados a uma distância apropriada, apresentam uma aceitável representação da realidade. *Pixel* é cada ponto (x, y) da figura, com os seus atributos (cor, luminância etc). *Voxel* é um objeto geo-espacial tridimensional que representa a mais pequena unidade de um volume, sendo o equivalente tridimensional do *pixel*.

A imagem gráfica com formato *raster* (2D/3D) é a mais comum tecnologia de representação de imagens em uso hoje, pois torna possível simular os efeitos de cor, luz e sombra nos objetos realísticos e permite manipular o menor detalhe da figura. Um arquivo raster pode ser considerado como sendo uma matriz cujos índices de linhas e de colunas identificam um ponto na imagem, e o correspondente valor do elemento da matriz identifica a cor naquele ponto. As figuras normalmente especificadas por este meio são imagens captadas através de dispositivos de entrada gráfica (*scanners*, câmeras de vídeo etc.) para serem modificadas através da aplicação de técnicas de manipulação de suas características originais. A esta manipulação chama-se de processamento de imagens (Persiano, 1989).

Os dispositivos de entrada *raster* (matricial) são em geral utilizados de forma não-interativa, desta forma, se adequando mais a aquisição de grandes volumes de dados. A estrutura dos dispositivos de entrada do tipo *raster* consiste de um sensor que capta sinais no espaço ambiente e um circuito analógico-digital que converte esses sinais analógicos para o formato matricial, como pode ser visto na figura 1.

Figura 1 - Estrutura dos dispositivos de entrada do tipo *raster* (matricial)



Fonte: França Neto, 1998

O processo de conversão de uma imagem para uma imagem digital é conhecido como digitalização. Os dispositivos de entrada *raster*, são em sua maioria destinados à digitalização de imagens. Dependendo do meio no qual se encontra a imagem a ser digitalizada tem-se o *frame grabber*, a câmera de vídeo, a câmera fotográfica digital, o *scanner* e o *film scanner* (França Neto, 1998).

Após o processo de aquisição da imagem, vem o processo de armazenamento da imagem que consiste em gravar as informações da imagem de uma forma que possa ser recuperada posteriormente. Como se pretende trabalhar mais com imagens do tipo *raster*, dar-se-á maior ênfase a esse formato. O processo pode ser feito de duas formas, usando um formato dedicado ou utilizando algum formato padrão (França Neto, 1998).

Para armazenar uma imagem é necessário que as informações sejam gravadas em um dispositivo de armazenamento como um arquivo de dados. Essa estrutura deve conter as informações do tipo da imagem, resolução de cor, dimensões, os dados que formam a imagem entre outros (Foley, 1990).

Desta forma o presente trabalho desenvolveu um protótipo de software para tratamento de imagens faciais raster 2D, para gerar um modelo 3D. O protótipo tem como principais etapas: (1) a etapa de extração de características, (2) mapeamento do modelo 2D para o modelo 3D e (3) a etapa de mapeamento da textura dos modelos 2D para o modelo 3D.

1.1 OBJETIVO

O trabalho proposto tem como objetivo principal desenvolver um protótipo de software para definição de um ambiente de tratamento de imagens faciais. Onde é feita a aquisição da imagem *raster* 2D da cabeça de uma pessoa em três perfis (frontal, direito e esquerdo) através de uma câmera de vídeo. Tem-se como objetivo também transformar estas imagens bidimensionais em uma única imagem tridimensional, e apresentá-la como resultado.

1.2 ESTRUTURA

O trabalho é estruturado da forma a seguir.

No primeiro capítulo, é apresentado uma visão geral deste trabalho, objetivos e a sua organização.

O segundo capítulo descreve conceitos sobre imagens digitais bidimensionais (2D) e tridimensionais (3D), destacando síntese e representação destas imagens.

No terceiro capítulo são abordadas as técnicas utilizadas para modelagem de imagens tridimensionais (3D).

No quarto capítulo são abordadas as técnicas de transformações de imagens raster 2D em modelos 3D.

No capítulo cinco, descreve-se a especificação, implementação e o funcionamento do protótipo.

No sexto capítulo, encontram-se relacionadas às considerações finais, análises, conclusões e sugestões para estudos futuros.

2 IMAGENS DIGITAIS

"Uma imagem vale por mil palavras...", esta conhecida frase já era proferida muito antes do nascimento dos computadores e, portanto, em épocas em que era difícil imaginar a possibilidade de que desenhos poderiam um dia vir a ser criados e interpretados automaticamente. Hoje, através da Computação Gráfica, isto, e muito mais, faz parte do dia-a-dia de inúmeras pessoas, desde os apaixonados por videojogos até os projetistas de aviões. Existe uma característica do ser humano, comprovada não só na prática como em estudos científicos, e que é utilizada pela Computação Gráfica: o homem consegue absorver e transmitir um número muito maior de informações quando estas se encontram sob a forma de imagens (Newman, 1973).

É importante salientar que o estudo de imagens dentro da Computação Gráfica divide-se, basicamente, em duas partes: o que trata de modelos bidimensionais (2D) e o que trata de modelos tridimensionais (3D). O modelo 2D é quando se trabalha em um plano (XY) e o 3D é quando trabalha-se no espaço (XYZ). Eventualmente pode-se ouvir falar em Computação Gráfica 4D, que é quando trabalha-se com o modelo 3D em conjunto com o tempo (que significa a animação) (Berg, 2001).

2.1 IMAGENS DIGITAIS 2D

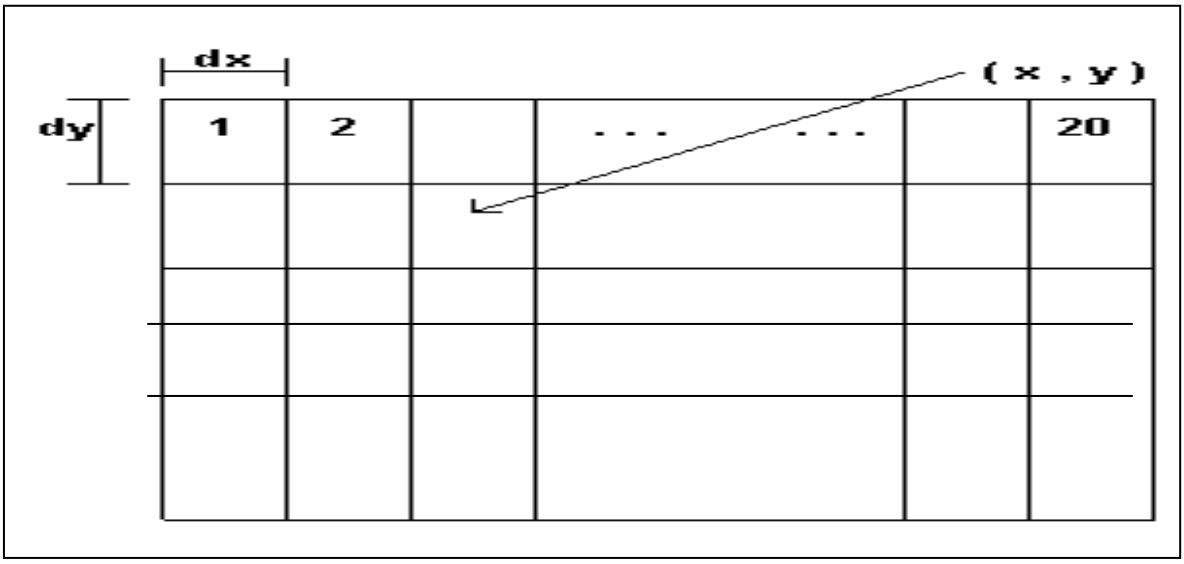
A computação gráfica é geralmente voltada a gerar, processar (tratar) ou interpretar informações visuais. Estas informações estão contidas dentro de uma imagem, a qual é uma representação gráfica, plástica ou fotográfica de pessoa ou de objeto (Velho, 1994).

Para entender melhor como as imagens são adquiridas e armazenadas, será visto primeiro como estas imagens são representadas digitalmente. Representar uma imagem consiste em modelar esta imagem de forma que os dados que formam esta possam ser interpretados pelo computador (França Neto, 1998).

A representação de uma imagem pode ser feita de forma vetorial ou *raster*. Neste caso, será trabalhado com a representação de imagens *raster*. A forma vetorial utiliza aspectos geométricos no domínio contínuo e não se aplica ao objetivo deste trabalho.

Para representar uma imagem digitalmente, deve-se discretizar as informações referentes a tal imagem. Uma das formas mais utilizadas para a discretização de uma imagem consiste em tomar o domínio como sendo um retângulo. Esse retângulo será representado como uma matriz bidimensional onde estarão as informações da imagem. As dimensões dessa matriz vão depender das proporções do retângulo e do grau de detalhes em que é feita a digitalização. Uma imagem real pode ser dividida em vários pedaços, pequenos retângulos formando um reticulado bidimensional, como pode ser visto na figura 2 (Velho, 1994).

Figura 2 - Reticulado bidimensional.



Fonte: Velho, 1994

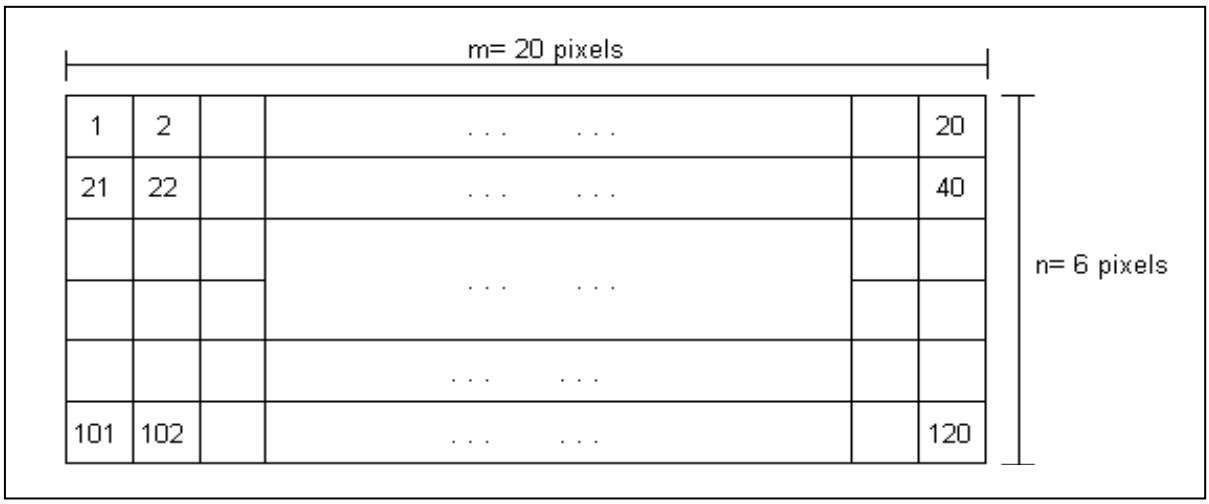
Os retângulos podem ter dimensões diferentes, ou seja, mais alto do que largo ou o contrário, mas em geral esses são quadrados. Cada ponto (x, y) com os seus atributos (cores, luminância, etc) são chamadas de elemento de imagem ou pixel, (do inglês *picture element*). O número de linhas da matriz (*m*) é chamado de resolução vertical da imagem e o número de colunas (*n*) é chamado de resolução horizontal. Denomina-se resolução espacial, ou resolução geométrica da representação de imagens *raster* ao produto *m* x *n* da resolução vertical pela resolução horizontal. A resolução espacial estabelece a freqüência de amostragem final da imagem. A resolução espacial dada em termos absolutos não fornece muita informação sobre a resolução real da imagem quando realizada em dispositivo físico. Isso ocorre porque ficamos na dependência do tamanho físico do *pixel* do dispositivo. Uma medida mais confiável de resolução é dada pela densidade de resolução da imagem que fornece o número de *pixels* por unidade

linear de medida. Em geral se utiliza o número de pixels por polegada, ppi (*pixels per inch*) também chamada de dpi (*dots per inch*) (Velho, 1994).

Chama-se de resolução de cor ao número de bits utilizado para armazenar o vetor de cor de cada pixel da imagem. Tem-se, por exemplo, uma imagem com 256 tons-de-cinza, então são necessários 8 bits para representar a informação de cor de cada pixel (8 *bits*/*pixel*).

O isomorfismo natural entre o espaço das matrizes *m* x *n* é bastante usado em processamento de imagem para identificar a representação matricial de uma imagem digital com um vetor, cujas coordenadas são as linhas de sua representação matricial. A representação de uma imagem por uma matriz *m* x *n* representada na figura 3, ou um vetor de dimensão *m*,*n* representado na figura 4, possibilita o uso de técnicas de álgebra linear no processamento de imagens.

Figura 3 - Matriz m x n - Imagem 20 x 6 (120 pontos).



Fonte: Velho, 1994

Figura 4 - Vetor mn - Imagem 20 x 6 (120 pontos).

Fonte: Velho, 1994

Os elementos de uma imagem digital consistem, essencialmente, das coordenadas dos pixels, e da informação da cor de cada pixel. Esses dois elementos estão diretamente

relacionados com a resolução espacial e resolução de cor da imagem. O número de componentes do pixel é a dimensão do espaço de cor utilizado. Dessa forma, cada pixel em uma imagem monocromática tem um único componente. O *gamute* de uma imagem digital é o conjunto das cores do espaço de cor quantizado da imagem. Uma imagem monocromática cujo gamute possui apenas duas cores é chamada de imagem de dois níveis (em inglês *bilevel*). Uma imagem monocromática cujo gamute possui mais de dois níveis é chamada de imagem com meio-tom, imagem com escala de cinza, (do inglês *halftone* ou *grayscale*) ou tons-de-cinza (França Neto, 1998).

Se o espaço da cor de uma imagem tem dimensão *k*, para grande parte dos processos podemos considerar cada componente de cor em separado. Nesse caso a imagem se reduz a *k* imagens com escala de cinza. Cada uma dessas imagens é chamada de componente da imagem original. É muito comum efetuar operações com imagens trabalhando separadamente com cada componente. O processamento por componentes simplifica bastante determinadas operações. No entanto, esse tipo de processamento não explora a correlação na informação de cor que existe entre os diversos componentes de cada *pixel* da imagem (França Neto, 1998).

2.2 SÍNTESE DE IMAGENS 2D

Para que um computador consiga ler e conseqüentemente reproduzir sua visualização, deve ser feita a descrição de seus objetos tais como segmentos de reta, polígonos, esferas etc. A este procedimento dá-se o nome de síntese de imagem. Portanto, síntese de imagens é uma sub-área da computação gráfica que se ocupa da produção de representações visuais a partir de especificações geométrica e visual de seus componentes (Celani, 1997).

O processo de síntese de imagens digitais consiste em transformar as imagens reais em imagens digitais. Este processo é feito através de dispositivos de entrada gráfica. De acordo com o tipo e destinação da imagem, os dispositivos de entrada gráfica podem ser classificados como vetoriais ou matriciais (Velho, 1994).

2.3 DISPOSITIVOS DE ENTRADA VETORIAL

Os dispositivos de entradas vetoriais são, em sua maioria, utilizados em sistemas interativos, onde o usuário tem uma participação direta com estes dispositivos. Como exemplo

típico de dispositivos vetoriais temos o *mouse*, que serve para fornecer coordenadas para dar entrada nos dados que compõem uma imagem.

Estes dispositivos se dividem em dispositivos de entrada absolutos e dispositivos de entrada relativo. Como exemplos de dispositivos cujo sistema de coordenadas é absoluto tem-se: o *light pen, a tablet, o touch screen, o 3D-digitizer*, entre outros. As características técnicas relevantes dos dispositivos de entrada vetorial absoluto são a sua resolução, linearidade, repetibilidade e área de ação.

Os dispositivos vetoriais mais comuns que operam com referencial relativo são: o *mouse*, a *trackball*, e o *joystick*. Esses dispositivos registram deslocamentos que são transformados em informações de movimento relativo (Celani, 1997).

2.4 DISPOSITIVOS DE ENTRADA MATRICIAL

Os dispositivos de entrada matricial são em geral utilizados de forma não-interativa, desta forma, se adequando mais a aquisição de grandes volumes de dados. Como exemplos mais comuns deste tipo de dispositivos, tem-se o *scanner* e a câmera de vídeo. Este tipo de dispositivo tem uma grande importância no processo de aquisição de grandes volumes de imagens. A estrutura dos dispositivos de entrada do tipo matricial consiste de um sensor que capta sinais no espaço ambiente e um circuito analógico-digital que converte esses sinais analógicos para o formato matricial (Corrigan, 1994).

O processo de conversão de uma imagem para uma imagem digital é conhecido como digitalização. Os dispositivos de entrada matricial são em sua maioria destinados à digitalização de imagens. Dependendo do meio no qual se encontra a imagem a ser digitalizada temos o *frame grabber*, a câmera de vídeo, a câmera fotográfica digital, o *scanner* e o *film scanner* (França Neto, 1998).

O *frame grabber* faz a digitalização a partir de um sinal analógico de vídeo. O sinal de vídeo pode ser gerado diretamente por uma câmera ou por um equipamento de reprodução de vídeo. A resolução geométrica da imagem digitalizada é a resolução de vídeo, que é aproximadamente de 512 x 512 pixeis. Os dispositivos mais sofisticados digitalizam com uma

resolução de cor de 24 bits e usam circuitos *charge-coupled device* (CCD) com mais resolução. O sinal analógico pode ser gerado da seguinte maneira:

a) Sinal de luminância: apenas a luminosidade é capturada, e temos a imagem em tons de cinza;

b) Sinal de vídeo composto colorido: os sinais das cores (RGB) são codificados em um único sinal seguindo um determinado padrão (NTSC, PAL-M, SECAM, etc);

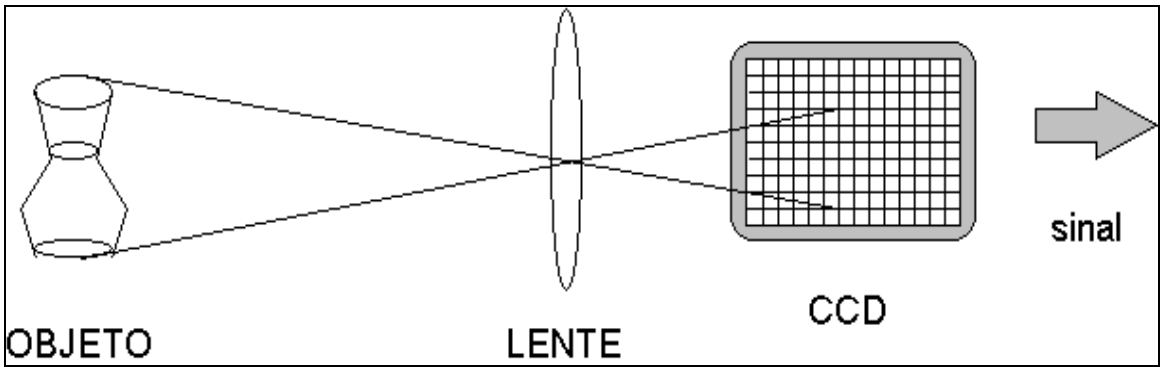
c) Sinal de luminância e crominância ou Y/C (S-video): o sinal é composto por duas partes a luminância e a crominância, como isso a imagem tem uma melhor qualidade do que no vídeo composto. Muito usado por vídeos SVHS, *laser disc*, DVD e outros aparelhos que geram imagens de boa qualidade (acima de 400 linhas);

d) Sinal RGB (*red, green, blue*): o sinal é separado pelas cores básicas, com isso é possível ter uma imagem mais pura. Utilizado em câmeras e gravadores profissionais e imagens geradas por computador, etc.

A câmera de vídeo faz a digitalização a partir de sensores CCD que digitaliza a imagem e a câmera gera um sinal analógico ou digital de vídeo. Existe também os sensores com tecnologia CMOS que tem a vantagem de ser menor, porem deixa a desejar por enquanto na qualidade. Existem vários tipos de câmeras, entre elas tem-se:

a) Câmera de Luminância - capta a imagem em tons de cinza, e gera um sinal composto só com a luminância da imagem. A imagem é gerada por um CCD monocromático que capta o tom de cinza que inside em cada célula do circuito, como pode ser visto na figura 5. É utilizada em geral para aplicações em visão computacional e nos casos onde a informação sobre a luminosidade da imagem é suficiente.

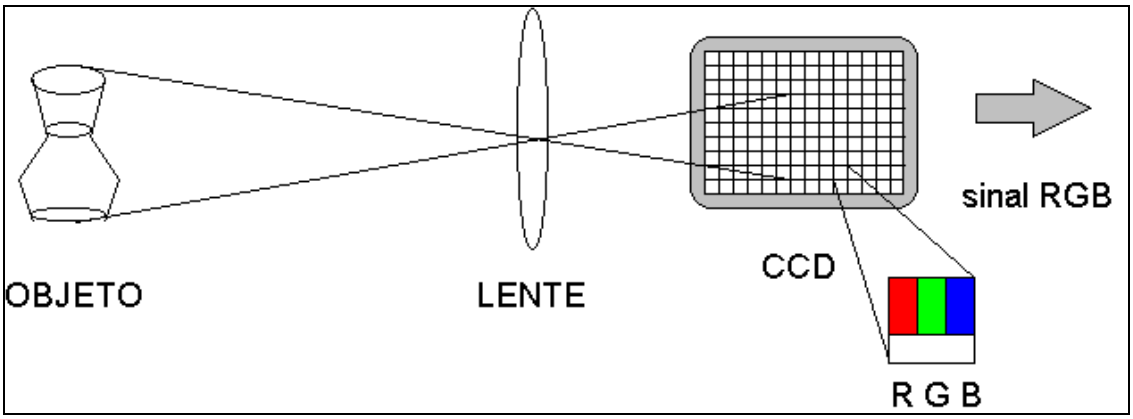
Figura 5 - Câmera de Luminância



Fonte: França Neto, 1998

b) Câmera de crominância (1 passo - 1 CCD): capta a imagem em cores e gera um sinal de vídeo composto colorido, em apenas uma passagem. A imagem, em geral, não é profissional, pois é usado um único CCD com filtros RGB em cada célula, como pode ser visto na figura 6. Com isso, existem espaços nas células onde não é sensível a determinada componente de cor, ou mesmo não tem sensor, o CCD fica com pouca resolução e uma qualidade inadequada para determinadas aplicações. É utilizada para aplicações em multimídia ou em casos onde não são necessárias imagens com muita qualidade. Esta é uma câmera do tipo doméstica (VHS, 8mm, VHS-C etc), desta forma tem um custo baixo.

Figura 6 - Câmera de crominância (1 passo - 1 CCD).

