UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

ANIMAÇÃO DE UMA MARIONETE VIRTUAL, A PARTIR DE CAPTURA ÓPTICA DE MOVIMENTO HUMANO, SEM UTILIZAÇÃO DE MARCAÇÕES ESPECIAIS

SÍLVIA HÊDLA CORREIA DE SALES

BLUMENAU
2004

2004/2-44

SILVIA HÊDLA CORREIA DE SALES

ANIMAÇÃO DE UMA MARIONETE VIRTUAL, A PARTIR DE CAPTURA ÓPTICA DE MOVIMENTO HUMANO, SEM UTILIZAÇÃO DE MARCAÇÕES ESPECIAIS

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Paulo César Rodacki Gomes - Orientador

BLUMENAU
2004

2004/2-44

ANIMAÇÃO DE UMA MARIONETE VIRTUAL, A PARTIR DE CAPTURA ÓPTICA DE MOVIMENTO HUMANO, SEM UTILIZAÇÃO DE MARCAÇÕES ESPECIAIS

Por

SILVIA HÊDLA CORREIA DE SALES

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Paulo César Rodacki Gomes, Dr. – Orientador, FURB
Membro:	Prof. Nome do professor, FURB
Membro:	Prof. Nome do professor, FURB

Blumenau, 11 de novembro de 2004

AGRADECIMENTOS

Agradeço a todos que contribuíram direto ou indiretamente no desenvolvimento do trabalho, em especial aos meus pais Rafael Pedro de Sales e Helena Correia de Sales que apesar da distância sempre se fizeram presente.

A todos os meus amigos, em especial ao Júlio Vilmar Gesser pelo companheirismo. Aos dedicados professores do curso de Ciências da Computação em especial ao José Roque Voltolini da Silva, Jomi Fred Hübner e ao Evandro Felin Londero.

Ao meu orientador, Paulo César Rodacki Gomes, por ter sido não só um orientador, mas um amigo durante todo o decorrer do curso.

RESUMO

Este trabalho aborda conceitos da localização e animação das partes do corpo humano em personagens virtuais a partir da captura óptica do movimento humano sem a utilização de marcações especiais sobre o corpo do ator. Descreve também a implementação de um protótipo de *software* que utiliza geometria computacional para localização das partes que compõem o corpo humano em um arquivo de vídeo, que contém um ator em movimento e em seguida a sua animação. O trabalho tem o objetivo de analisar o arquivo de vídeo e extrair a movimentação do ator, usando a técnica de remoção de fundo da cena. Para isso o posicionamento da câmera e o cenário devem ser estáticos. A área extraída é segmentada para obtenção da localização 2D das principais partes do corpo humano que consiste em cabeça, mãos e pés. Para a estimação da localização das principais partes foi usada geometria computacional, já que o ator não possui marcações especiais. Em seguida é feito um tratamento nos dados com o objetivo de adquirir um movimento mais realístico da marionete e por fim os movimentos do ator são reconstruídos em uma marionete virtual com a estimação da localização 2D das partes encontradas.

Palavras chaves: Animação; Personagens Virtuais; Segmentação do Corpo Humano; Visão Computacional.

ABSTRACT

This work presents concepts for the localization and animation of human body parts in virtual personages from optical motion capture of the human movement without special markings on the actor body. It also describes the implementation of a software prototype that uses computational geometry for localization of the parts that compose the human body in a video file, that contains an actor in movement and in sequence its animation. The work has the objective of analyze the video archive and extract the actor's movement, using the technique of scene background removal. For this, the position of the camera and the scene must be static. The extracted area is segmented for obtainment of the 2D localization of the main human body parts that consists in head, hands and feet. For the esteem of the localization of the main parts it was used computational geometry, since the actor does not possess special markings. After that a treatment is made in the data with the objective to acquire a more realistic movement of the marionette and finally the movements of the actor are reconstructed in a virtual marionette with the esteem of the 2D localization of the encountered parts.