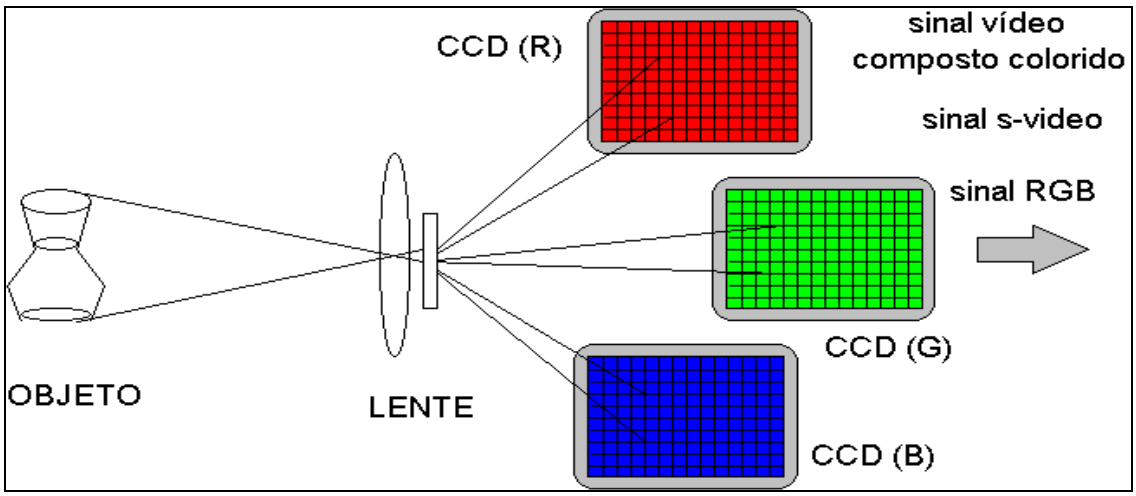


Fonte: França Neto, 1998

c) Câmera de crominância (1 passo - 3 CCD): capta a imagem em cores e pode gerar sinal de vídeo composto colorido, s-video ou sinal RGB. Tem uma qualidade de

imagem profissional, pois são usados 3 (três) CCDs com filtros separados R, G e B em cada um, como pode ser visto na figura 7. O fato de possuir 3 CCDs faz com que a qualidade da imagem seja superior a gerada por uma câmera de um CCD. Por ter 3 CCDs independentes, cada um pode ter uma resolução maior, garantindo uma melhor resolução da imagem. É utilizada em aplicações profissionais, onde são necessárias imagens com boa qualidade. Esta é uma câmera do tipo usado em produtoras e emissoras de TV (U-matic, BetaCAM, SVHS, Hi8 etc), desta forma tem um custo elevado.

Figura 7 - Câmera de crominância (1 passo - 3 CCD).

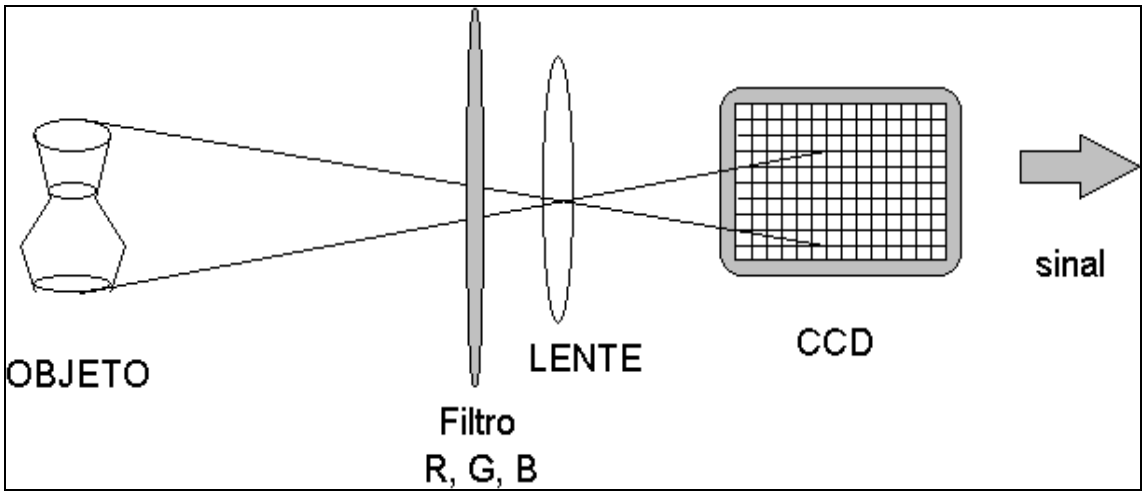


Fonte: França Neto, 1998

d) Câmera de crominância (3 passos - 1 CCD): capta a imagem em cores, porém este processo é feito em 3 passos. É utilizado um único CCD para captar a imagem, sendo que para gerar uma imagem colorida é colocado um filtro externo para cada componente R, G e B. A digitalização da imagem então é feita em 3 passos, ou seja, para cada filtro é feita uma digitalização. Assim temos a informação das intensidades de cada componente RGB. Com esta informação é composta uma imagem colorida, pois para cada ponto temos a contribuição R, G e B. Este processo tem uma desvantagem pelo fato de que as imagens devem ser estáticas, pois é preciso trocar os filtros e fazer nova captação para os outros filtros. Tem uma boa qualidade de imagem, pois este CCD pode ter uma boa resolução, proporcionando uma melhor resolução da imagem. É utilizada em geral para aquisição de imagens de telescópio, onde são necessárias imagens com alta definição e as imagens são relativamente estáticas. Esta

é uma câmera que pode ter um custo baixo, no caso de CCDs de pouca qualidade (baixa resolução), ou de alto custo se o CCD tiver alta resolução (figura 8).

Figura 8 - Câmera de crominância (3 passos - 1 CCD).

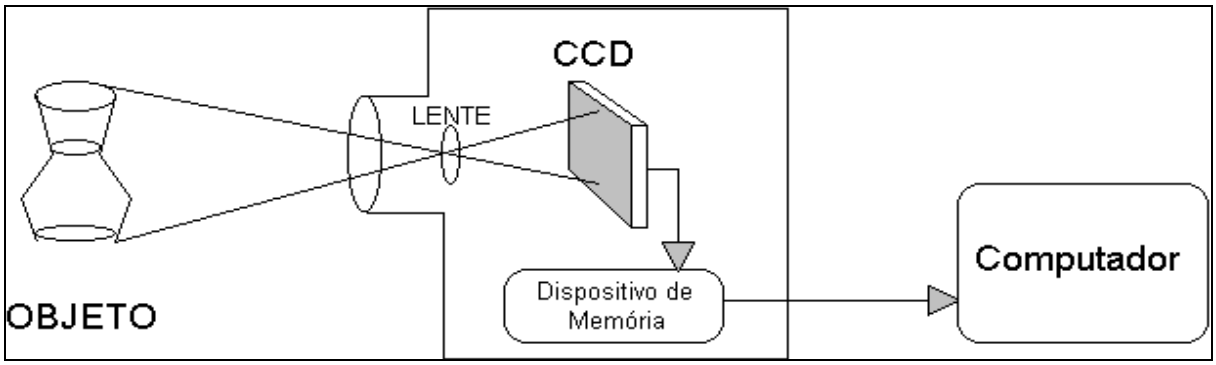


Fonte: França Neto, 1998

A câmera fotográfica digital é um dispositivo que funciona semelhante a uma câmera fotográfica tradicional, porém a imagem não é armazenada em um filme e sim de forma digital em memória. Esta imagem é digitalizada através de um CCD, e armazenada de forma compactada ou não em um dispositivo de memória. A qualidade da imagem depende da qualidade e resolução do CCD e da compressão utilizada para armazenar a imagem digitalizada. O CCD utilizado pode gerar imagens em cores ou tons-de-cinza.

A resolução, atualmente, varia entre 320x240 até algo em torno de 1600x1200 pontos por imagem. A imagem pode ser armazenada em vários tipos de memória como memórias não voláteis, cartões de memória, disquetes magnéticos etc. As informações com as imagens são transferidas para o computador por cabos ou leitores dos dispositivos de memória, e então são processadas, como é visto na figura 9. O custo deste dispositivo pode ser baixo no caso de câmeras domésticas com poucos recursos e baixa resolução, ou muito alto quando a câmera possui recursos profissionais e alta resolução.

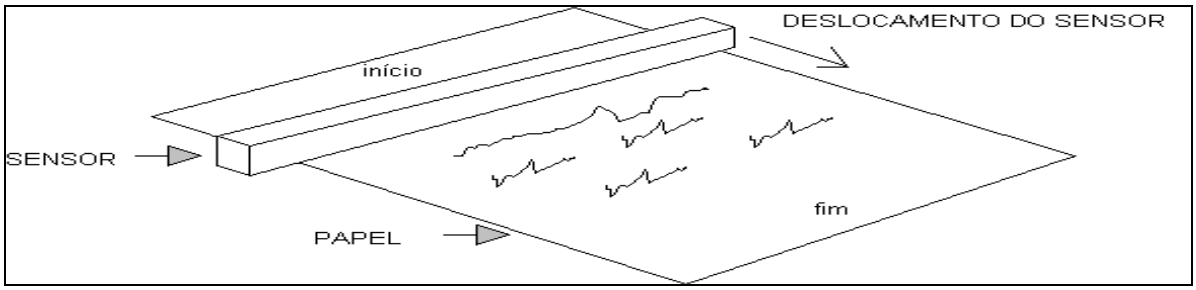
Figura 9 - Câmera fotográfica digital.



Fonte: França Neto, 1998

O *scanner* digitaliza a partir de imagens em papel. A imagem é colocada sobre uma superfície transparente, em geral plana ou cilíndrica, que se move numa direção ortogonal à um elemento de digitalização de linha. Este elemento se compõe de uma fonte de luz e de um sensor que mede a luz refletida linha por linha, em sincronismo com o deslocamento da imagem, ou do sensor. A resolução deste dispositivo está situada entre 50 dpi a 4000 dpi (pontos por polegada). Um dispositivo semelhante é o *film scanner*, que obtêm uma imagem digital a partir de imagens em transparências, utilizando o laser para maior resolução. Este dispositivo também atinge resoluções superiores a 2000 dpi, e seu esquema de funcionamento pode ser visto na figura 10 (França Neto, 1998).

Figura 10 - Esquema de funcionamento do scanner.



Fonte: França Neto, 1998

2.5 IMAGENS DIGITAIS 3D

Segundo Foley (1990), muitas aplicações de Computação Gráfica envolvem a visualização de objetos e cenas 3D. Para exemplificar, os sistemas CAD permitem a manipulação de "automóveis", modelos de componentes de máquinas e partes de um avião; sistemas de

simulação apresentam uma imagem de um mundo tridimensional que se movimenta continuamente, por exemplo, para um piloto de avião. Estas aplicações diferem das aplicações bidimensionais não somente na adição de uma dimensão: elas também exigem um “realismo” na visualização dos objetos.

Na verdade, o maior problema para se fazer síntese de imagens tridimensionais vem do fato de que a maioria dos periféricos de E/S utilizados em Computação Gráfica são próprios para manipular elementos bidimensionais. Esse tipo de limitação leva a utilizar uma série de recursos e técnicas para tentar compatibilizar o universo tridimensional com o qual se deseja trabalhar com as limitações dos dispositivos periféricos (Berg, 2001).

A produção de imagens com realismo de uma cena 3D em um dispositivo 2D, também apresenta muitos problemas: como a terceira dimensão será exibida na tela; como as partes dos objetos que estão "escondidos" por outros objetos serão identificadas e removidas da imagem final; como a iluminação, a cor, a sombra e a textura podem contribuir ao *rendering*; entre outros. Pode-se dizer que quando se trabalha com síntese de imagens tridimensionais facilmente se distingue três fases no processo, como pode ser visto na tabela 1 (Berg, 2001).

Tabela 1 – Fases no processo de síntese de imagens tridimensionais

Modelagem de Objetos	Modelagem de cenas	Visualização de Cenas
é a fase onde deve-se descrever, de alguma forma, os objetos com os quais se trabalhará	é o instanciamento dos objetos em um universo 3D	é a fase onde se procura gerar uma imagem de uma cena 3D anteriormente modelada. Geralmente obtém-se uma projeção 2D da cena 3D original, tal como em uma fotografia

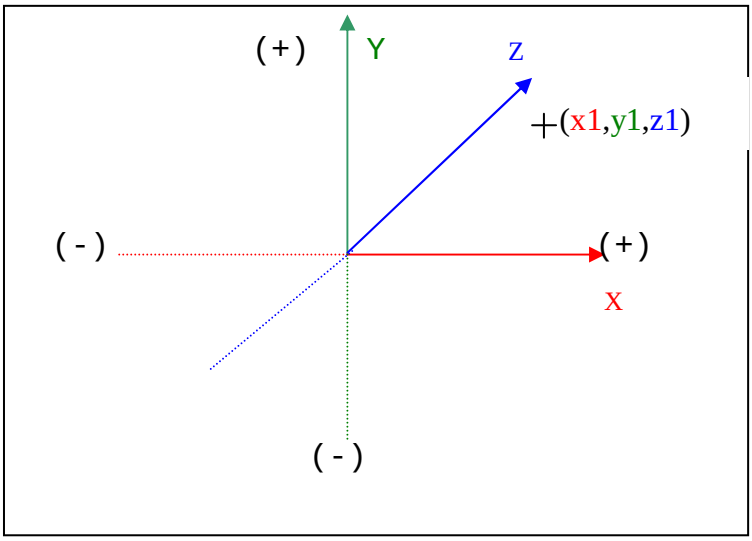
2.6 SISTEMAS DE COORDENADAS 3D

Um desenho é composto de elementos básicos combinados: pontos, segmentos de reta e áreas (espaços preenchidos), cada um com determinadas características como cor, dimensionamento, tipo de traçado (tracejado, pontilhado, etc). A adoção de um sistema de

coordenadas faz-se importante pelo fato de possibilitar a descrição matemática do posicionamento relativo dos elementos que compõem o desenho. O sistema de coordenadas, também denominado sistema de referências, define uma referência (ou origem) em relação à qual se descrevem todos os posicionamentos (Casacurta, 1998).

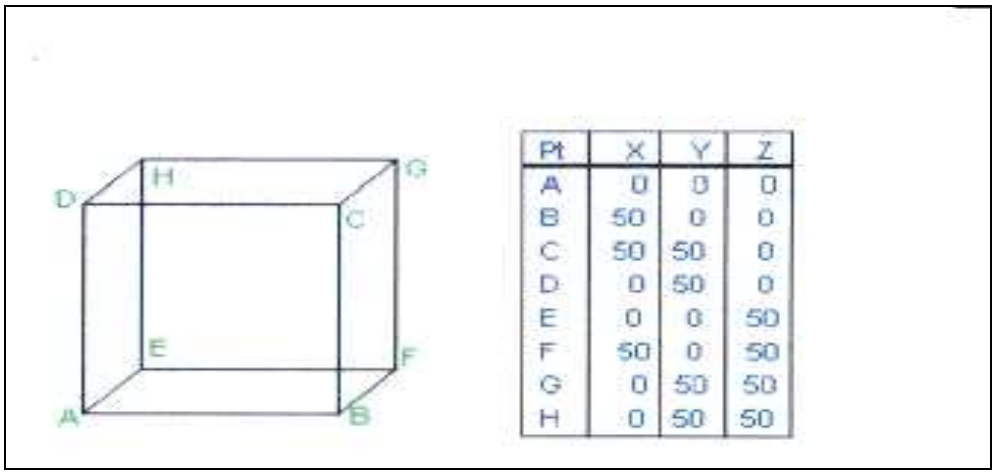
Os modelos tridimensionais são descritos através de suas coordenadas espaciais, ou seja, coordenadas X, Y e Z. O sistema de coordenadas espaciais pode ser visto na figura 11.

Figura 11 – Sistema de coordenadas espaciais.



Nos modelos de três dimensões além das entidades ponto e aresta surge à entidade face, que é uma seqüência ordenada de pontos e arestas. O sentido correto da seqüência dos pontos será importante para os algoritmos de remoção de elementos ocultos. Tem-se com um exemplo mais simples, um cubo (com sua definição em termos de coordenadas espaciais) representado na figura 12. Nesta definição, o cubo está sendo representado apenas pelos vértices que compõem as suas extremidades.

Figura 12 – Representação do cubo em modo aramado.



Pt	X	Y	Z
A	0	0	0
B	50	0	0
C	50	50	0
D	0	50	0
E	0	0	50
F	50	0	50
G	0	50	50
H	0	50	50

Fonte: Casacurta, 1998

A figura 12 representa um cubo tridimensional, onde a tabela fornece as coordenadas de cada vértice do cubo. Esta representação é denominada de aramada, pois não tem-se o conceito de faces. A descrição final do cubo será dada em função apenas de seus vértices e das arestas que unem um vértice ao outro. Uma descrição mais completa deveria indicar: os vértices, as arestas que unem os vértices, e as faces que são compostas por um conjunto de arestas. Como a representada na figura 13 (Casacurta, 1998).

Figura 13 – Estrutura de dados 3D.

VERTICE	X	Y	Z
A	0	0	0
B	50	0	0
C	50	50	0
D	0	50	0
E	0	0	50
F	50	0	50
G	50	50	50
H	0	50	50

ARESTA	VERTICES
A1	A,B
A2	B,C
A3	C,D
A4	D,A
A5	E,F
A6	F,G
A7	G,H
A8	H,E
A9	B,F
A10	C,G
A11	D,H
A12	A,E

FACE	VERTICES
F1	A,B,C,D
F2	B,F,G,C
F3	E,F,G,H
F4	A,E,H,D
F5	A,B,F,E
F6	D,C,G,H

Fonte: Casacurta, 1998

A descrição aramada está nas tabelas de vértices e arestas, e a descrição completa está nas tabelas de vértices, arestas e faces.

Uma vez que o objeto já esteja modelado, ou seja, possua uma descrição de sua estrutura, pode-se começar a pensar na fase de visualização.

3 TÉCNICAS DE MODELAGEM 3D

Os três principais tipos de modelos geométricos são o aramado (*wireframe*), faces limitantes e sólidos. A modelagem de sólidos é um importante aspecto da modelagem geométrica que é usada para criar e comunicar informações sobre sua forma. Ela envolve a criação e manutenção do modelo sólido para futuro acesso e análise. Os primeiros sistemas de projeto assistido por computador (CAD) representavam desenhos (entidades bidimensionais) ao invés de sólidos tridimensionais. Em tais sistemas, além de usar esquemas ambíguos, as mudanças feitas em uma vista do objeto não se propagavam automaticamente para as outras vistas porque o sistema não sabia que as várias vistas eram projeções do mesmo objeto (Casacurta, 1998).

As representações de sólidos foram classificadas em ambíguas e não-ambíguas. As representações não-ambíguas são caracterizadas por serem completas e na sua maioria também por serem únicas. Os métodos de representação de sólidos denominam também as técnicas de modelagem de sólidos, que são as seguintes:

a) instanciamento de primitivas;

b) enumeração de ocupação espacial;

c) decomposição em células;

d) geometria sólida construtiva (*Construtive Solid Geometry* - CSG);

e) representação por *sweeping*;

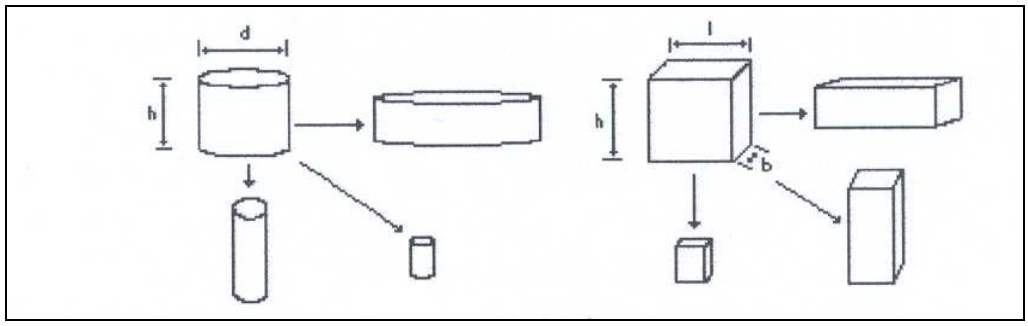
f) representação por fronteira.

3.1 INSTANCIAMENTO DE PRIMITIVAS

Essa técnica é baseada na noção de famílias de objetos onde cada membro da família é distinguido por alguns parâmetros. Cada família de objetos é chamada de primitiva genérica, e cada objeto dentro de uma família é chamado de instância primitiva.

Formas parametrizadas constituem um caso especial de instâncias, pois a forma do objeto instanciado muda de acordo com o valor dos parâmetros. Tem-se na figura 14 um exemplo de instâncias (Casacurta, 1998).

Figura 14 –Exemplo de Instâncias.



Fonte: Casacurta, 1998

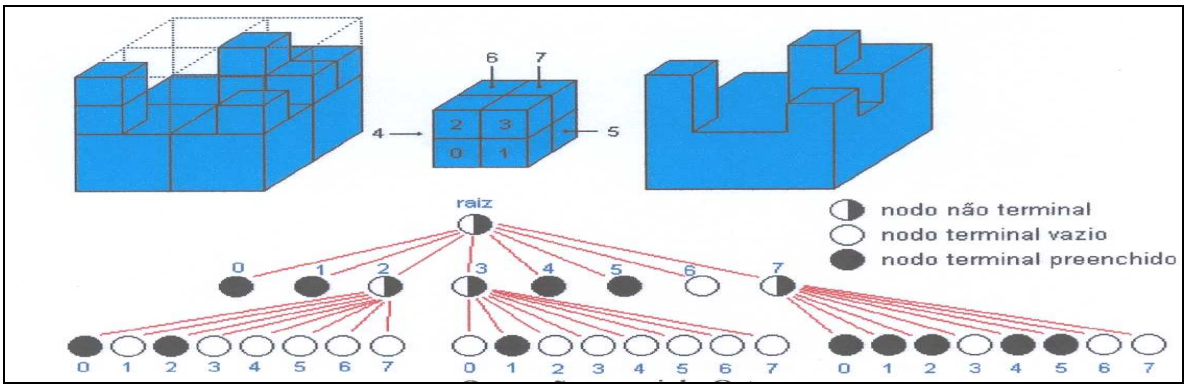
3.2 OCUPAÇÃO ESPACIAL

A representação do sólido no esquema de ocupação espacial é essencialmente uma lista de células espaciais ocupadas pelo sólido.

As células, algumas vezes chamadas de *voxels*, são cubos de um tamanho fixo e estão em uma rede espacial fixa. Cada célula pode ser representada por coordenadas de um ponto, tal como o centro da célula. Usualmente uma ordem específica de exploração espacial é imposta, onde o correspondente conjunto ordenado de triplas é chamado de arranjo espacial.

Um dos métodos mais conhecidos de representação por ocupação espacial é a *octree*. A representação *octree* é uma generalização tridimensional de uma árvore binária, como pode ser visto na figura 15 (Casacurta,1998).

Figura 15 – Ocupação espacial *Octree.*

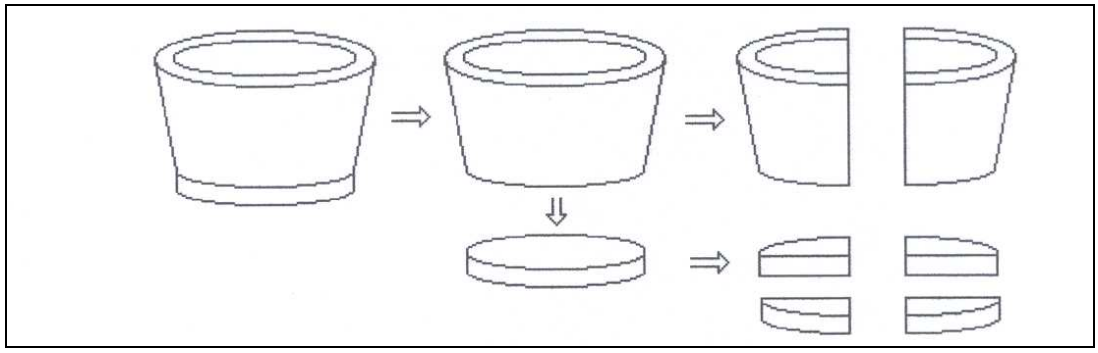


Fonte: Casacurta, 1998

3.3 DECOMPOSIÇÃO EM CÉLULAS

Um sólido pode ser representado pela sua decomposição em células como é mostrado na figura 16. Esquemas de ocupação espacial são um caso particular em que todas as células no esquema devem ser cúbicas e estar em uma rede espacial fixa (Casacurta,1998).

Figura 16 - Decomposição em células.



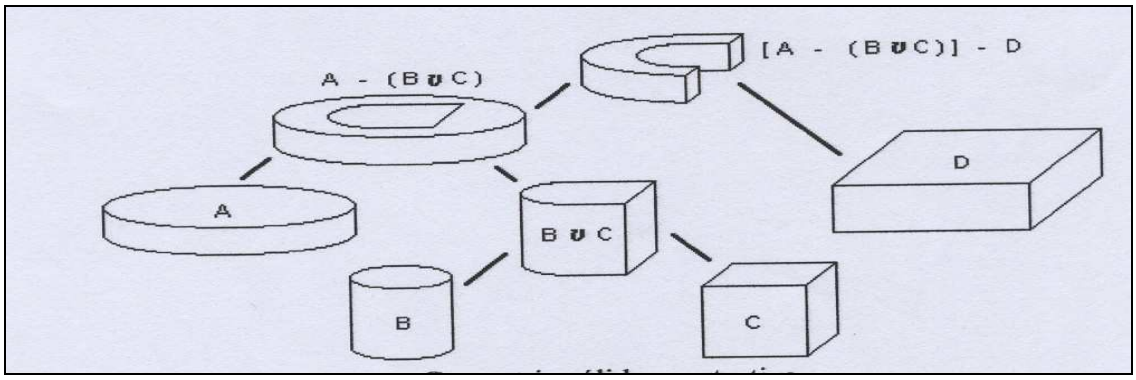
Fonte: Casacurta, 1998

3.4 GEOMETRIA SÓLIDA CONSTRUTIVA (CSG)

A geometria sólida construtiva sugere um esquema de representação de sólidos através de construções booleanas ou combinações de componentes sólidos a partir de operadores de conjuntos (soma, subtração e interseção).

A representação CSG é uma árvore binária ordenada. Os nodos intermediários representam operadores que podem ser movimentos rígidos, uniões, interseções ou diferenças. Os nodos terminais são folhas primitivas que representam subconjuntos do espaço tridimensional ou de transformações que contenham os argumentos de definição de movimentos rígidos. A representação CSG está representada na figura 17 (Casacurta,1998).

Figura 17 - Geometria sólida construtiva.



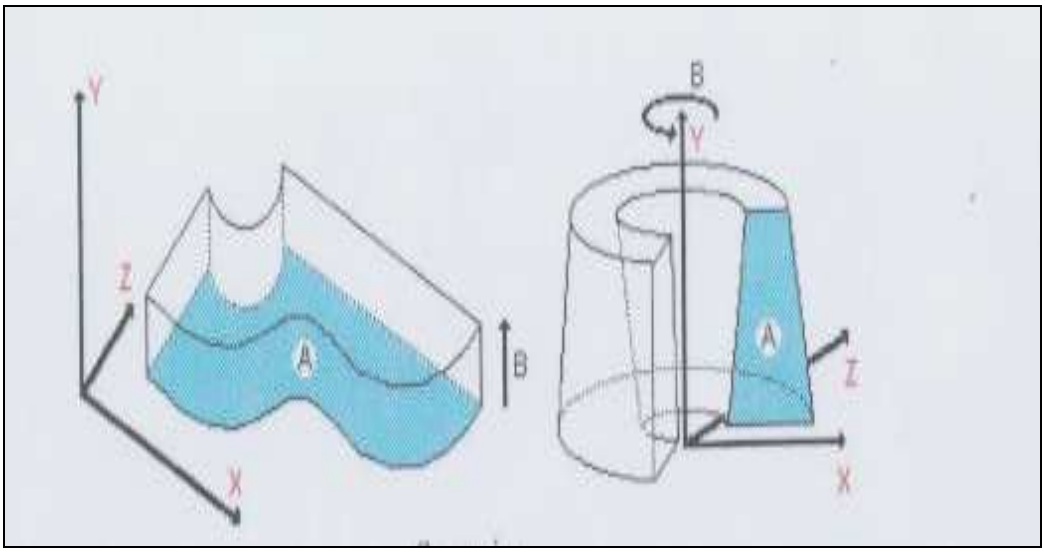
Fonte: Casacurta, 1998

3.5 SWEEPING

A noção básica embutida nos esquemas de varredura (*sweep*) é muito simples: um conjunto de pontos movendo-se através do espaço pode varrer um volume (um sólido) que pode ser representado por um objeto em movimento mais uma trajetória. O local dos pontos gerados por esse processo define um objeto de uma, duas ou três dimensões. Para modelagem de sólidos, a representação por *sweep* é a mais simples de ser compreendida.

As propriedades dos esquemas de *sweeping* translacional e rotacional são em grande parte determinadas por esquemas específicos usados para representar conjuntos bidimensionais. O *sweeping* rotacional é usado por muitos Sistemas de Modelagem de Sólidos que tratam exclusivamente com peças que giram, e *sweeping* translacional é usado nos sistemas para descrever peças mecânicas fabricadas a partir de chapas. Um exemplo claro de *sweeping* pode ser visto na figura 18 (Casacurta, 1998).

Figura 18 – Sweeping.



3.6 FACES LIMITANTES

O método de modelagem de Faces Limitantes também é chamado de Representação por *Boundary-Representation*. Neste tipo de modelagem apenas a superfície do objeto é armazenada, aliada a uma função que determina se um ponto está dentro ou fora do sólido. Essa função é necessária para cálculos que envolvam a região interior do objeto.

A superfície limitante de um sólido separa os pontos de dentro do sólido dos pontos de fora. É esta superfície que determina a interface entre o sólido e o ambiente a sua volta. Todas as características visuais do sólido tais como reflexão, transparência, textura e cor são características dessa superfície. Entre as diversas maneiras de se representar a superfície limitante de um sólido uma das mais comuns é através de faces (Berg, 2001).

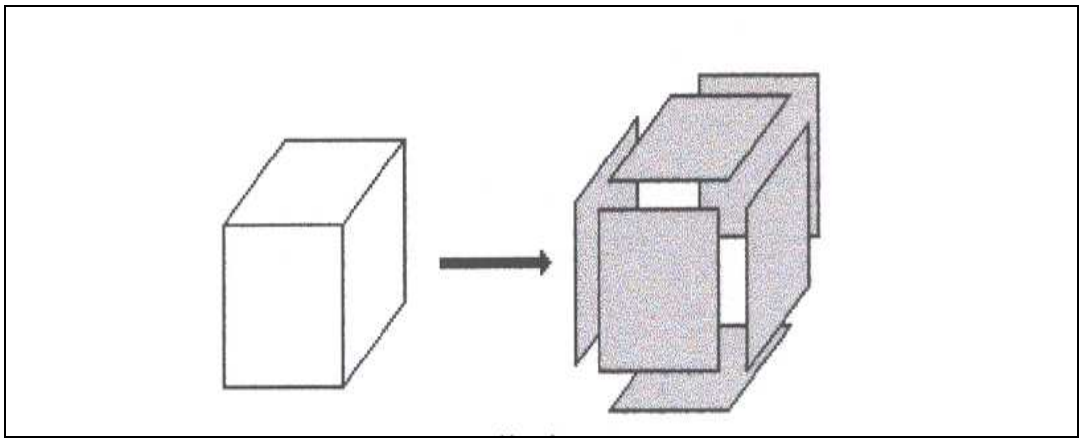
Segundo Berg (2001) um sólido representado através de faces deve respeitar um conjunto mínimo de restrições, descritas a seguir.

a. Um número finito de faces definem a superfície do sólido;
b. Uma face do sólido corresponde a um sub-conjunto da superfície limitante do mesmo;
c. A união de todas as faces do objeto define sua superfície limitante;
d. Cada uma das faces é uma região limitada de alguma superfície maior;

e. Cada face deve ter uma área finita e ser dimensionalmente homogênea.

Nesse esquema de representação, o sólido é definido pelos segmentos de sua fronteira; um número finito de subconjuntos limitados, usualmente chamados de faces ou *patches*, onde cada face é representada por suas arestas e vértices limitantes. Como pode ser visto na figura 19 (Casacurta,1998).

Figura 19 - Faces limitantes.



Fonte: Casacurta, 1998

3.7 WIRE-FRAME

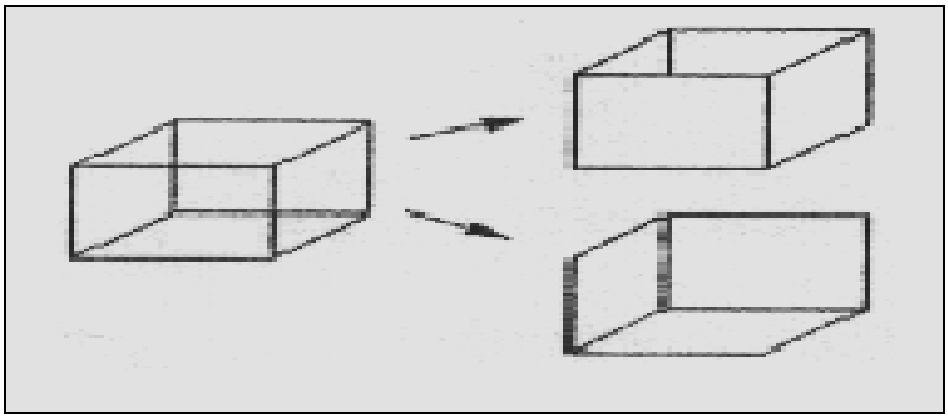
Segundo Berg (2001) na grande maioria das aplicações de computação gráfica existe uma fase de criação e manipulação do modelo onde o grau de realismo não precisa ser muito grande. Neste caso, representam-se apenas as arestas do objeto, de tal forma que o projetista possa vê-la e possa alterar qualquer ponto do desenho, ou seja, representa-se o modelo *wire-frame* (ou aramado) do objeto.

O modelo *wire-frame* de um objeto consiste apenas de pontos, linhas e curvas. O uso de modelos *wire-frame* puros apresenta uma série de desvantagens, entre elas:

a. A possibilidade de se criarem objetos sem sentido;

b. A possibilidade da dupla interpretação, pois não permite a identificação do objeto como um sólido, como mostra a figura 20.

Figura 20 – Wire-Frame (Aramado).



Fonte: Berg, 2001

3.8 MODELAGEM PROCEDURAL

Segundo Berg (2001) a modelagem procedural é de uma área de pesquisa mais recente em Computação Gráfica que engloba uma série de métodos alternativos à modelagem geométrica tradicional. Uma das principais motivações nessa área tem sido o desafio de representar a complexidade dos objetos do mundo real tanto em termos da sua forma quanto do seu comportamento.

Modelos procedurais podem descrever:

a. Objetos que podem interagir com eventos externos para se modificarem. Como exemplos de objetos nessa classe pode-se citar: terreno, vegetação, gases, líquidos, fogo e os próprios animais. Essa classe de objetos se caracteriza por formas naturais que são muito familiares ao ser humano. Porém, em geral, são difíceis de representar tanto do ponto de vista da forma como do movimento.

b. A geometria em função de uma série de parâmetros que variam com o tempo. Neste caso o modelo procedural passa a ser ativo em oposição ao modelo sólido que é representado por configurações estáticas e passivas. Por exemplo: o modelo procedural de uma ponte, que consiste em uma estrada "dividida ao meio", uma estrutura, pilares e parapeitos, é especificado através das descrições destes elementos de acordo com uma orientação para determinar a posição da ponte. Cada

> um dos seus componentes, por sua vez, é especificado por um número de parâmetros e o procedimento então gera o modelo a partir destas descrições. Torna-se importante salientar que o modelo gerado não precisa obrigatoriamente ser composto de uma coleção de sólidos, e que a especificação de apenas alguns parâmetros levam à criação de um grande modelo.

Resumindo, a modelagem procedural consiste no desenvolvimento de um procedimento que, baseando-se nos parâmetros recebidos, irá construir um modelo. O "poder" deste tipo de modelagem está no fato de que pequenas mudanças nas especificações (parâmetros) resultam em drásticas mudanças na forma do modelo.

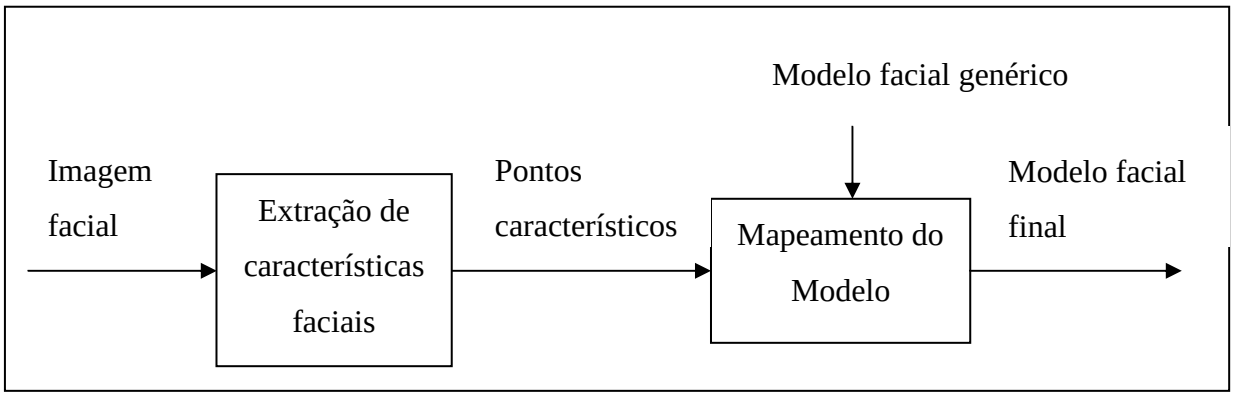
4 TRANSFORMAÇÕES DE IMAGENS FACIAIS RASTER 2D PARA MODELOS 3D

Segundo Huang (1996) os sistemas de codificação baseados em modelos atraíram grandes interesses em telecomunicações visuais e teleconferência, tendo como motivo principal o alcance de uma baixa taxa de codificação de vídeo. Visto que as faces humanas são a parte principal nas cenas em telecomunicações visuais e teleconferência, tem-se como proposta a construção de um modelo para faces humanas em 3D, onde são analisados somente as partes salientes da face e os parâmetros de movimento faciais. Finalmente as expressões faciais originais poderiam ser sintetizadas deformando o modelo facial que usa os parâmetros de movimento faciais. Assim, um fator de compressão muito alto poderia ser alcançado.

Um dos problemas básicos relacionados a sistemas de codificação baseados em modelos é como criar um modelo facial 3D de uma pessoa usando várias imagens faciais 2D, em particular, as imagens do perfil frontal, direito e equerdo do rosto. Normalmente isto pode ser feito ajustando um modelo de face genérico para comparar imagens de acordo com várias características faciais. Foram desenvolvidos vários algoritmos de modelagem automática, conseguindo-se que algumas limitações bastante difíceis fossem solucionadas. Outros algoritmos tentam achar características faciais usando minimização de esforço em *splines*, como *snakes* e *deformable templates*. Estes normalmente conseguem assumir posições iniciais de acordo com as características faciais (Huang, 1996).

Acredita-se fortemente que as características faciais podem ser bem definidas somente quando são vistas como um todo. Um padrão de imagem só é considerado quando, por exemplo, uma boca satisfazer o arranjo global da boca. Desta forma motiva-se o desenvolvimento de um algoritmo de extração automática de características faciais e também um algoritmo de contorno de modelo para construção de modelos faciais 3D. O algoritmo de extração de características faciais localiza automaticamente todos os pontos de característica faciais (pontos de controle). Depois, é aplicada uma série de transformações geométricas que construirá o modelo genérico de acordo com as características apontadas, podendo assim obter-se um modelo facial específico. A Figura 21 esboça o procedimento de modelagem facial automática (Huang, 1996).

Figura 21 – Processo de modelagem facial geral.



4.1 ALGORITMO GENÉTICO FGPFM

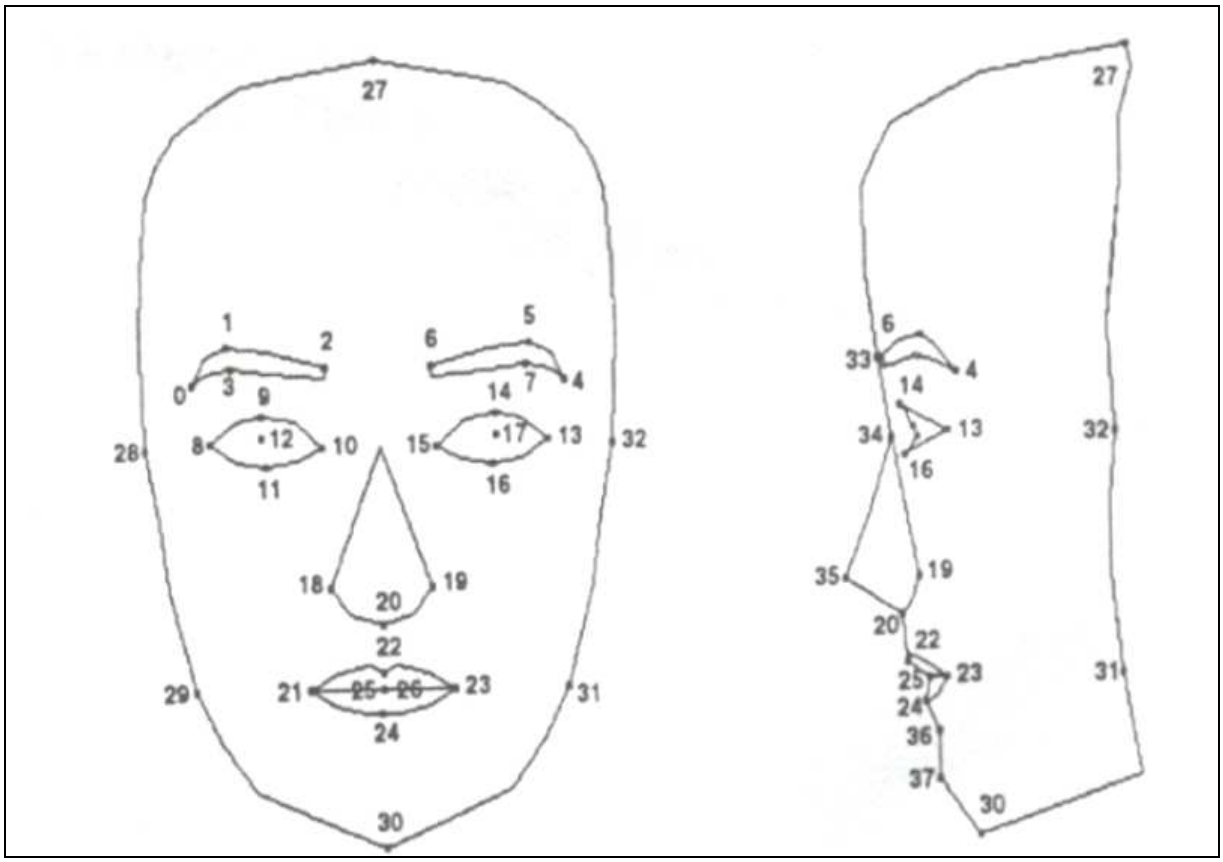
É uma abordagem de um algoritmo genético para modelagem de faces, que modela a partir de uma imagem de face descalibrada, e também toma como parâmetro um modelo facial genérico flexível (FGPFM). O FGPFM pode ser facilmente modificado, usando as características faciais como parâmetros para o FGPFM construir com precisão um modelo facial 3D específico que utiliza uma fotografia de um indivíduo (Ho, 2000).

O problema da modelagem facial é formulado como um problema de otimização de parâmetro. Além disso, uma abordagem grosso-para-fino baseado em um algoritmo genético inteligente que pode resolver os grandes problemas de otimização de parâmetro eficazmente é usada para apressar a procura da melhor solução. Além disso, a análise de sensibilidade e resultados experimentais com o traçado da textura demonstram a efetividade do método proposto (Ho, 2000).

4.2 ALGORITMO DE EXTRAÇÃO AUTOMÁTICA DE CARACTERÍSTICAS FACIAIS

Características faciais são os aparecimentos salientes na face humana, como os olhos, boca, etc. Com a finalidade de modelar a face, precisa-se também extrair locais ao redor dos pontos chave das características faciais. Figura 22 mostra todas as características apontadas por este algoritmo. Estes pontos de características são um subconjunto dos marcos faciais definidos na face humana para uso na antropometria da cabeça humana (Huang,1996).

Figura 22 – Pontos característicos da face humana.