Key-Words: Animation; Virtual Personages; Human Body Segmentation; Computational Vision.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Hoje, há um grande número de aplicações que precisam determinar a localização das partes do corpo humano em algum sistema de coordenadas, ao longo de um intervalo de tempo. Em especial, pode-se citar as aplicações que envolvem animação de personagens humanos virtuais (jogos, filmes, desenhos animados, realidade virtual entre outras). Normalmente, esse tipo de aplicação captura os movimentos realizados por um ator, e posteriormente reproduz os mesmos movimentos no personagem virtual. Esse tipo de atividade é conhecido formalmente por captura de movimento, ou MoCap (Fernandes, 2002). Para a determinação das partes do corpo humano do ator, as tecnologias atualmente existentes para MoCap são: mecânica, ótica, eletromagnética e acústica, porém a que menos causa interferência na movimentação do ator é a tecnologia ótica.

A técnica de Captura de Movimento foi criada para dar a aplicações de animação, a visualização realística dos movimentos humanos, visto que os movimentos de um ser humano envolvem dezenas de músculos e articulações, sendo hoje impraticável uma representação fisicamente correta por meio de técnicas tradicionais como animação por quadros chaves (*keyframing*). A técnica MoCap foi utilizada pela primeira vez em filmes como Terminator 2: Judgment Day, e consiste em capturar a posição e orientação de certas partes de objeto em movimento utilizando processos óticos ou eletromagnéticos (SILVA; CAVALCANTI , 1996).

Na captura do movimento humano com o sistema óptico, normalmente são utilizadas marcações especiais para a identificação do posicionamento 2D das partes do corpo do ator. Um exemplo é o filme Final Fantasy (FERNANDES, 2004?), onde todos os personagens foram construídos com tecnologia digital. Para que os atores virtuais tivessem um movimento realístico, foram capturados os movimentos de atores reais. Cada ator vestia-se dos pés à cabeça com uma malha preta, a qual foram presas 37 tiras refletoras em pontos importantes do corpo. Os movimentos do ator foram capturados por 16 câmeras espalhadas pelo cenário, e foi identificado o posicionamento 2D das tiras refletoras em uma seqüência de vídeo de cada uma das câmeras. A partir deste posicionamento foi reconstruído o movimento 3D para o ator virtual.

A determinação da localização das partes do corpo humano sem o uso de marcações especiais é uma difícil tarefa, pois o ator pode assumir diferentes posturas o que dificulta a localização.

Neste trabalho pretende-se estudar técnicas para segmentação de silhueta humana em imagens, para identificação do posicionamento 2D das partes do corpo sem a utilização de marcações especiais e reconstrução dos movimentos 2D em uma marionete virtual a partir de um sistema óptico de captura de movimento humano.

O protótipo especificado e implementado neste trabalho tem como entrada um arquivo de vídeo que contém um ator em movimento. O arquivo de vídeo foi gravado com a câmera em uma posição fixa. Os primeiros segundos do arquivo correspondem ao aprendizado do cenário, isto é, não existe movimentação do ator. Passado o tempo de aprendizado o ator entra em cena, o *software* extrai as informações referentes ao movimento do ator utilizando remoção do cenário, que consiste na comparação entre os valores das componentes de cor que compõem os *pixels* do cenário atual e os *pixels* do cenário obtidos na etapa de aprendizado. Na próxima etapa do protótipo é feita a extração dos ruídos e a localização da figura humana com os *pixels* resultantes da etapa de remoção do fundo da cena. A partir da localização da figura humana são encontrados os pontos 2D na figura, que tenham maior probabilidade de pertencer a uma das principais partes do corpo ator (cabeça, mãos e pés) utilizando geometria computacional. Os pontos encontrados são classificados com sendo uma das partes principais do corpo humano, através das proporções entre os membros do corpo humano. Em seguida os pontos são corrigidos, interpolados e ajustados, uma marionete é animada com os pontos referentes à estimação 2D das principais partes do corpo do ator.