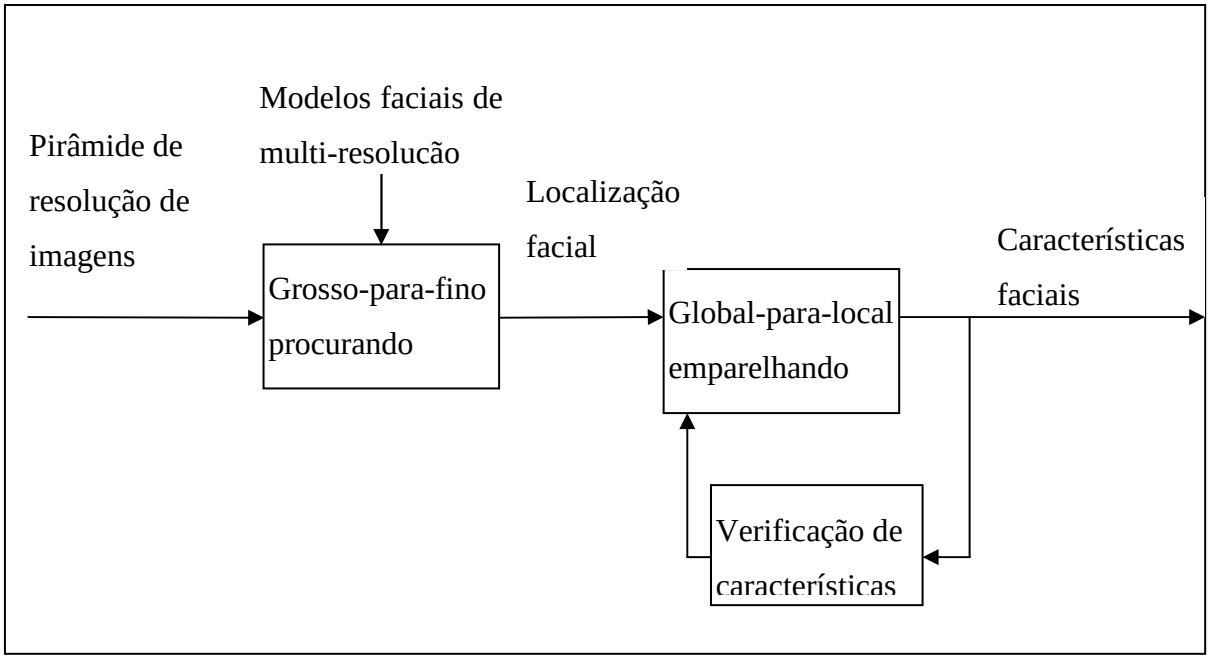


Fonte: Huang, 1996

O algoritmo é uma integração de procura grosso-para-fino (descoberta de face) e emparelhamento global-para-local (extração de característica). Uma série de modelos com multiresoluções são construídos para a face inteira e também para as características faciais individuais. Uma estrutura de pirâmide de resolução também é estabelecida para a imagem facial de entrada. Este algoritmo primeiro tenta achar os locais faciais salientes na imagem à mais baixa resolução emparelhando isto globalmente com os modelos faciais.

As imagens com resoluções mais altas e modelos são usados para refinar o local da face. Então cada característica facial é situada usando uma combinação de técnicas, inclusive processamento de imagens, emparelhamento de modelos e deformação de modelos. Finalmente, um procedimento de avaliação é executado para verificar características que usam a antropometria de faces humanas e se necessário, as características serão emparelhadas novamente. A Figura 23 dá uma visão sistemática deste algoritmo (Huang,1996).

Figura 23 – Sistema de extração automática de características faciais.



O mesmo procedimento é aplicado nas visões de perfis laterais.

São impostas as seguintes condições para fazer este algoritmo funcionar:

1. Assume-se que as imagens faciais têm orientação conhecida, visão dianteira ou visão de perfil lateral. Porém, não se faz nenhuma suposição sobre o local da face na imagem.

2. As características faciais não devem ser bloqueadas por outros objetos como cabelos longos, bigode volumoso e barba.

3. A pessoa não deve usar óculos. De fato, esta também é uma exigência básica para face que será modelada desde que não esteja usando um modelo facial com óculos.

4. A face deve ter uma expressão neutra.

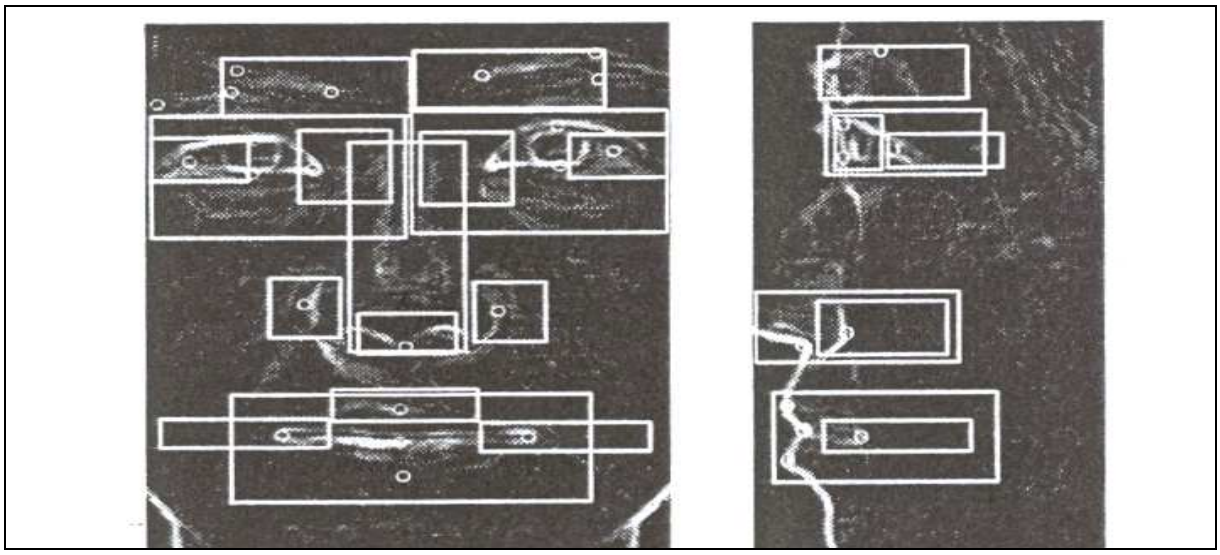
5. Como para o fundo, não há suposição necessária de uniformidade para área de detecção da face e extração das características faciais. Porém, o algoritmo requer que o fundo ao redor do limite da face (na frente e visões laterais) esteja em contraste afinado com a área da face, de forma que a característica apontada no limite da face possa ser determinada.

4.2.1 Multiresolução de Modelos

Uma área que cobre principalmente as sobrancelhas, olhos, nariz e boca são selecionadas como o modelo facial. Como mostra a Figura 24 onde a versão de medida de variação da imagem facial é usada.

Áreas menores ao redor de características faciais são escolhidas como os modelos de características. Modelos que cobrem áreas maiores são usados para localizar globalmente as características de modelos menores e usados para determinar os pontos de característica. Os locais deste sub-modelos também são mostrados dentro da Figura 24. Todos os pontos característicos mostrados na Figura 24 podem ser predefinidos no modelo facial e servir como uma estimativa inicial de características faciais onde a face é detectada em uma imagem (Huang,1996).

Figura 24 – Modelos faciais.



Fonte: Huang, 1996

Uma série de modelos faciais de multiresolução são criados por consolidação de *pixel*. Esta série de modelos faciais é usado para extrair os pontos de características faciais para imagens de entrada na experiência.

Em pouco tempo, a pirâmide de resolução é criada também para as imagens faciais.

4.2.2 Detector da Face

Freqüentemente imagens faciais podem ser detectadas mais facilmente em imagens com baixas resoluções. Há duas razões para isto. Primeiro, as computações são muito mais rápidas devido à redução de dimensionalidade. Segundo, detalhes confusos apresentados em imagens com resoluções mais altas podem não aparecer em resoluções reduzidas. Embora os tamanhos e proporções entre características faciais diferem significativamente em imagens com alta resolução, todas estas imagens permitem gerar várias imagens com baixa resolução. Porém, extração de características faciais requer detalhes que só são revelados em imagens com alta resolução. Então, uma estratégia minuciosa chamada "procurando grosso-para-fino" é introduzida para detectar a área facial antes de fazer o emparelhamento do modelo de multi-resolução (Huang,1996).

A procura começa do nível mais alto (com a mais baixa resolução) da pirâmide de resolução da imagem facial. Todo modelo facial com o tamanho menor que o da imagem facial é emparelhado dentro da imagem facial inteira para adquirir o coeficiente máximo de correlação, é também registrada sua posição na imagem. O processamento dos modelos e um grande coeficiente de correlação são o melhor emparelhamento. Deste modo, podem ser achados a posição saliente e o tamanho aproximado da face na imagem. Desde que à mais baixa resolução, a imagem e o modelo são pequenos, e a procura pode ser feita rapidamente embora uma procura completa ao longo da imagem facial inteira seja requerida nesta fase. Então a procura é implementada ao nível mais interno da pirâmide de resolução, o tamanho da imagem é dobrado. Também se escolhe o modelo que tem a resolução consideravelmente aumentada por um fator de 2 por um previamente. Junto com dois modelos vizinhos, os três modelos são comparados novamente com a imagem de resolução mais alta. Neste passo, as correlações são implementadas somente em uma área de procura pequena ao redor do local facial saliente. Um coeficiente de

correlação máximo é escolhido para refinar o local facial e o tamanho. Este procedimento é repetido até que a imagem de resolução completa seja processada (Huang,1996).

4.2.3 Extração das características faciais

Um possível local da característica inteira deve ser determinado antes que os pontos de característica sejam extraídos. Isto conduz ao esquema "emparelhando global-para-local" que é adotado para extrair todos os pontos de característica. Sendo que as áreas de característica locais envolvem mais detalhes, este emparelhamento normalmente é implementado na imagem facial original (Huang,1996).

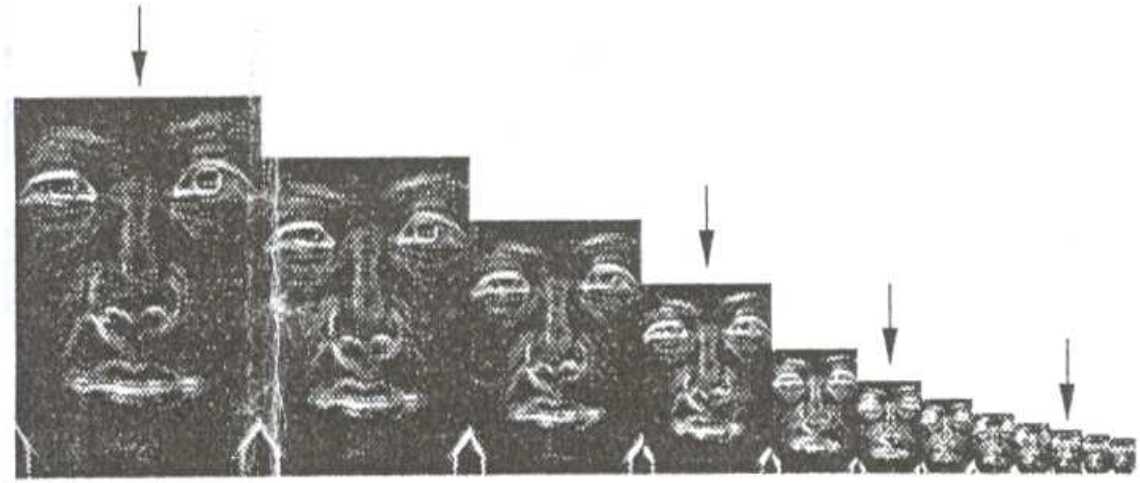
Depois que o local facial foi detectado, a posição inicial de todas as características faciais pode ser calculada imediatamente de acordo com o modelo facial. As características faciais calculadas não devem estar distantes das posições atuais das mesmas. Então, para cada característica facial, primeiro tenta-se achar seu local global emparelhando um modelo que inclui a característica inteira com a imagem da face em uma área de procura designada ao redor da posição inicial. A área de procura para cada característica é predefinida baseada nas proporções entre características faciais. Depois que algumas características faciais foram determinadas, podem ser decididas às áreas de procura para o resto das características com maior precisão. São usados modelos de área menores para achar locais mais precisos dos pontos de característica (Huang,1996).

Finalmente, uma estratégia de avaliação é introduzida para verificar a extração de pontos característicos, para assegurar que eles estão todos corretamente identificados. São usados dois critérios, a simetria e as medidas de antropometria para testar as características apontadas e se necessário, as características serão re-emparelhadas (Huang,1996).

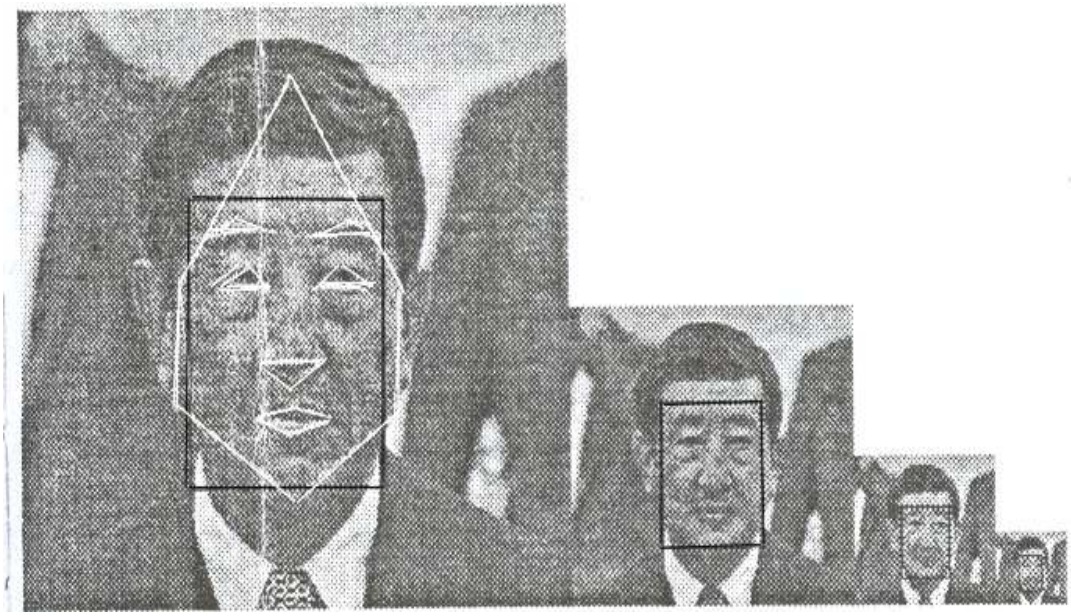
A Figura 25 mostra os resultados da extração de característica. Os vértices dos polígonos branco na Figura 25(b) demonstra os pontos de característica. As setas na Figura 25(a) demonstra o modelo melhor emparelhado a cada nível de procura.

Figura 25 – Extração de características faciais.

(a) – Modelos faciais de multiresolução.



(b) – Pirâmide de resolução das imagens faciais.

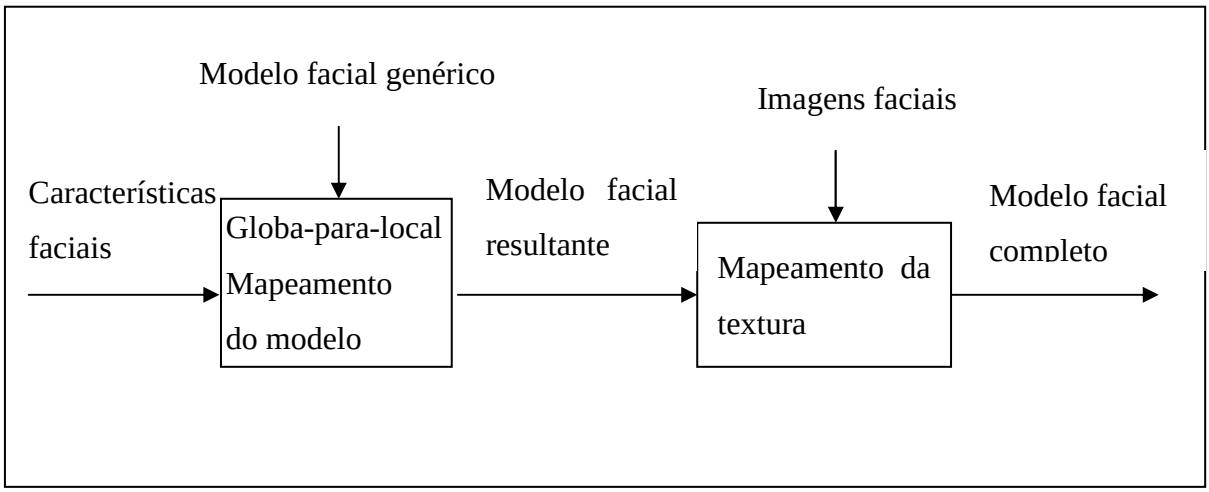


Fonte: Huang, 1996

4.2.4 Mapeando modelo facial genérico

Tendo extraído todos os pontos de característica faciais, um modelo facial 3D personalizado pode ser criado deformando o modelo facial genérico que usa procedimento de contorno “global-para-local”, que é uma série de transformações geométricas. Serão traçados os vértices de controle no modelo facial genérico exatamente aos pontos de característica correspondentes através de tais transformações. Finalmente, um modelo facial realístico que usa as imagens faciais 2D pode ser criado mapeando a textura. A Figura 26 esboça o processo de modelagem facial (Huang,1996).

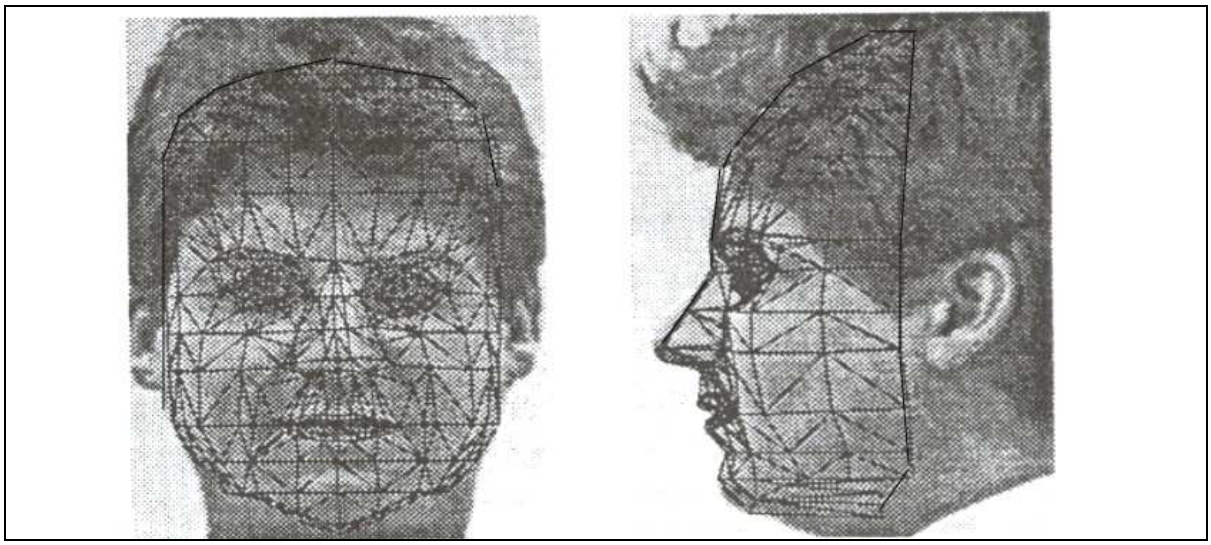
Figura 26 – Processo de mapeamento do modelo facial.



4.2.5 Mapeamento do modelo “Global-para-Local”

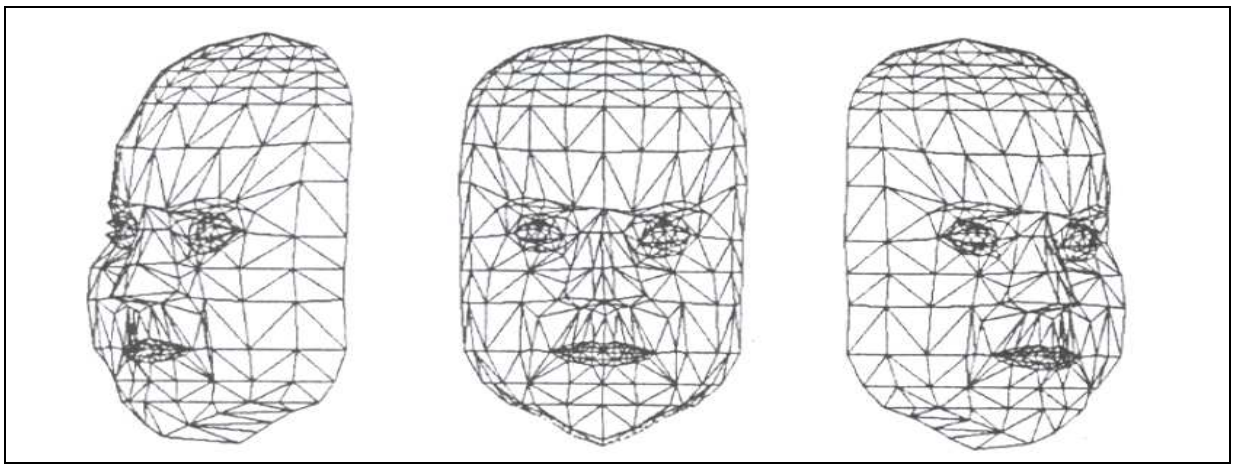
O modelo genérico é mapeado primeiro globalmente para ajustar os limites faciais. Então cada uma das características faciais (olhos, nariz, boca etc) é traçada para ajustar os pontos de característica localmente. Finalmente, são combinados modelos faciais mapeados em visões diferentes das imagens faciais para adquirir um modelo facial 3D completo. Figura 27 mostra um exemplo de mapeamento do modelo facial genérico através duas visões da imagem facial de uma pessoa. O modelo de face resultante pode ser visto em visões diferentes como pode ser visto na Figura 28 (Huang,1996).

Figura 27 – Modelos faciais mapeados.



Fonte: Huang, 1996

Figura 28 – Múltiplas visões de modelos faciais mapeados.