1.1 OBJETIVOS DO TRABALHO

O trabalho tem por objetivo fazer a segmentação de imagens para obtenção da localização das principais partes da silhueta humana, reconstrução dos movimentos e animação em uma marionete virtual a partir da seqüência de imagens de um arquivo de vídeo, e de polígonos que delimitam o contorno da silhueta humana nestas imagens.

Os objetivos específicos do trabalho são:

a) segmentar cada imagem do vídeo focando a área onde encontra-se a silhueta humana;
b) identificar as partes do corpo (pés, mãos, cabeça e tronco) do ator sem a utilização de marcações especiais.
c) determinar as coordenadas 2D das partes do corpo do ator mencionadas no item anterior;

d) reconstituir os movimentos originais do ator através da animação em uma marionete virtual.

1.2 ESTRUTURA DO TRABALHO

A seguir será apresentada uma breve descrição sobre a organização dos capítulos do trabalho.

O capítulo 2 apresenta os conceitos sobre geometria computacional necessários para identificação das partes do corpo humano, sem a utilização de marcações especiais sobre o corpo do ator. Apresenta também os conceitos de construção e animação de personagens virtuais.

No capítulo 3 são mostradas técnicas para a estimação do movimento humano usando captura óptica de movimento humano, que são utilizadas em trabalhos correlatos.

O capítulo 4 aborda os requisitos identificados, a especificação e implementação do protótipo, o capítulo 5 expõe as principais conclusões referentes ao trabalho, bem como sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

O corpo humano é uma estrutura complexa composta por dezenas de ossos, músculos e articulações, reproduzir o movimento humano de forma natural em um computador é uma tarefa difícil levando em consideração a flexibilidade entre os membros que o compõe e as várias posturas que ele pode assumir.

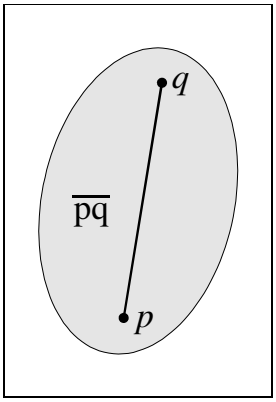
Em um sistema óptico de captura de movimento, a localização 2D das partes do corpo humano geralmente são obtidas através da identificação do posicionamento 2D de marcações especiais que estão em contato com o corpo do ator para identificação das mesmas.

Este capítulo apresenta os conceitos fundamentais necessários para identificação das principais partes do corpo humano que compõem o movimento humano, sem a utilização de marcações especiais. Na sessão 2.1 são abordados conceitos de geometria computacional necessários para a identificação das partes do corpo. Na sessão 2.2 são descritos os conceitos de animação, construção de personagens virtuais e as proporções geométricas de seus corpos.

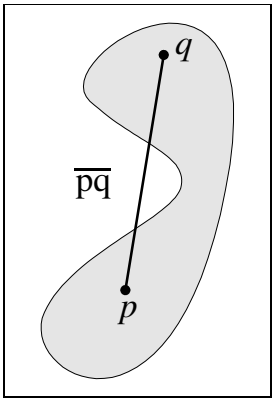
2.1 FECHO CONVEXO

O fecho convexo também chamado de envoltória convexa ou de *convex hull*, de um conjunto S de pontos é o menor polígono convexo P para qual cada ponto contido em S pertence ao interior ou ao limite de P (BERG, 2000).

Dada um conjunto finito de pontos S, o subconjunto de S é chamado de convexo se e somente se qualquer segmento de reta formado por um par de pontos p, q $\in$ S estiver inteiramente contido no subconjunto convexo (Cormen, 2000). As fig. 2.1 e 2.2 demonstram um fecho convexo e um não convexo respectivamente.