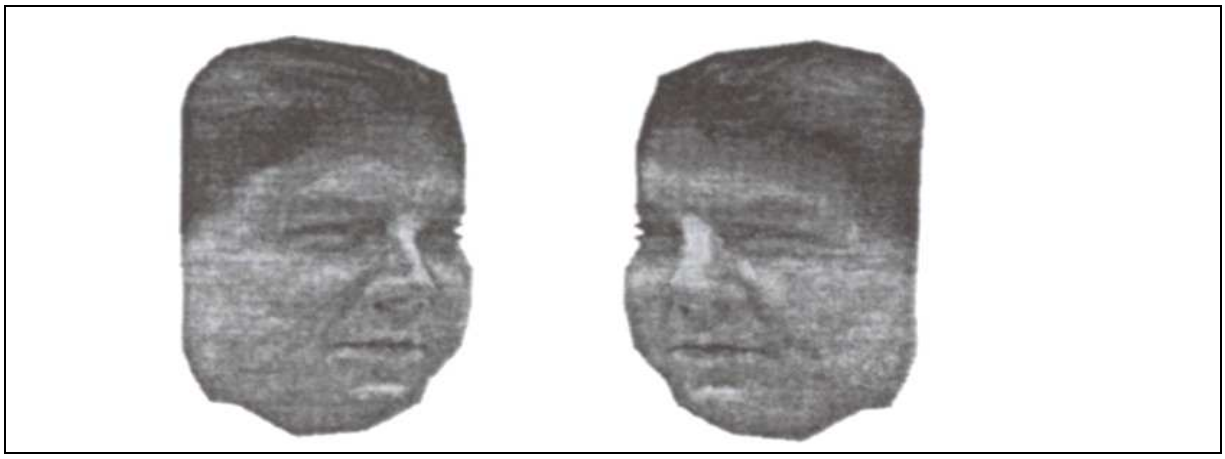
Fonte: Huang, 1996

4.2.6 Mapeamento da textura

O mapeamento geométrico facial tem como resultado um modelo aramado 3D que representa a forma facial de uma pessoa. Para ver este modelo como uma face humana realística, alguma textura deve ser usada para mapear as superfícies de polígono. Um algoritmo de subdivisão semelhante ao de Catmull (Zoz, 1999) (é desenvolvido para que seja mapeada a textura no sistema. A idéia principal é sub-dividir interativamente os triângulos no modelo facial

3D e os triângulos correspondentes no mapa de textura 2D, até cada triângulo alcançar designado tamanho. O valor da textura a cada ponto final do triângulo na imagem facial 2D é nomeada ao vértice correspondente no modelo facial 3D. O interior de cada triângulo é renderizado usando a técnica de sombreamento *Gouraud* (Foley, 1990). A Figura 29 mostra um exemplo de textura que o modelo facial mapeou (Huang, 1996).

Figura 29 – Uma textura mapeada no modelo facial.



5 DESENVOLVIMENTO DO PROTÓTIPO

Com base nos conceitos apresentados nos capítulos anteriores, tornou-se possível o desenvolvimento do protótipo de *software* que permite gerar um modelo de uma face humana em 3D, a partir de imagens 2D. Neste capítulo serão abordados a especificação, a implementação e o funcionamento do protótipo.

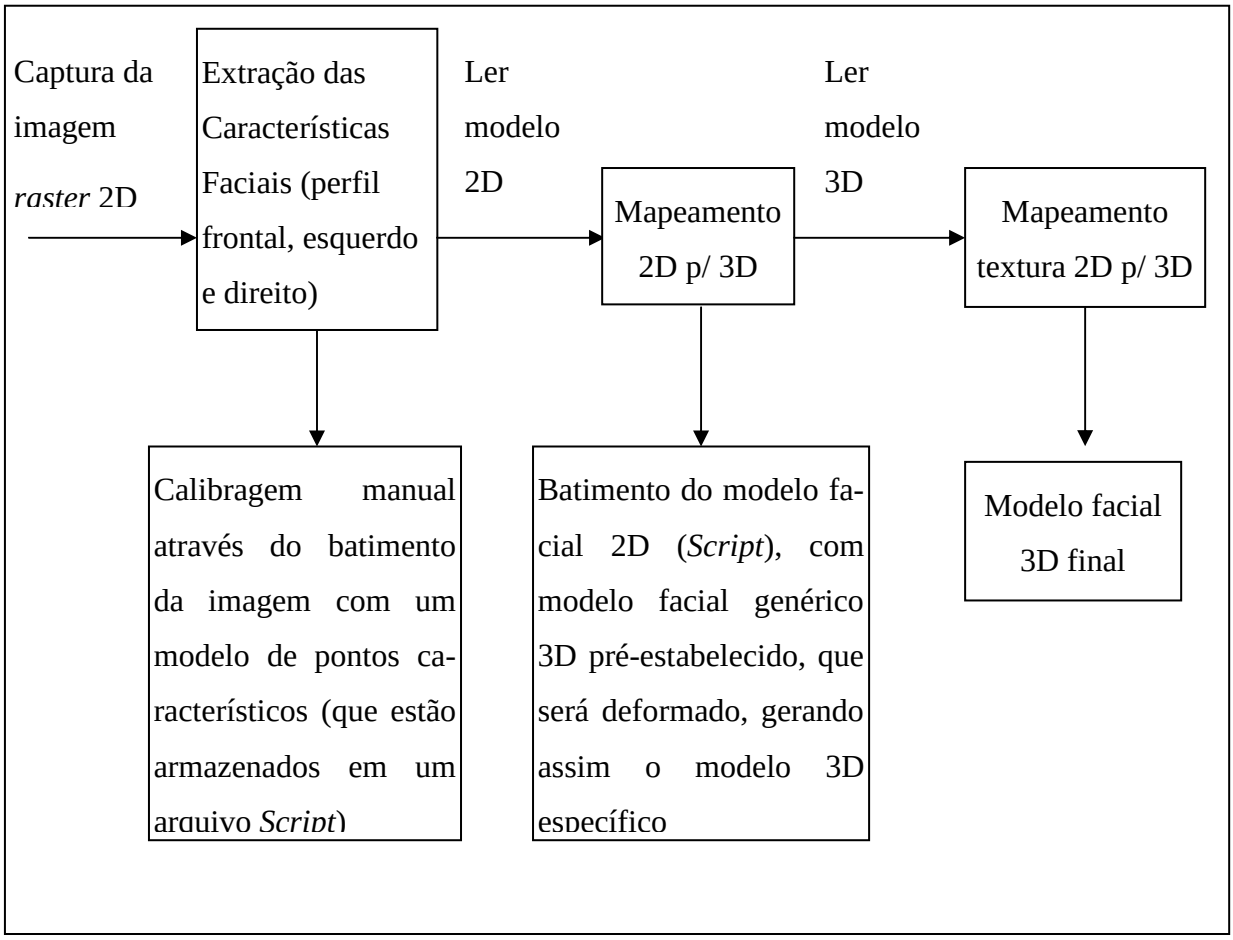
O protótipo de software desenvolvido aborda a modelagem de faces 3D tendo como embasamento os conceitos apresentados nas diversas referências consultadas descritas acima.

O protótipo desenvolvido segue quase todos os procedimentos apresentados em Huang (1996). Este, o proposto por Huang, apresenta um algoritmo que extrai automaticamente as características faciais (sobrancelhas, olhos, nariz, boca, etc.) de uma pessoa em três perfis (frontal, direito e esquerdo). A extração é feita a partir de imagens em 2D dos três perfis respectivos. Feito isso, armazena-se todos os pontos característicos, onde estes pontos irão ser mapeados para um modelo genérico 3D pré-estabelecido. O que acontece neste mapeamento é a deformação do modelo genérico, para que este fique com as características faciais extraídas das imagens 2D. Tendo como etapa final à passagem da textura facial dos modelos 2D para o modelo 3D que com isso, ganhará maior realismo.

A principal diferença do protótipo desenvolvido com a abordagem descrita por Huang (1996) é que na etapa de extração das características faciais, a extração é feita manualmente através de um ajuste dos pontos característicos genéricos nas imagens 2D.

O processo de modelagem facial 3D é feito em três etapas, sendo elas, (1) a etapa de extração das características manualmente, (2) mapeamento do modelo 2D para o modelo 3D e (3) a etapa de mapeamento da textura dos modelos 2D para o modelo 3D. Na Figura 30 pode ser visto um resumo do funcionamento do protótipo.

Figura 30 – Resumo do protótipo proposto.



Na etapa Extração das Características Faciais no momento da edição dos pontos não pode ser efetuada a geração de novos pontos e a remoção dos pontos lidos. Só poderá ser feita a calibragem dos pontos já existentes com a face que será modelada.

O dispositivo de entrada matricial utilizado no trabalho foi a Câmera de crominância que capta a imagem em cores (seção 2.4), e gera um sinal de vídeo composto colorido. A imagem, em geral, não é profissional. A Câmera de crominância é utilizada para aplicações em multimídia ou em casos onde não são necessárias imagens com muita qualidade, pois esta é uma câmera do tipo doméstica (VHS, 8mm, VHS-C etc). Foi utilizada no protótipo uma câmera da Creative modelo WebCam Plus.

5.1 ESPECIFICAÇÃO DO PROTÓTIPO

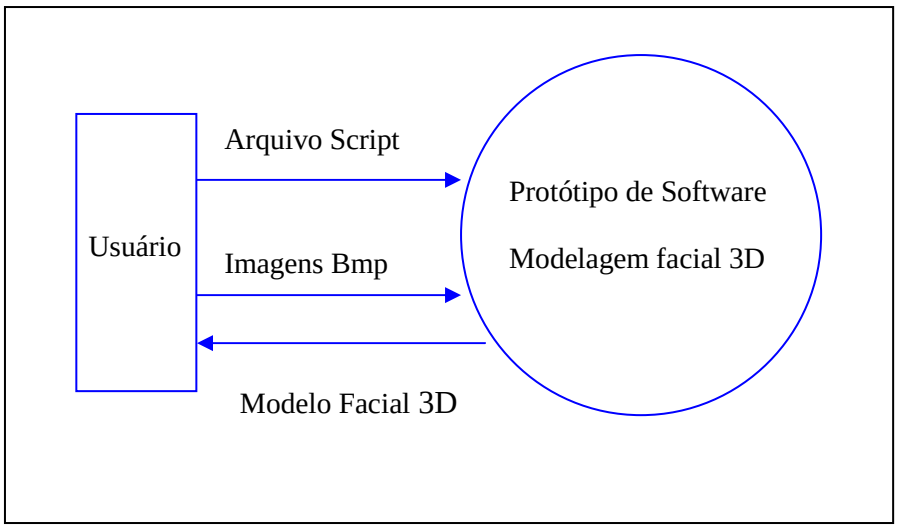
Segundo Melendez (1990), para o desenvolvimento de sistemas de informação, a prototipação representa uma boa solução para a maioria dos problemas. A metodologia de prototipação de sistemas utilizada neste protótipo é a Prototipação Fundamental ou Básica, originária do método Evolutivo, onde o produto final será o próprio sistema na sua forma mais aperfeiçoada. Conforme Melendez (1990), o método evolutivo é usado na identificação do problema e na construção de modelos concretos, adaptados e corrigidos à medida que o usuário e o analista vão conhecendo a realidade e a solução do problema.

Sendo que o resultado deste processo evolutivo de prototipação tem-se o Diagrama de Contexto, Diagrama de Fluxo de Dados, Modelo entidade relacionamento e o Dicionário de Dados, bem como, os arquivos de Script utilizados no protótipo descritos a seguir.

5.1.1 Diagrama de Contexto

O Diagrama de Contexto é uma representação gráfica do sistema como um todo e os seus relacionamentos. Na figura 31, tem-se como escopo deste protótipo as diferentes entradas e a visualização da animação pelo usuário.

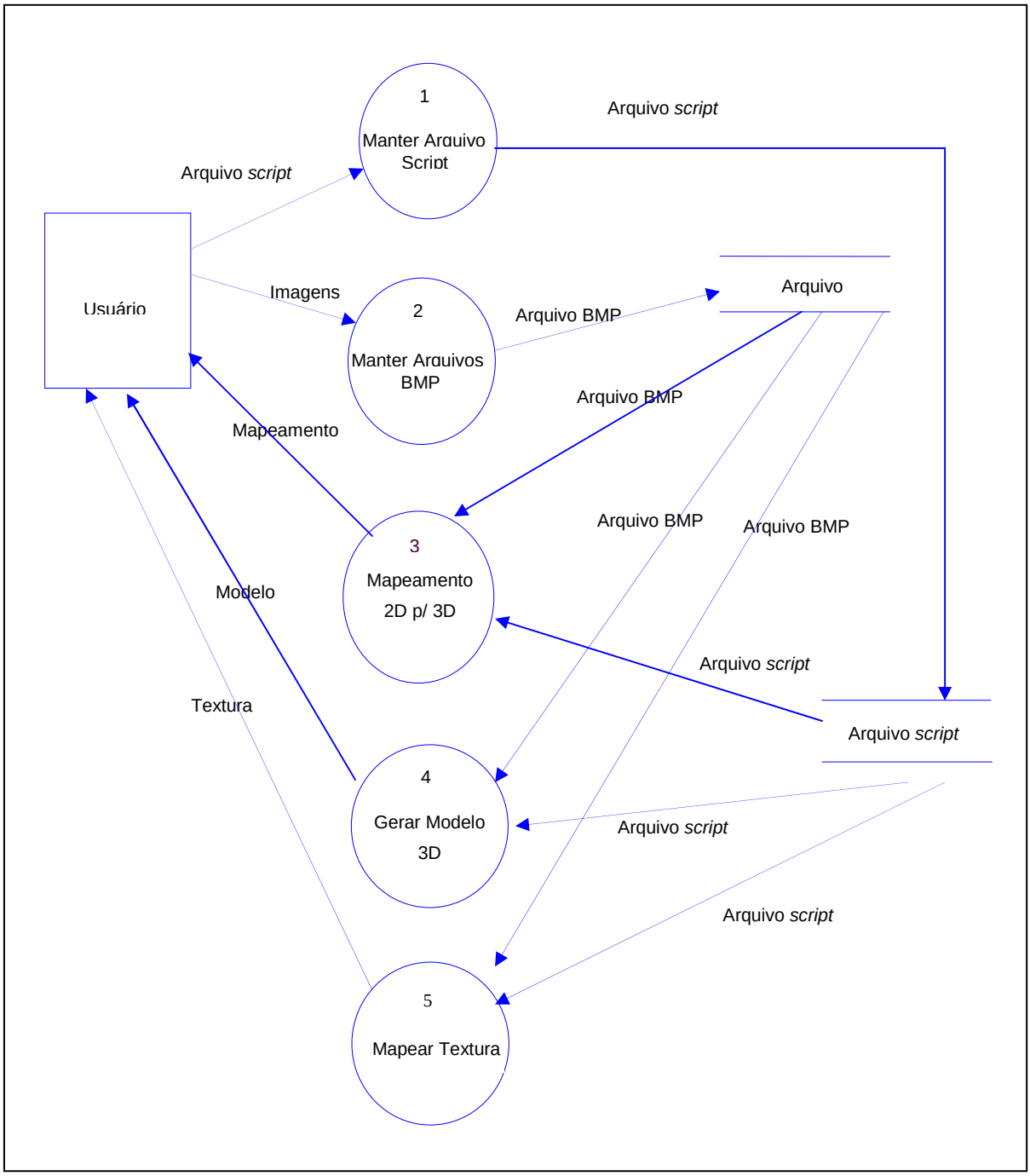
Figura 31 – Diagrama de contexto.



5.1.2 Diagrama de Fluxo de Dados

Na figura 32, encontra-se o Diagrama de Fluxo de Dados de nível 1, que descreve o fluxo de informações e as transformações que são aplicadas à medida que os dados se movimentam da entrada para a saída.

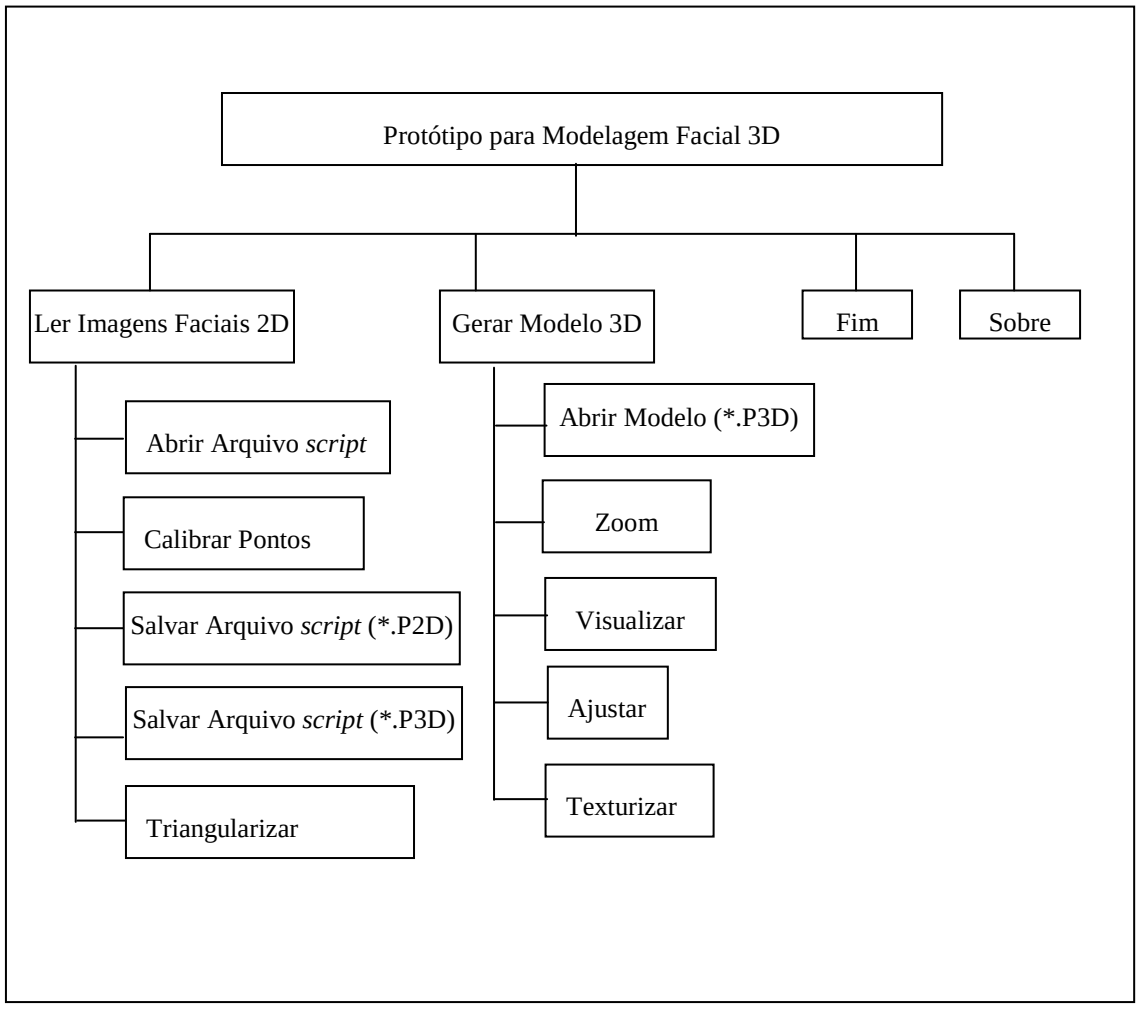
Figura 32 – Diagrama de fluxo de dados.



5.1.3 Diagrama Hierárquico Funcional (DHF)

Na Figura 33 encontra-se o Diagrama Hierárquico Funcional.

Figura 33 – Diagrama Hierárquico Funcional.



5.1.4 Dicionário de Dados

No Quadro1, encontra-se o Dicionário de Dados.

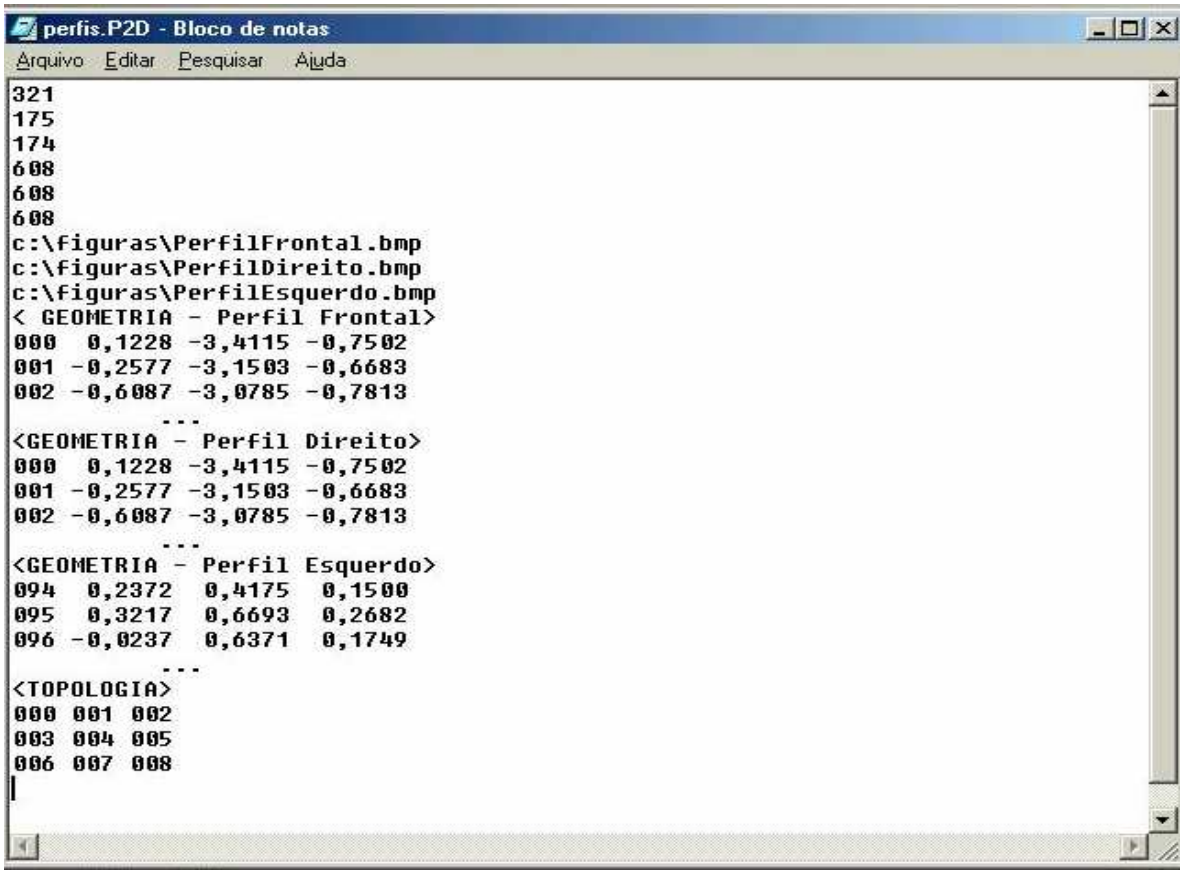
Quadro 1 - Dicionário de Dados

Nome	Tipo	Bytes	Descrição
QtdePontosFrontal QtdePontosDireito QtdePontosEsquerdo QtdeTriangFrontal QtdeTriangDireito QtdeTriangEsquerdo	Integer	4	Quantidade de pontos e de triângulos das imagens 2D; Fluxo de dados do arquivo *script* para o sistema.
PtoFrontal_x PtoFrontal_y PtoDireito_x PtoDireito_y PtoEsquerdo_x PtoEsquerdo_y	TipoVetor	400	Vetor onde contem o valores das coordenadas X e Y (pontos) de cada uma das imagens; Fluxo de dados do arquivo *script* para o sistema.
NumPtoFrontal NumPtoDireito NumPtoEsquerdo	TipoVetor	400	Vetor que contem o índice dos pontos de cada perfil. Fluxo de dados do arquivo *script* para o sistema.
TriFigFrontal TriFigDireito TriFigEsquerdo	Array	400	Array que armazena os triângulos das imagens; Fluxo de dados do arquivo *script* para o sistema.
TipoVetor	Array [1..100]	400	Armazena os índices e as coordenadas **x** e **y** das imagens.
NomeFigFrontal NomeFigDireito NomeFigEsquerdo	String	256	Nome dos arquivos das imagens; Fluxo de dados do arquivo *script* para o sistema.