Fonte: Berg (2000, p.2)
Figura 2.1 – Fecho convexo



Fonte: Berg (2000, p.2)
Figura 2.2 – Fecho não convexo

Existem diversos técnicas para a obtenção do fecho convexo, entre elas estão:

a) varredura rotacional: os vértices são processados na ordem dos ângulos polares, que é formados com um vértice de referência. O algoritmo proposto por Graham disponível em Figueiredo (1991) e Cormen (2000) é conhecido como "Varredura de graham ou graham scan" e o algoritmo proposto por Jarvis conhecido como "embrulho para presente ou marcha de Jarvis" disponível em Cormen (2000) são implementados utilizando esta idéia;

b) método incremental: este método foi desenvolvido com a necessidade de se obter o fecho convexo de um polígono onde os vértices são adicionados dinamicamente, necessitando de um algoritmo que permita o incremento de pontos. O algoritmo de construção incremental disponível em Preparata (1988) foi desenvolvido utilizando essa idéia;

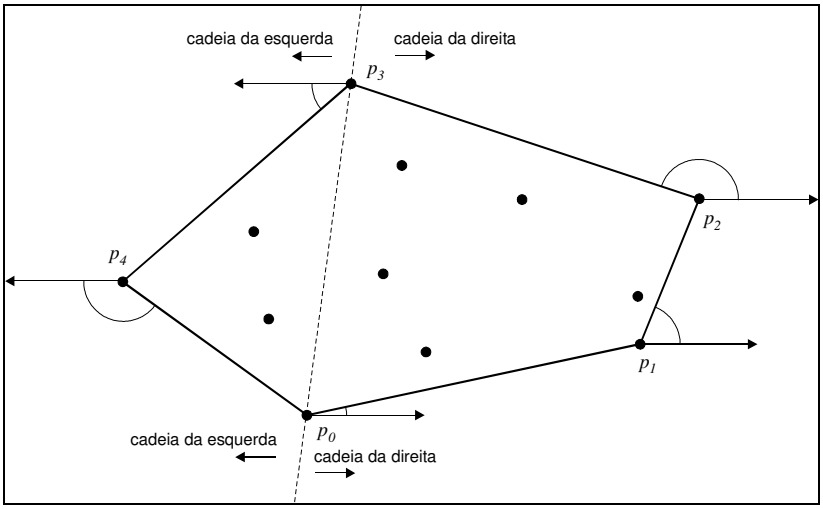
c) método dividir para conquistar: este método procura dividir o conjunto de pontos em conjuntos menores recursivamente para a obtenção do fecho convexo. Este método está dividido em três partes: a primeira, chamada de separação, onde o conjunto com *n* pontos é dividido em dois subconjuntos. A segunda é a obtenção dos fechos convexos sobre os subconjuntos adquiridos no passo anterior, que são calculados recursivamente e a terceira parte é chamada de combinação onde um algoritmo é usado para combinar os fechos convexos resultantes do passo anterior. Como exemplo, pode ser citado o algoritmo MergeHull disponível em Figueiredo (1991).

A seguir são descritos alguns desses algoritmos para obtenção do fecho convexo de um conjunto de pontos no plano.

2.1.1 Marcha de Jarvis

A marcha de Jarvis é um algoritmo usado para resolver o problema da obtenção do fecho convexo em um conjunto de pontos. Esse algoritmo também é conhecido como embrulha pacote ou embrulho para presente.

Consiste em obter um vértice p_0 que seja extremo dentro do conjunto de vértices *S* com a menor ordenada, esse ponto já faz parte do fecho convexo. O próximo ponto p_1 a ser inserido no fecho convexo é o ponto com o menor ângulo polar em relação ao ponto *p*. Se existir mais de um ponto com o mesmo ângulo, o ponto mais afastado é escolhido. De forma semelhante, o ponto p_2 tem o menor ângulo em relação ao p_1. Esse passo se repete até ser encontrado o ponto p_k com a maior ordenada. Esse conjunto de vértices é chamado de cadeia da direita conforme ilustra a fig. 2.3. Para construção da cadeia da esquerda o ponto inicial usado será o p_k que é representado na fig. 2.3 pelo ponto p_3, o próximo ponto a ser inserido p_{k+1} corresponde ao menor ângulo em relação ao p_k, mas a partir do eixo *x* negativo representado na fig. 2.3 pelo ponto p_4. Esse passo se repete até o ponto a ser encontrado seja igual ao ponto inicial p_0. Para a construção da cadeia da esquerda é sempre usado o ângulo polar a partir do eixo *x* negativo.



Fonte: Cormen (2000, p. 755)

Figura 2.3 – Desenho explicativo para o algoritmo Jarvis

O quadro 2.1 apresenta um pseudo-código para demonstrar a idéia descrita anteriormente, onde o *S* é o conjunto de pontos, *P* é o fecho convexo do conjunto *S*, e *p* é um ponto qualquer pertencente ao conjunto *S*. A função *menorAngulo(S, p)* retorna o ponto com o menor ângulo polar dentro do conjunto *S* em relação ao ponto *p*. Se existirem dois pontos dentro do conjunto *S* com o mesmo ângulo a função retorna o ponto com a maior distância. A função *menorAnguloEixoXNegativo(S,p)*, retorna o ponto com o menor ângulo polar dentro do conjunto *S* em relação ao ponto *p* a partir do eixo *x* negativo, se existirem dois pontos dentro do conjunto *S* com o mesmo ângulo a função retorna o ponto com a maior distância.

```
p[1] := ponto de ordenada mínima dentro do conjunto S;
i := 2;
p[i]  :=  menorAngulo(S,p[i]);
Enquanto p[i] <> p[1] faça
      se ( p[i] = ponto com maior ordenada dentro do conjunto S) então
           P[i+1] := menorAnguloEixoXNegativo(S,p[i]);
      senão
           P[i+1] := menorAngulo(S,p[i]);
      i := i+1;
   fim se
fim enquanto
```

Quadro 2.1 – Pseudo-código que demonstra o algoritmo proposto por Jarvis

O algoritmo determina o fecho convexo em um tempo O(*nh*) onde o *n* significa o número de pontos do polígono e *h* o número de vértices do fecho convexo. O pior caso para o algoritmo é quando todos os pontos do polígono fazem parte do fecho convexo o tempo é igual a $O(n^2)$, como mostra a Fig. 2.4 e 2.5.

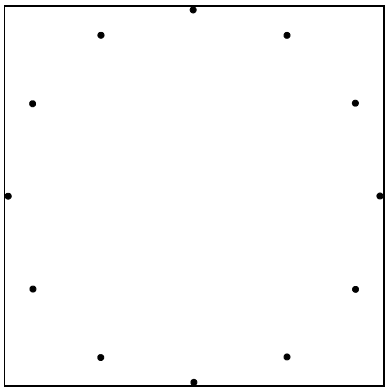


Figura 2.4 – Disposição dos pontos

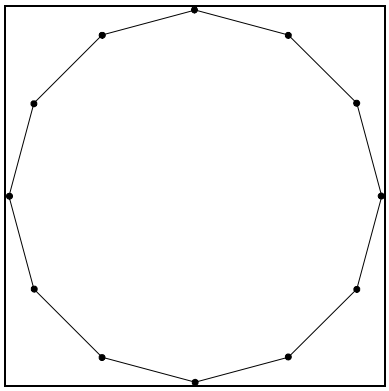


Figura 2.5 – Fecho convexo composto por todos os pontos

2.1.2 Varredura de Graham

O algoritmo de varredura de Graham soluciona o problema de obtenção do fecho convexo mantendo uma pilha de pontos candidatos, contendo cada um dos pontos do conjunto de entrada. Cada ponto é testado para verificar se faz parte do fecho. Em caso negativo, o ponto é retirado da pilha. Ao término do algoritmo, a pilha contém exatamente os vértices que fazem parte do fecho convexo ordenados da direita para esquerda (CORMEN, 2000).