5.1.5 Arquivos Script

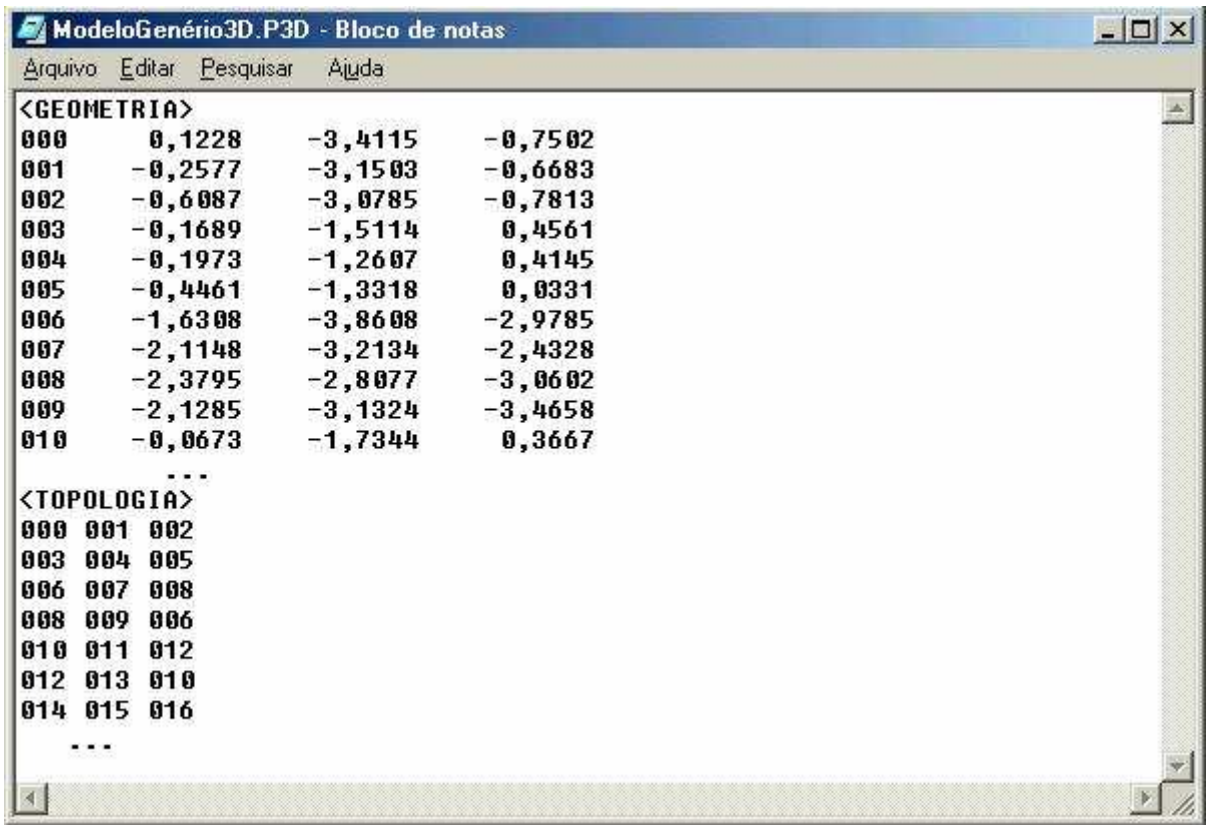
Os arquivos Script são de formato tipo texto, que são identificados com as extensões (*.P2D e *.P3D). O arquivo com extensão *.P2D é utilizado no módulo Ler Imagens Faciais 2D do protótipo (figura 33). Cada linha de comando no arquivo Script representa um dado relevante sobre as figuras que estão nele descritas, como mostra a figura 34. Este arquivo traz todos os dados sobre as imagens de cada perfil facial, tendo na sua primeira linha a quantidade de pontos da imagem do perfil frontal; na segunda linha a quantidade de pontos da imagem do perfil direito; na terceira linha a quantidade de pontos da imagem do perfil esquerdo, seguidas pelas linhas quatro, cinco e seis que contém quantidade de triângulos gerados em cada imagem. Depois vem a seção GEOMETRIA que contém o conjunto de pontos da figura com seu índice e suas coordenadas (x, y e z), com tamanho fixo de 3 caracteres para o índice e valor de cada eixo com o tamanho de no máximo 7 caracteres. A seção TOPOLOGIA contém o conjunto de ligações entre os pontos para que formem a malha triangularizada da figura, com tamanho fixo de 3 caracteres para cada índice de ponto.

Figura 34 - Arquivo Script (*.P2D).



Já o arquivo com extensão *.P3D é utilizado no módulo Gera Modelo 3D do protótipo, onde a seção GEOMETRIA contém o conjunto de pontos da figura com seu índice e suas coordenadas (x, y e z), com tamanho fixo de 3 caracteres para o índice e no máximo 7 caracteres para o valor de cada eixo. A seção TOPOLOGIA contém o conjunto de ligações entre os pontos para que formem a figura, com tamanho fixo de 3 caracteres para cada índice de ponto.

Figura 35 – Arquivo Script (*.P3D)



O arquivo *Script* pode ser alterado somente utilizando-se um editor de texto externo a aplicação do protótipo, acrescentando-se mais pontos, trabalhando diretamente com as coordenadas de cada figura, inclusive, mudando os seus valores.

5.2 IMPLEMENTAÇÃO DO PROTÓTIPO

Esta seção descreve a implementação do protótipo para modelagem facial 3D com base na técnica de Faces Limitantes (seção 3.6). Neste, estão descritos os formatos das estruturas de dados utilizadas, a interface e os algoritmos utilizados para geração dos modelos.

Este protótipo teve a sua implementação no ambiente *Delphi* 5.0, onde se utilizou a programação estruturada composta praticamente por *unit´s* distintas, sendo elas: `CapturaImagem`, que contém dados referentes ao menu principal e aos controles das janelas que estão ativas; A *unit* `Novo`, que contém todos os dados referentes às figuras 2D que estão sendo trabalhadas, os arquivos *Script* (texto) e todas as outras variáveis para o controle do software; a *unit* `GeraModelo3D`, a qual tem a função de gerar o modelo facial genérico 3D; a *unit* `VisualizaModelo3D` que tem por finalidade a visualização do modelo facial genérico 3D pelo usuário; a *unit* `Sobre` tem como função demonstrar todas as informações sobre o protótipo; e por fim a *unit* `Tipos` que descreve alguns tipos de dados utilizados pelo protótipo.

Além dos componentes da biblioteca de visualização, foram utilizados componentes padrões do ambiente Delphi, como `ScrollBar` para alterar valores de visualização da câmera (zoom), caixas `Edit` para visualizar os valores de foco da visualização, `OpenDialog` para possibilitar a abertura de arquivos e `SaveDialog` para o salvamento dos mesmos.

Utilizou-se a propriedade `Timage` para a inserção das três imagens 2D de perfis faciais utilizadas no módulo Ler Imagens Faciais 2D. E a propriedade `TCanvas`, para a coloração dos pontos em tonalidades de vermelho, e amarelo para os envelopes dos pontos para melhor visualização pelo usuário e a tonalidade de azul para quando for selecionado um ponto com o *mouse*.

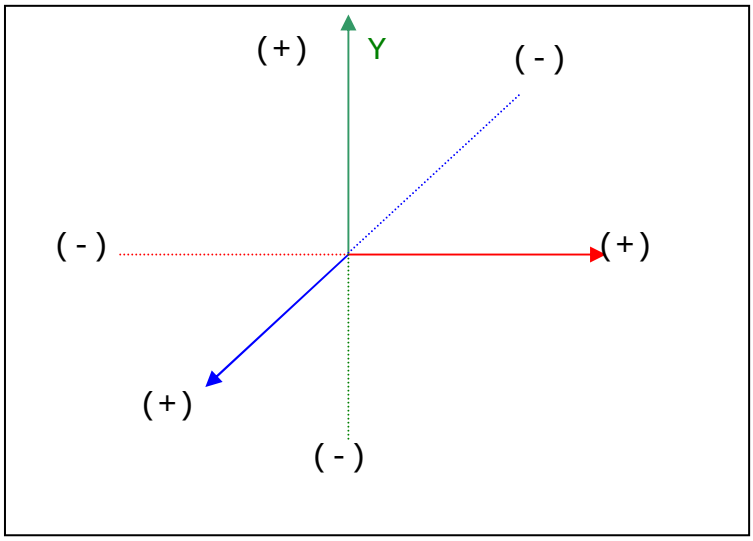
Utilizou-se ainda a propriedade `TMemo`, que é um arquivo temporário para efetuar a leitura e escrita do arquivo Script (*. P2D, *.P3D), e para armazenar os seus dados nas respectivas variáveis.

Durante a execução do protótipo, à medida que o usuário vai interagindo, fazendo todos os passos necessários para gerar o modelo facial, os arquivos utilizados podem ser atualizados.

Basicamente, o protótipo possui três fases principais, sendo elas: a fase de Calibragem das características faciais no modelo 2D, fase de mapeamento 2D para 3D, e por fim a fase de mapeamento da textura do modelo 2D para o modelo 3D, que serão descritas mais detalhadamente na seção Funcionamento do Protótipo (5.2.3).

O sistema de coordenadas utilizado é o da mão-direita, sistema que obedece a seguinte ordem: o eixo X cresce para a direita, a eixo Y cresce para cima e o eixo Z perpendicular ao papel e se aproximando do observador (Figura 36).

Figura 36 – Sistema de coordenadas (Mão Direita)



5.2.1 Utilizando Delphi e OpenGL

A implementação do protótipo utilizou-se da biblioteca OpenGL no ambiente Delphi, requerendo apenas que se cumpram os dois seguintes passos:

- No diretório \WINDOWS\SYSTEM devem estar gravados os arquivos GLU32.DLL e OPENGL32.DLL, que são as bibliotecas necessárias para a execução de aplicações que utiliza o OpenGL;
- No ambiente Delphi, deve-se instalar o pacote de componentes *SignSoft Visit Components 2.0* (Signsoft, 2001).

Após a execução do segundo passo acima citado, estarão disponíveis as pastas Visit e Visit Obj no ambiente Delphi (Figuras 37 e 38).

Figura 37– Pasta de componentes Visit.

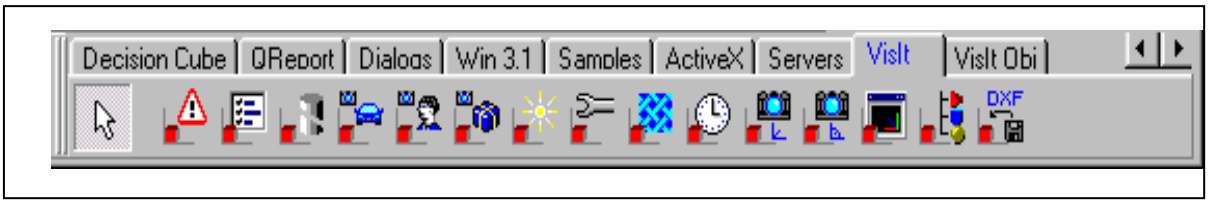
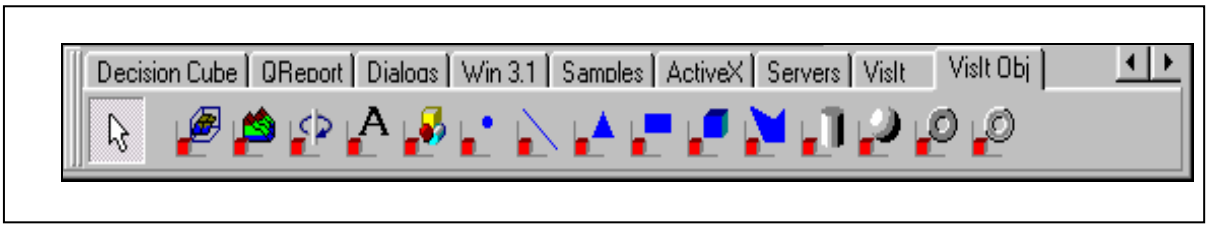


Figura 38– Pasta de componentes Visit Obj.



5.2.2 Visualização do OpenGL

Para visualização dos objetos, foram utilizados os componentes da biblioteca OpenGL, `VisView`, que é um componente básico necessário para renderizar a imagem desejada. Seu comportamento é similar a controles `Windows` comuns. Todas as ações irão ocorrer dentro deste componente; o `VisPerspectiveCamera`, que simula uma câmera, podendo-se definir a posição de visualização de uma imagem; `VisLight,` que serve para controle de luz; `VisAttributes, VisMaterial` que servem para definir atributos dos objetos, como a cor e por último o `VisTimer,` que serve para temporizar a atualização do que esta sendo exibido no `VisView`.

O componente `VisTimer` é disparado quando habilitado e executa suas linhas de código depois do tempo determinado na propriedade `Intervall`. Neste caso, é executado depois de 50 milisegundos, este tempo foi determinado para que o `VisView` atualize a tela e a atualização seja imperceptível pelo usuário. A procedure `Timer` possui a chamada de uma procedure para atualização da tela, como pode ser visto no quadro 2.

Quadro 2 - Procedure TimerTimer.

```
procedure TFrSweep.TimerTimer(sender: TObject; lagcount: Integer);
begin
 MeuView.InvalidateGL;
end;
```

A chamada à procedure para a atualização da tela, na qual devem estar todas as instruções para a geração de objetos, deverá estar no evento `OnRender` do componente `VisView`, sem o qual os objetos não serão exibidos. Nesta procedure também deverão estar as chamadas aos eventos `OnRender` dos componentes `TvisLight`, `TvisAttributes` e `TvisMaterial`.

Para facilitar a manipulação dos objetos, foi criado um componente zoom, de controle da propriedade da câmera. Além dos botões para ajustar a imagem e visualização (Figura 39).

Figura 39- Controles de Visualização.

A opção zoom (Figura 39), modifica o valor do ângulo de abertura da lente da câmera (ou foco), fazendo assim, com que o observador tenha a impressão que o objeto está se aproximando.

A opção ajusta (Figura 39) tem como função alterar os valores de translação da câmera, de modo que qualquer objeto que esteja sendo visualizado possa ser observado por inteiro, e também no caso do mesmo ter saído do campo de visão do usuário devido a alguma alteração de posicionamento da câmera. Para tanto, utilizou-se o procedimento mostrado no Quadro 3.

Quadro 3 - Cálculo de Ajuste de Visualização.

```
i ) Achar mínimos e máximos  (x, y e z);
ii) Calcular as dimensões;
  dimX := (Xmax – Xmin);
  dimY := (Ymax – Ymin);
  dimZ := (Zmax – Zmin);
iii) Encontrar dimensões;
iv) Xcamera :=  -1 * (Xmin + (dimX/2));
     Ycamera  :=  -1 * (Ymin + (dimY/2));
     Zcamera  :=  -2 * (Xmax + maiorDimensao);
```

A opção visualizar (Figura 39), faz a câmera dar um giro de 360° em torno do eixo definido pelo usuário, para que o observador possa ter uma noção melhor do objeto gerado, visualizando o mesmo, de vários ângulos. Para isso, calculou-se o raio inicial, e depois foi-se incrementando o ângulo e recalculando as novas posições de `x,y` e `z` a cada novo ângulo gerado.

5.2.3 Mapeamento da Textura

É o processo de desenvolver e designar atributos de material a um objeto, para proporcionar uma aparência realística. Antes de as texturas serem aplicadas, todos os objetos em um programa 3D possuem uma aparência padrão, seja cinza ou em algumas cores. O mapeamento dá ao objeto uma cor, acabamento, ou textura específicos.

Para se fazer o mapeamento da textura das imagens faciais 2D para o modelo 3D optou-se por fazer uma aproximação através de um processo recursivo de interpolação utilizando a equação paramétrica. Este processo funciona da seguinte maneira:

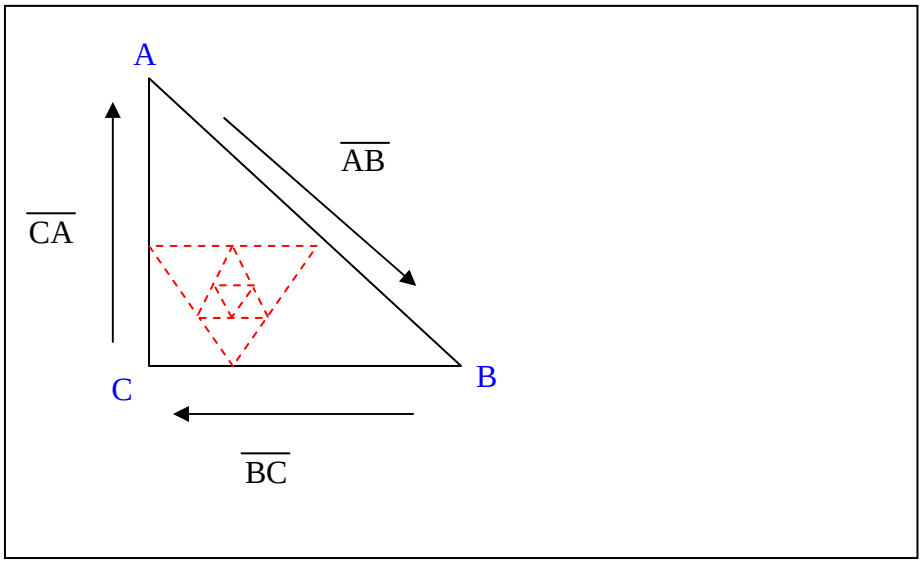
- Passo 1 : Primeiramente para cada triangulo do modelo 3D gerado são encontrados 10 (dez) pontos para cada segmento de reta que forma o triangulo, aplicando-se a equação paramétrica;

- Passo 2: Depois aplica-se a equação paramétrica para encontrar o *pixeis* equivalentes nas imagens 2D, capturando-os e armazenando sua cor em um vetor;

- Passo 3: Divide-se cada triângulo do modelo 3D em quatro novos triângulos encontrando através da equação paramétrica 3 (três) pontos que equivalem a metade de cada segmento de reta do triângulo, depois é aplicado o passo 1 novamente;

A Figura 40 mostra como funciona o algoritmo de texturização.

Figura 40 – Esquema de texturização.



Para se fazer o mapeamento da textura das imagens faciais 2D para o modelo 3D no protótipo utilizou-se o código apresentado no Quadro 4.

Quadro 4 – Código do Mapeamento de textura.

```
procedure TFmVisualiza3D.texturizar(Ax3,Ay3,Az3,Bx3,By3,Bz3,Cx3,Cy3,Cz3,Nivel:real);
 var
  Recursivo, i:integer;
  NovoAx3,NovoAy3,NovoAz3, NovoBx3,NovoBy3,NovoBz3, NovoCx3,NovoCy3,NovoCz3: real;
  x3,y3,z3,x2,y2:real;
incremento:real;
cor:tcolor;
 begin
    If Nivel<=(strtoint(FmVisualiza3D.EdRecursividade.text)) then
      begin
          for i:=1 to 10 do {segmento A-B}
              begin
                  incremento:=(i/10);
                  x3:= Ax3+((Bx3 - Ax3)*incremento);
                  y3:= Ay3+((By3 - Ay3)*incremento);
                  z3:= Az3+((Bz3 - Az3)*incremento);
                  x2:= x3;
                  y2:= y3;
                  x2:=((x2 *(strtofloat(FmPrincipal.EdEscala.text)))+(352/2));
                  y2:=((y2 *(-(strtofloat(FmPrincipal.EdEscala.text))))+(300/2));
                  cor:= FmImagens2D.ImageFrontal.canvas.pixels[trunc(x2),trunc(y2)];
                  FmVisualiza3D.MeuVisMat.ColorDiffuse.Color:= cor;
                  {glBegin(GL_POINTS);
                  glVertex3f(x3,y3,z3);
                  glEnd;}
                  inc(contador);
                  tipoX[contador]:=x3;
                  tipoy[contador]:=y3;
                  tipoz[contador]:=z3;
                  tipocor[contador]:=cor;
              end;
           for i:=1 to 10 do {segmento B-C}
                          . . .
           for i:=1 to 10 do {segmento C-A}
                          . . .
         {dividir triangulo em 4 pedaços} {novo ponto A}
         NovoAx3:= Ax3+((Bx3 - Ax3)*0.5);
         NovoAy3:= Ay3+((By3 - Ay3)*0.5);
         NovoAz3:= Az3+((Bz3 - Az3)*0.5);
         {novo ponto B}
         NovoBx3:= Bx3+((Cx3 - Bx3)*0.5);
         NovoBy3:= By3+((Cy3 - By3)*0.5);
         NovoBz3:= Bz3+((Cz3 - Bz3)*0.5);
         {novo ponto C}
         NovoCx3:= Cx3+((Ax3 - Cx3)*0.5);
         NovoCy3:= Cy3+((Ay3 - Cy3)*0.5);
         NovoCz3:= Cz3+((Az3 - Cz3)*0.5);
         {recursividade}
      texturizar(Ax3,Ay3,Az3,NovoAx3,NovoAy3,NovoAz3,NovoCx3,NovoCy3,NovoCz3,Nivel+1);
      texturizar(NovoAx3,NovoAy3,NovoAz3,Cx3,Cy3,Cz3,NovoBx3,NovoBy3,NovoBz3,Nivel+1);
      texturizar(NovoBx3,NovoBy3,NovoBz3,Bx3,By3,Bz3,NovoCx3,NovoCy3,NovoCz3,Nivel+1);
      texturizar(NovoAx3,NovoAy3,NovoAz3,NovoBx3,NovoBy3,NovoBz3,NovoCx3,NovoCy3,
                 NovoCz3,Nivel+1);
end;
```

```
end;
```

5.3 FUNCIONAMENTO

Esta seção explica todas as funções permitidas pelo protótipo para a modelagem facial 3D.

Ao executar o protótipo, é mostrada uma janela do tipo *About* do fabricante da biblioteca gráfica Signsoft (Figura 41), isto se deve ao fato de ser uma versão demonstrativa. Esta janela será fechada clicando-se no botão “OK” encontrado na mesma.

Figura 41- Tela *About* do fabricante da biblioteca.

Após o fechamento da tela anterior, encontra-se o *layout* geral contendo o menu principal, com os módulos do protótipo (Figura 42). Tem-se também nesta tela a vizualização do `memo`, o qual registra os *Scripts* da figura, no lado direito está o painel com os botões dos módulos do protótipo, e logo abaixo um campo onde o usuário deverá fornecer o fator de escala dos pontos 2D.

No módulo Ler Imagens Faciais 2D, permite-se abrir um arquivo de extensão “*.P2D”. Ao executar esta opção, uma caixa de diálogo é apresentada para a localização do arquivo *Script* (Figura 43). O arquivo *Script* selecionado é lido e armazenado no componente `memo` (Figura 42). Os dados contidos nos arquivos “*.P2D” é o conjunto de pontos genéricos e também a seqüência de índices que farão estes pontos serem triangularizados.

Figura 42 – Tela Principal do Protótipo

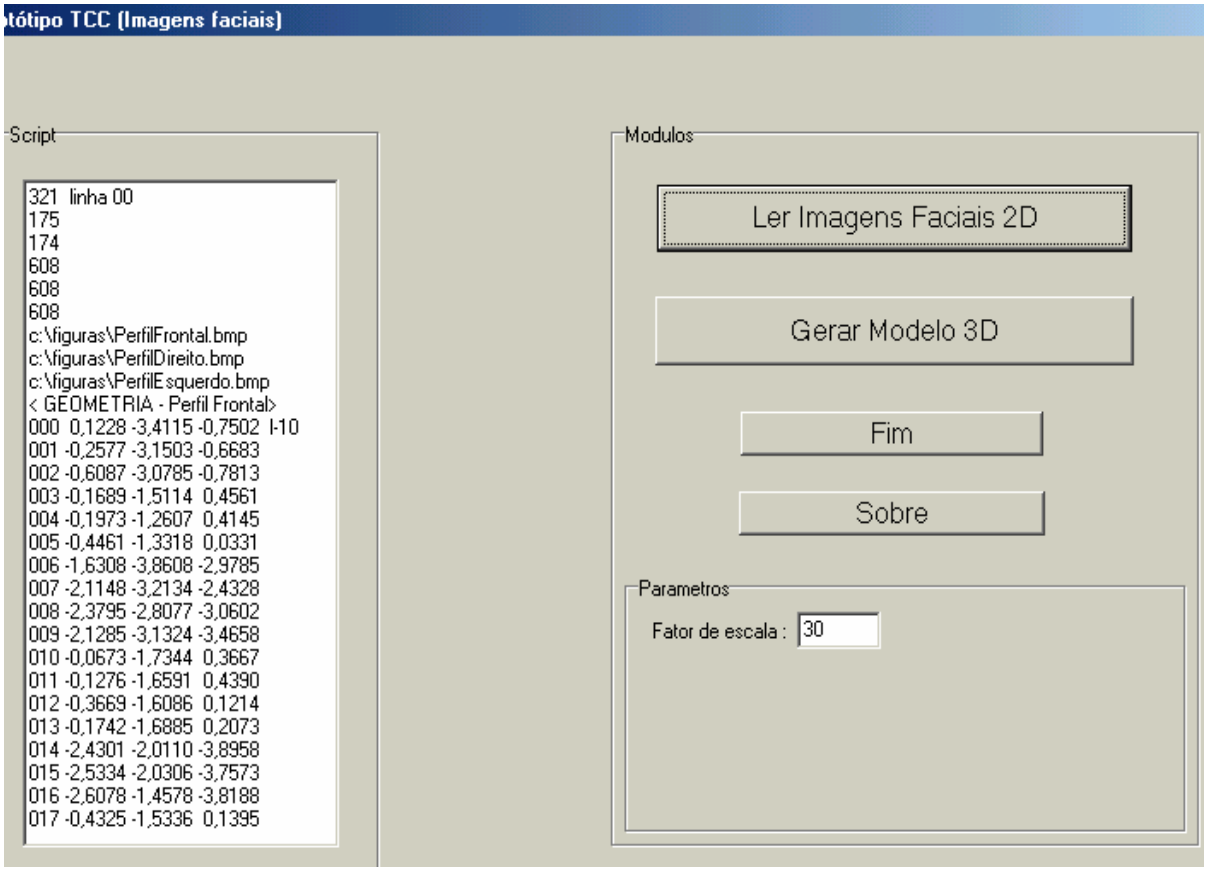
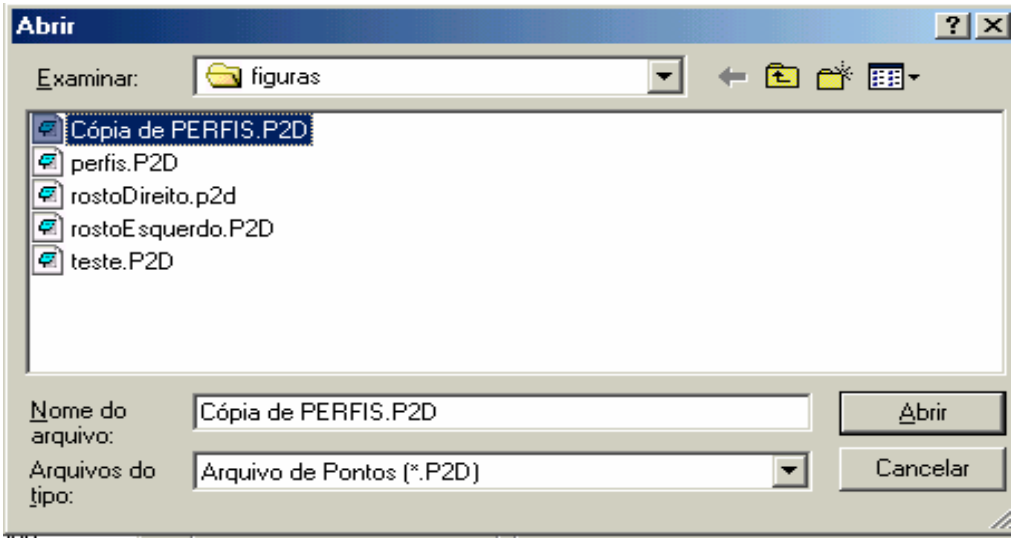


Figura 43- Caixa de diálogo (abrir arquivos *.P2D)



Depois de lido o arquivo Script o programa abre outra tela (Figura 44). Nesta tela aparece as três imagens dos perfis faciais (frontal, direito e esquerdo) que também foram lidos seus endereços (*path*) no arquivo *Script*, e sobre elas o conjunto de pontos correspondentes a cada perfil. A partir deste momento o usuário pode com o *mouse* mover os pontos, com isso o usuário pode calibrar os pontos de cada perfil de acordo com a face da pessoa na imagem (Figura 45).

Enquanto o usuário desliza o *mouse* sobre as imagens, vai aparecendo no visor logo abaixo da imagem do perfil direito, a posição do ponto (x ,y) e a imagem que o *mouse* está naquele momento (Figura 44).

Figura 44 – Tela de Calibragem do modelo 2D

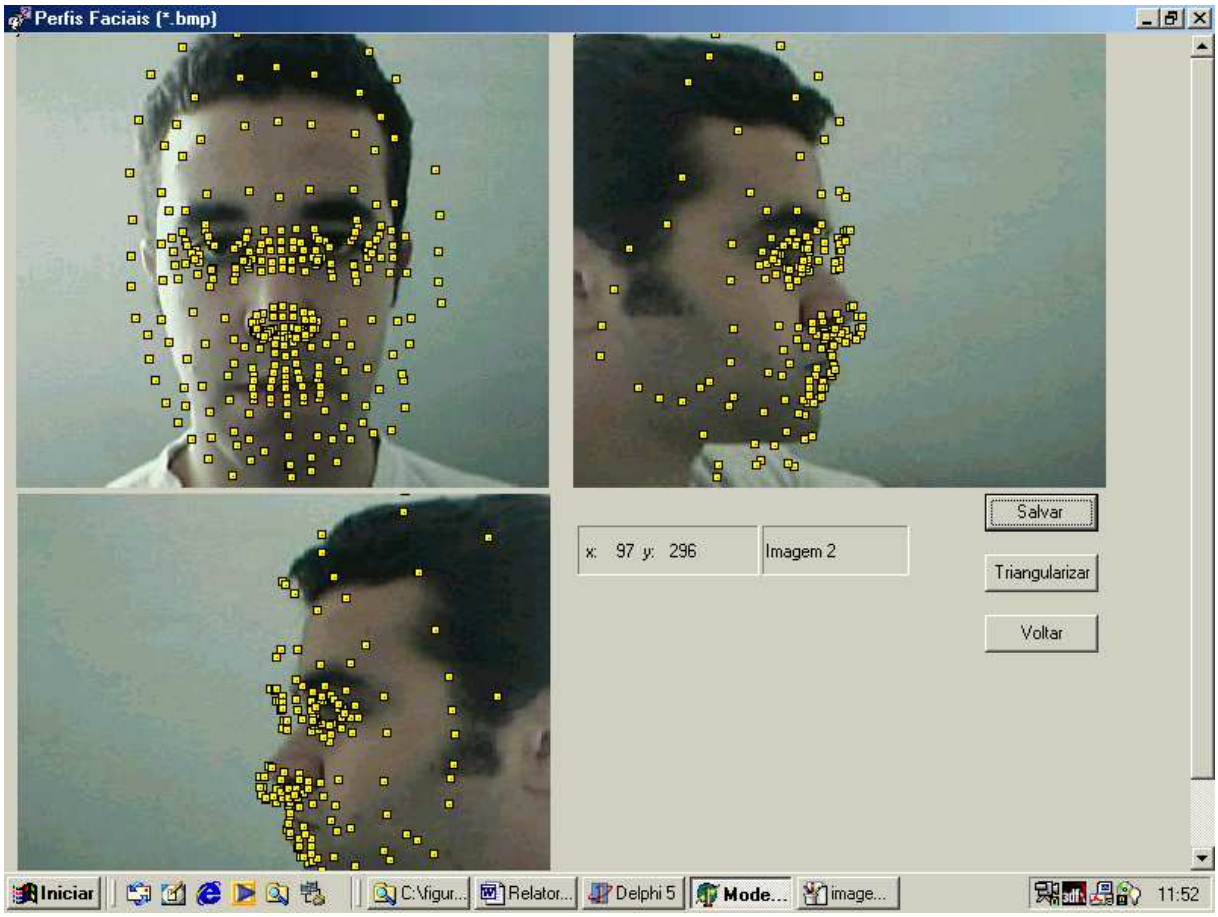
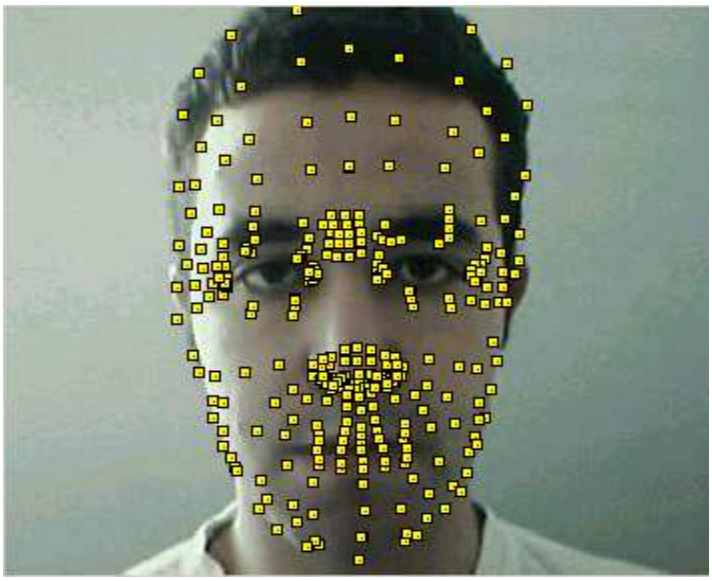
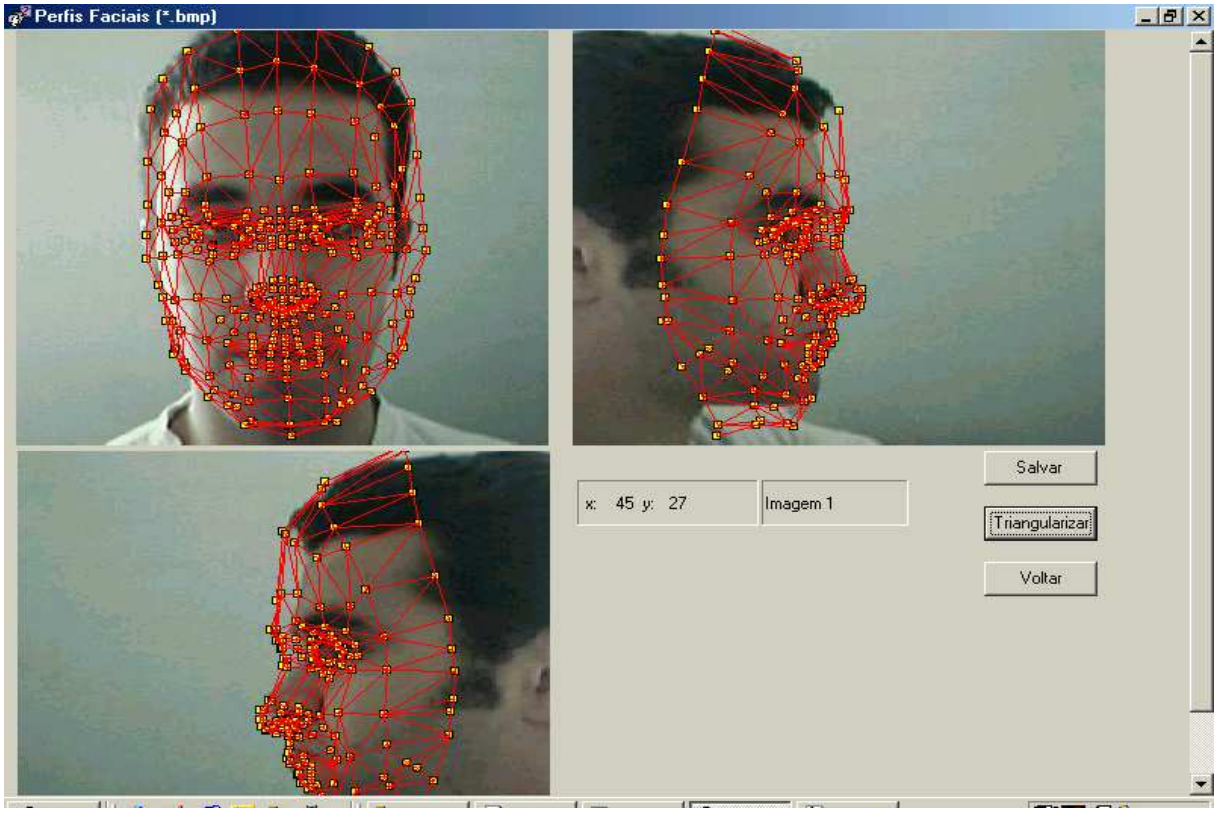


Figura 45 – Imagem facial frontal, com os pontos já calibrados.



Tem-se também o botão Triangularizar, que vai fazer a triangularização o conjunto de pontos (Figura 46).

Figura 46 – Malha de Pontos triangularizadas.



Pode-se nesta tela também salvar o conjunto de pontos calibrados, sendo estes utilizados para deformar linearmente os pontos no Modelo 3D. Para isto, basta acionar o botão Salvar, que então aparecerá uma caixa de diálogo para que o usuário salve um arquivo do tipo "*.P2D", onde estarão os dados da malha calibrada (Figura 47). Logo após o usuário salvar o arquivo *script* "*.P2D", aparecerá uma nova caixa de diálogo dando a opção para que o usuário salve um arquivo "*.P3D" (Figura 48), onde este arquivo poderá ser aberto no módulo `Geração do Modelo 3D`, e conseqüentemente poderá ser visualizado o modelo em 3D. Após fazer a calibragem dos pontos e salvar os arquivos o usuário deve acionar o botão `Voltar` para retornar a tela principal do protótipo.

Figura 47 – Caixa de dialogo para salvar arquivo *script* (*.P2D).

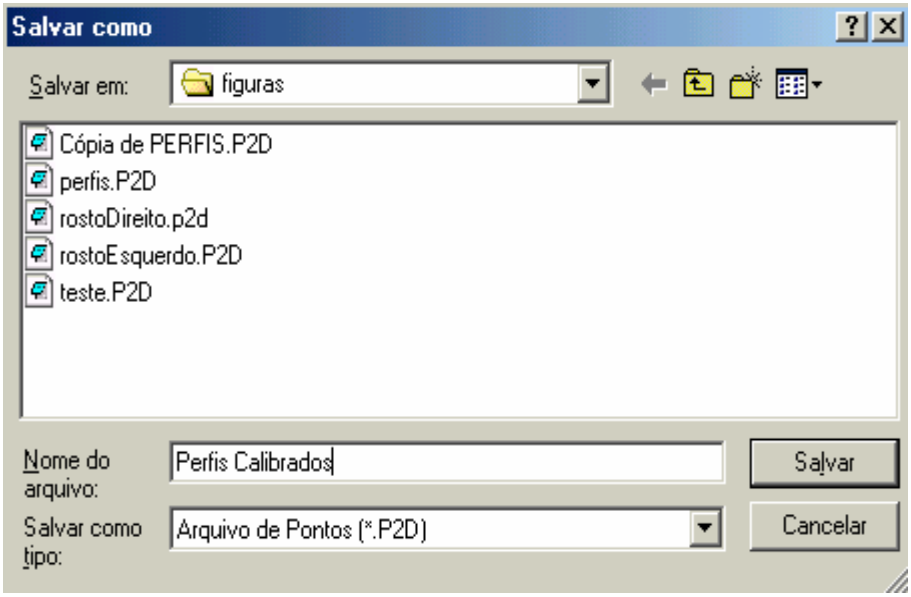
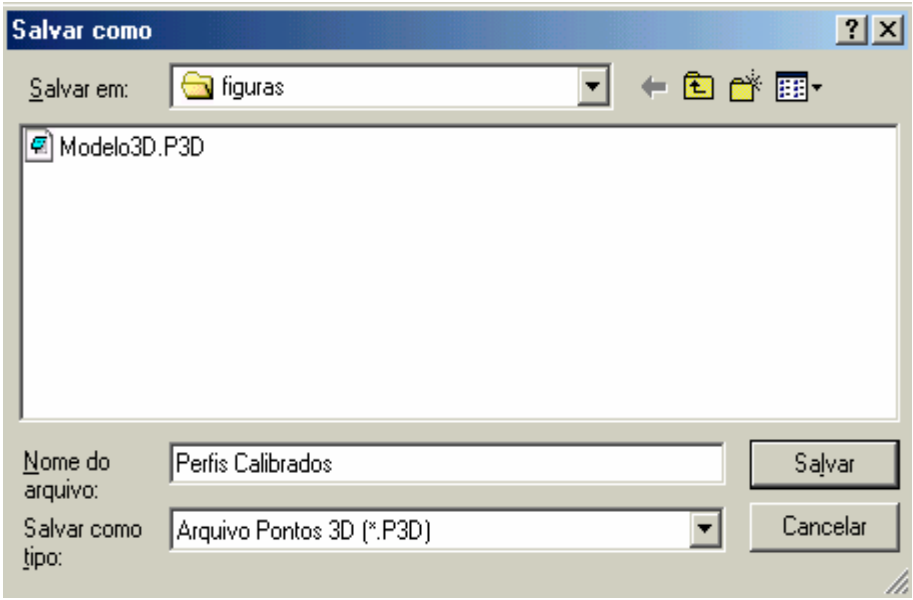
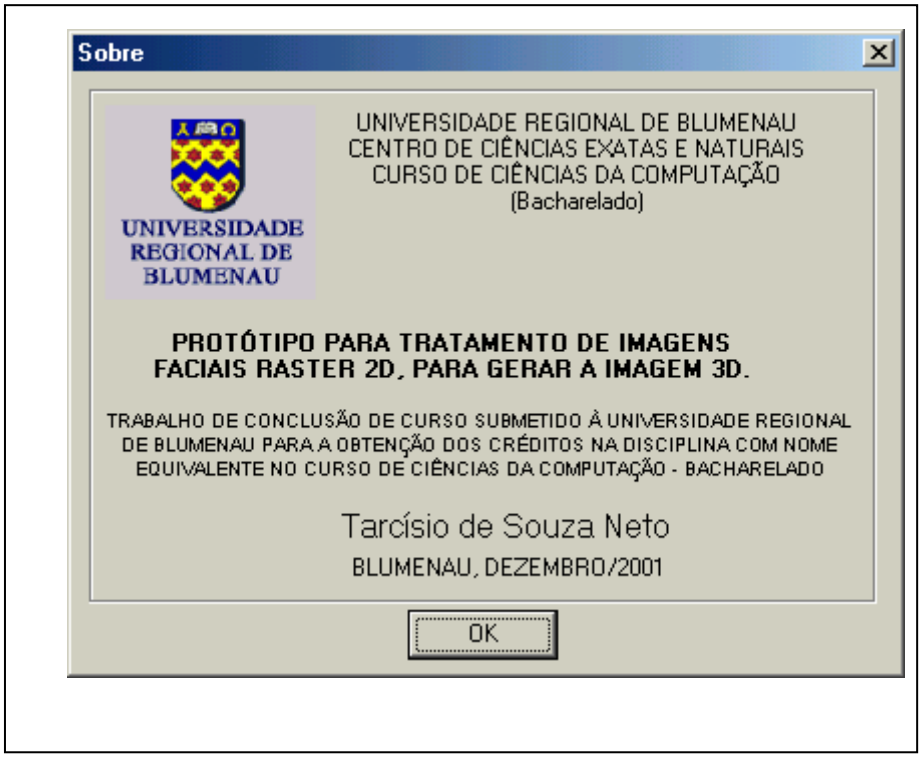


Figura 48 - Caixa de dialogo para salvar arquivo *script* (*.P3D).



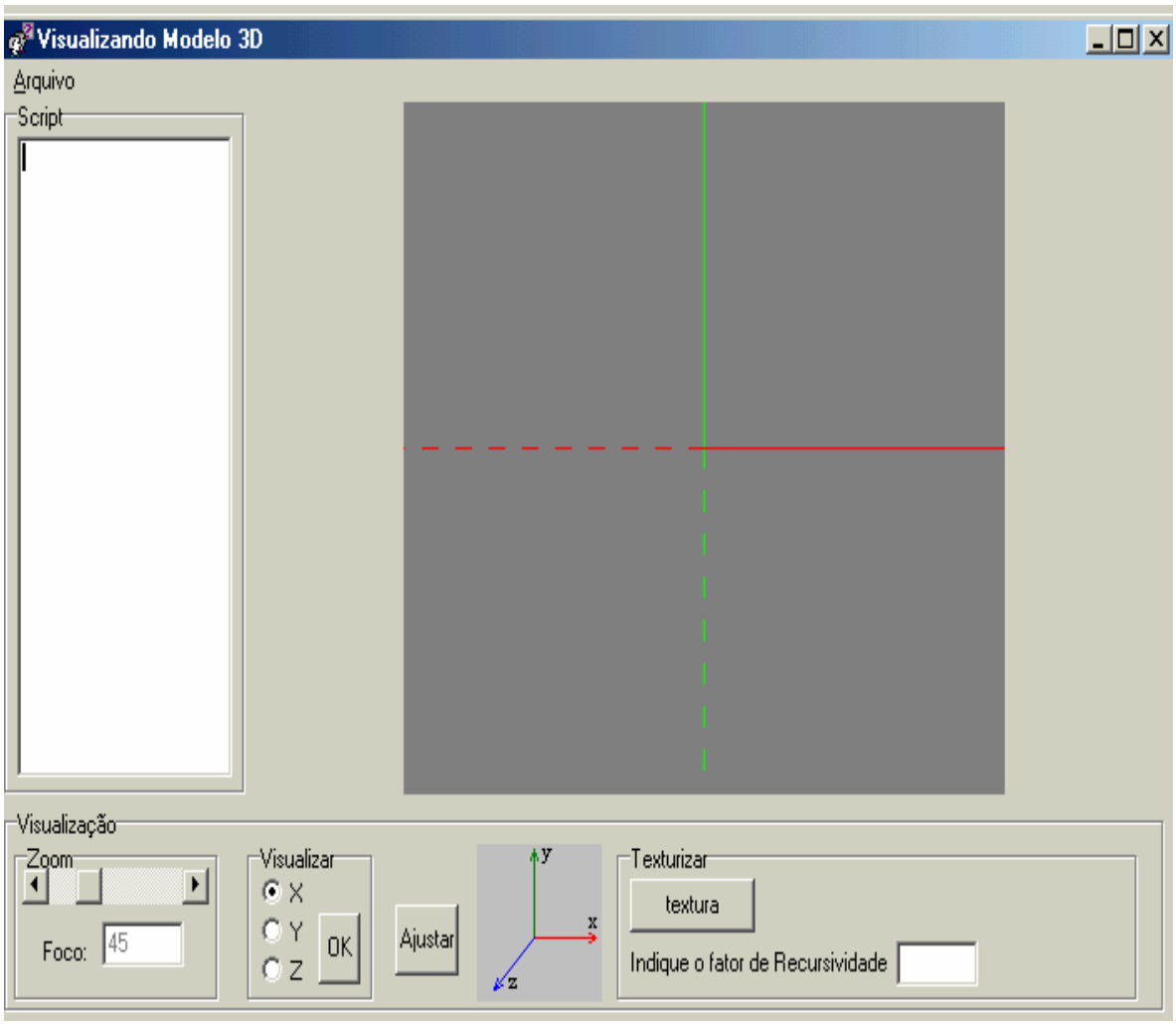
De volta a tela principal do protótipo o usuário pode acionar o módulo Sobre que mostra uma janela com informações gerais sobre o desenvolvimento do protótipo (Figura 49). Tem-se também o módulo Fim onde o usuário pode encerrar o programa.

Figura 49 – Módulo Sobre.



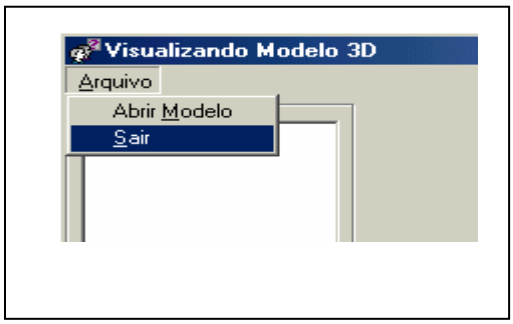
Tem-se por fim na tela principal do protótipo o módulo Gera Modelo 3D, que quando acionado abre novamente a tela tipo *About* do fabricante da biblioteca gráfica Signsoft (Figura 41). Logo após o usuário acionar o botão OK, abrirá uma nova tela, que é a tela de visualização dos modelos 3D (Figura 50), tanto o modelo genérico quanto os modelos gerados no módulo Ler Imagens Faciais 2D.

Figura 50 –Tela de Visualização do Modelo 3D.



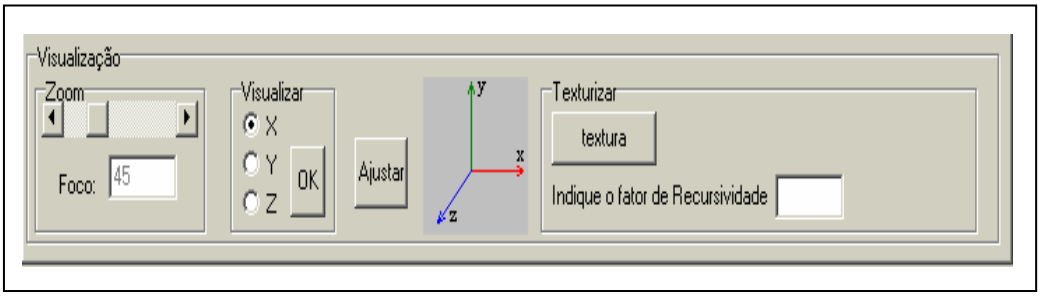
Na tela de visualização dos modelos 3D, tem-se o componente `memo` onde serão armazenados os dados lidos dos arquivos *script*, ao lado o visor onde será apresentado o modelo 3D, e logo abaixo as funções de visualização do modelo 3D. Na Figura 51 pode ser visto o menu Arquivo com suas opções de Abrir um Modelo para visualização e Sair da tela de visualização.

Figura 51 – Menu Arquivo.



A Figura 52 mostra os controles de visualização, que permitem alterar a propriedade de zoom da câmera, mostra as opções Visualizar**,** o botão Ajustar, mostra também a representação do sistema de referência (mão-direita) que está sendo usado e por fim a opção de Texturizar o modelo 3D. O item Visualizar, faz a câmera dar um *loop* de 360° em torno dos eixos x, y ou z, dando ao usuário uma melhor perspectiva do objeto como um todo.

Figura 52 - Controles de visualização.



Ao abrir o modelo através do menu Arquivo/Abrir Modelo, será aberta uma caixa de diálogo para que o usuário escolha o arquivo do tipo “*.P3D”(Figura 53). Pode-se abrir o arquivo do modelo genérico 3D (Figura 54) do qual foram retirados os pontos para construção do conjunto de pontos para as imagens 2D, como também pode-se abrir qualquer outro modelo deformado como mostra a Figura 55.

Figura 53 – Caixa de dialogo para abrir modelos 3D.

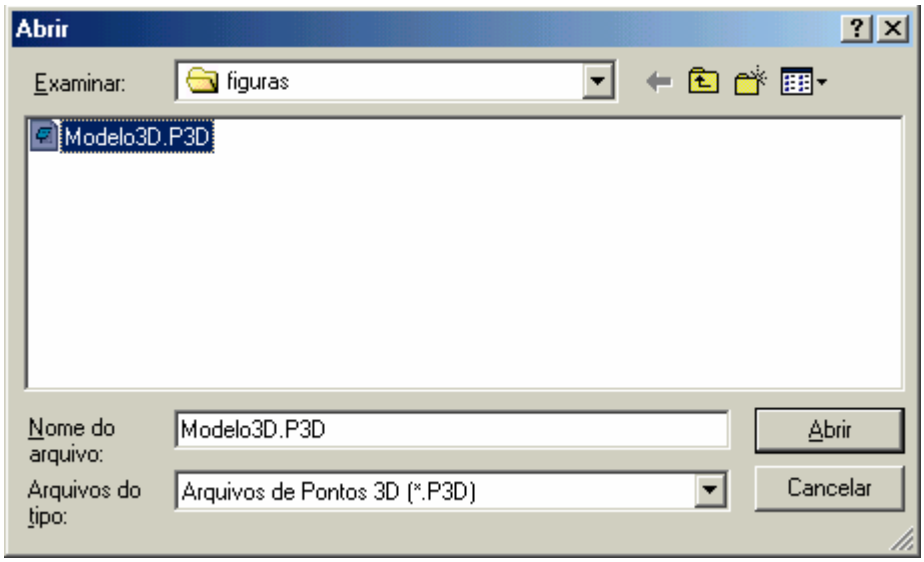
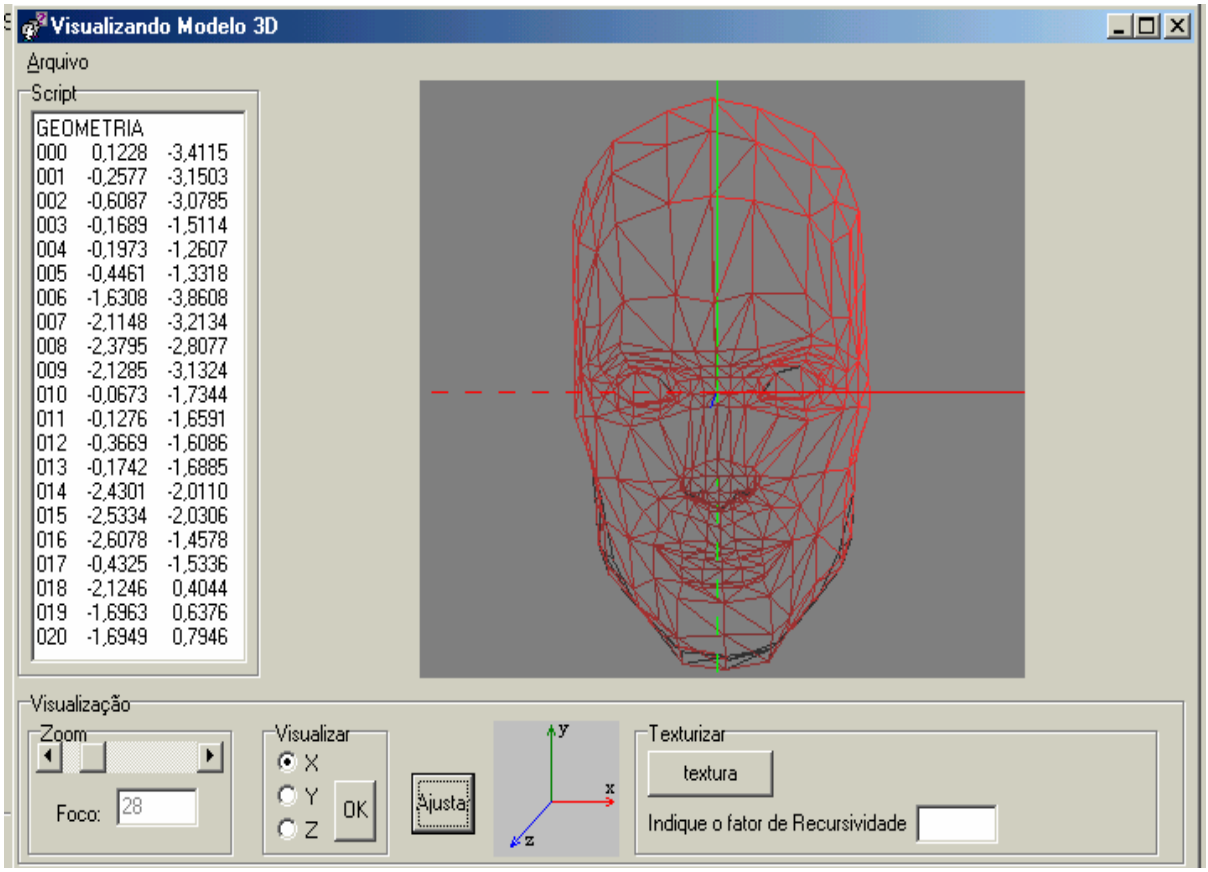
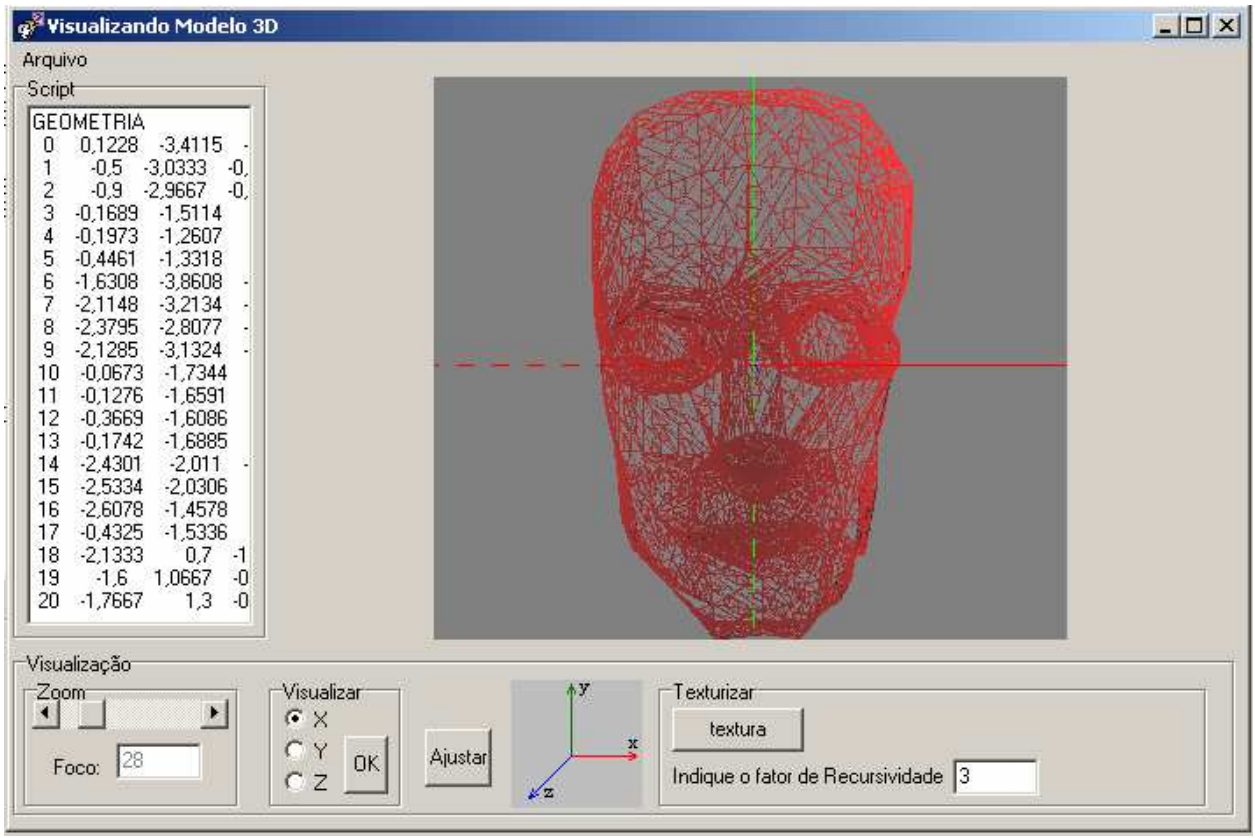


Figura 54 –Modelo facial genérico 3D.



Tem-se também a opção texturizar que antes de ser acionada o usuário terá que preencher o campo Indique o fator de Recursividade. Depois de acionado o protótipo utiliza o algoritmo descrito na seção 5.2.3 que utiliza equações paramétricas transferir a textura das imagens 2D para o modelo 3D (Figura 54).

Figura 55 –Modelo facial 3D deformado e texturizado.



Depois de vário testes, observou-se que no processo de texturização, o componente de textura da biblioteca gráfica Singsoft Visit (Singsoft, 2001) não conseguiu texturizar os pontos no modelo 3D com as cores extraídas das imagens 2D. O componente só pintava os pontos de vermelho ou branco. Tentou-se obter informações sobre a biblioteca no *site* do fabricante, mas não foi encontrada nada. Acha-se que este problema ocorreu por se tratar de uma versão demonstrativa da biblioteca.

6 RESULTADOS FINAIS

Nesta seção serão relatadas as conclusões obtidas com este trabalho, bem como as sugestões para trabalhos futuros.

6.1 CONCLUSÕES

Com a pesquisa e o estudo das referências bibliográficas, tornou-se possível a implementação do protótipo para tratar imagens faciais *Raster* 2D e gerar a partir destas uma aproximação da imagem facial 3D de uma certa pessoa, com base na técnica de modelagem 3D chamada de Faces Limitantes.

Conseguiu-se através da calibragem manual dos pontos extrair as características faciais da imagem e salvar estas características através de um arquivo *script*, onde o mesmo poderá ser lido no módulo `Gera modelo 3D`. Os dados lidos formam o modelo facial 3D da pessoa que esta nas três imagens faciais 2D (perfil frontal, direito e esquerdo).

O ambiente de desenvolvimento Delphi mostrou-se eficiente no que diz respeito a oferecer recursos para a manipulação das variáveis e geração dos arquivos. A biblioteca gráfica OpenGL também mostrou-se eficiente, tendo uma vasta quantidade de recursos, sendo utilizados no projeto apenas alguns deles.

Na fase de Extração das Características Faciais, sentiu-se uma grande dificuldade para calibragem manual dos pontos nas imagens 2D, pois o modelo facial genérico utilizado tinha muitos pontos, conclui-se então que a calibragem manual seria uma melhor solução, caso fosse utilizado um modelo genérico com menos pontos. Mas em conseqüência, a diminuição dos pontos pode propiciar um menor grau de realismo na fase de deformação do modelo 3D e de mapeamento da textura.

Já na fase de mapeamento 2D para 3D, utilizou-se a alteração das coordenadas `x`, `y` e `z` dos arquivos *scripts* para fazer a deformação do modelo genérico 3D, e assim podendo-se visualizar o modelo facial 3D de uma pessoa específica. No processo de texturização optou-se por fazer uma aproximação através de um processo recursivo de interpolação utilizando a equação paramétrica, tendo em vista que a versão da biblioteca gráfica utilizada era

demonstrativa, já que esta versão não disponibilizava recursos de mapeamento de textura exigidos.

6.2 EXTENSÕES

Como extensões para trabalhos futuros, pode-se citar:

- Implementação de um algoritmo para extração automática das características faciais;
- Implementar um protótipo que possa gerar e visualizar não só do rosto de uma pessoa mais sim toda a cabeça, ou até mesmo do corpo inteiro;
- Aprimoramento do algoritmo de mapeamento de textura, para ter-se um maior realismo no modelo 3d resultante, talvez utilizar métodos de texturização disponíveis na própria biblioteca gráfica (OpenGL), como o método *Gouraud* proposto por Huang.

REFERÊNCIAS BIBLIOGRÁFICAS

BANON, Gerald Jean Francis. **Bases da computação gráfica**. Rio de Janeiro: Campus,1989.

BROWN, C. Wayne; SHEPHERD, Barry J. ***Graphics file formats – reference and guide***. Greenwich: Manning Publications Company, 1995.

BERG, Alexandre Cruz. **Tutorial de Computação Gráfica,** Canoas, jan. [1999?]. Disponível em : <http://www.ulbra.tche.br/~berg/>. Acesso em : 16 out. 2001.

CASACURTA, Alexandre. **Introdução à computação gráfica,** São Leopoldo, dez. [1998?]. Disponível em: http://www.inf.unisinos.br/~osorio/CG-Doc/CG-Web/cg.html. Acesso em: 12 set. 2001.

CELANI, Maria G. **Multimídia, hipermídia e arquitetura;** algumas aplicações em pesquisa e desenvolvimento. 1997. 80 f. Dissertação (Mestrado em Arquitetura e Urbanismo)- Departamento de Arquitetura e Urbanismo, Universidade de São Paulo, São Paulo.

CORRIGAN, John. **Computação gráfica –** segredos e soluções. Rio de Janeiro: Ciência Moderna, 1994.

FOLEY, James D. **Computer graphics:** principles and pratice**.** New York: Addison-Wesley, 1990.

FRANÇA NETO, Leopoldo Rodrigues. **Um Ambiente para Processamento de Grandes Acervos de Imagens**, 1998. 137 f. Dissertação (Mestrado em Informática) – Departamento de Informática, Universidade Federal de Pernambuco, Recife.

HO, Shinn-Ying.; HUANG, Hui-Ling. Facial modeling from an uncalibrated face image using a coarse-to-fine genetic algorithm. **The journal of the pattern recognition society**. Taiwan, v.34, p. 1015-1031, fev. 2000.

HUANG, Thomas S.; TANG, L. Automatic construction of 3D human face models based on 2D images. **IEEE International Conference on Image Processing**. Switzerland, v.3, p. 467-470, set. 1996.

LINDLEY, Craig A. ***Practical image processing in C***. New York: John Wiley & Sons, 1991.

MELENDEZ, Rubem Filho. **Prototipação de Sistemas de Informações.** Rio de Janeiro : LTC – Livros Técnicos e Científicos Ed, 1990.

NEWMAN, William M.; SPROULL, Robert. **Principles of Interactive Computer Graphics**. New York: McGraw-Hill, 1973.

PERSIANO, Ronaldo César Marinho; OLIVEIRA, Antônio Alberto Fernandes de. **Introdução à computação gráfica**. Rio de Janeiro: Livros Técnicos e científicos, 1989.

SIGNSOFT. ***Signsoft GmbH - Software Development, Consulting, 3D Graphics***, Alemanha, mar. 2001. Disponível em: < http://www.signsoft.com/visit>. Acesso em: 15 nov. 2001.

VELHO, Luiz. **Computação Gráfica : Imagem**. Sociedade Brasileira de Matemática. Rio de Janeiro, 1994.

ZOZ, Jeverson. **Estudo de métodos e algorítmos de splines, catmull, casteujal e bezier**. 1999. Trabalho de conclusão de curso (Bacharel Em Ciências da Computação) - Fundação Universidade Regional de Blumenau, (Orientador) Dalton Solano dos Reis.