Dado um conjunto de pontos finitos S = {P} com *n* pontos, onde $n \geq 3$. O algoritmo inicia procurando um ponto que faça parte do fecho convexo, como por exemplo o ponto de menor coordenada y. Esse ponto servirá como base ou pivô para encontrar o fecho convexo do conjunto de pontos {P} chamado de P_0. A partir do ponto P_0 são calculados os ângulos formados por cada um dos segmentos P_0P_i com o eixo horizontal x. Os pontos são ordenados e nomeados (P_1, P_2,... P_{n-1}) de forma crescente de acordo com o ângulo (se existir mais de ponto com o mesmo ângulo, o pontos são removidos do conjunto exceto o ponto mais distante) no sentido anti-horário ou da direita para esquerda. Como é mostrado na fig. 2.6 o ponto nomeado por P_2 tem ângulo (P_0, P_2) maior que o ângulo do ponto nomeado por P_1 (P_0, P_1).

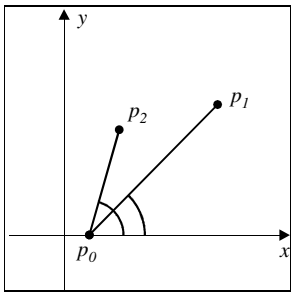
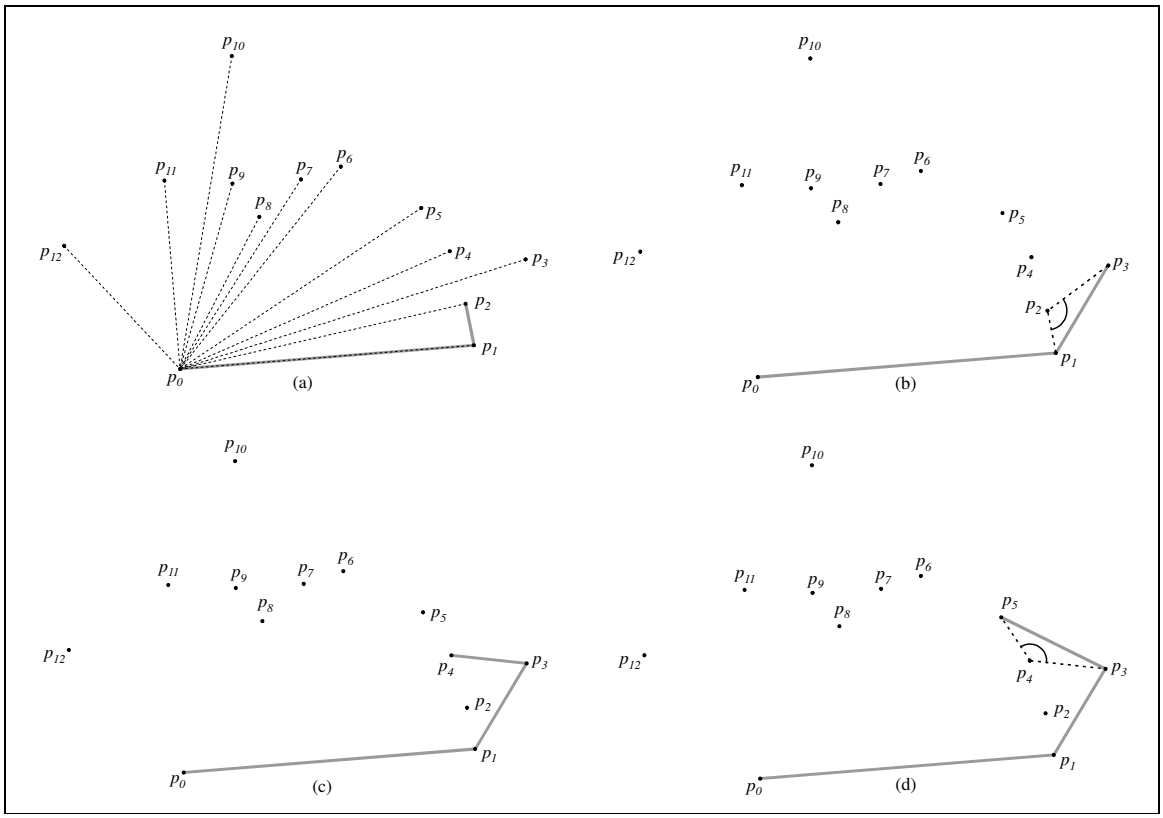
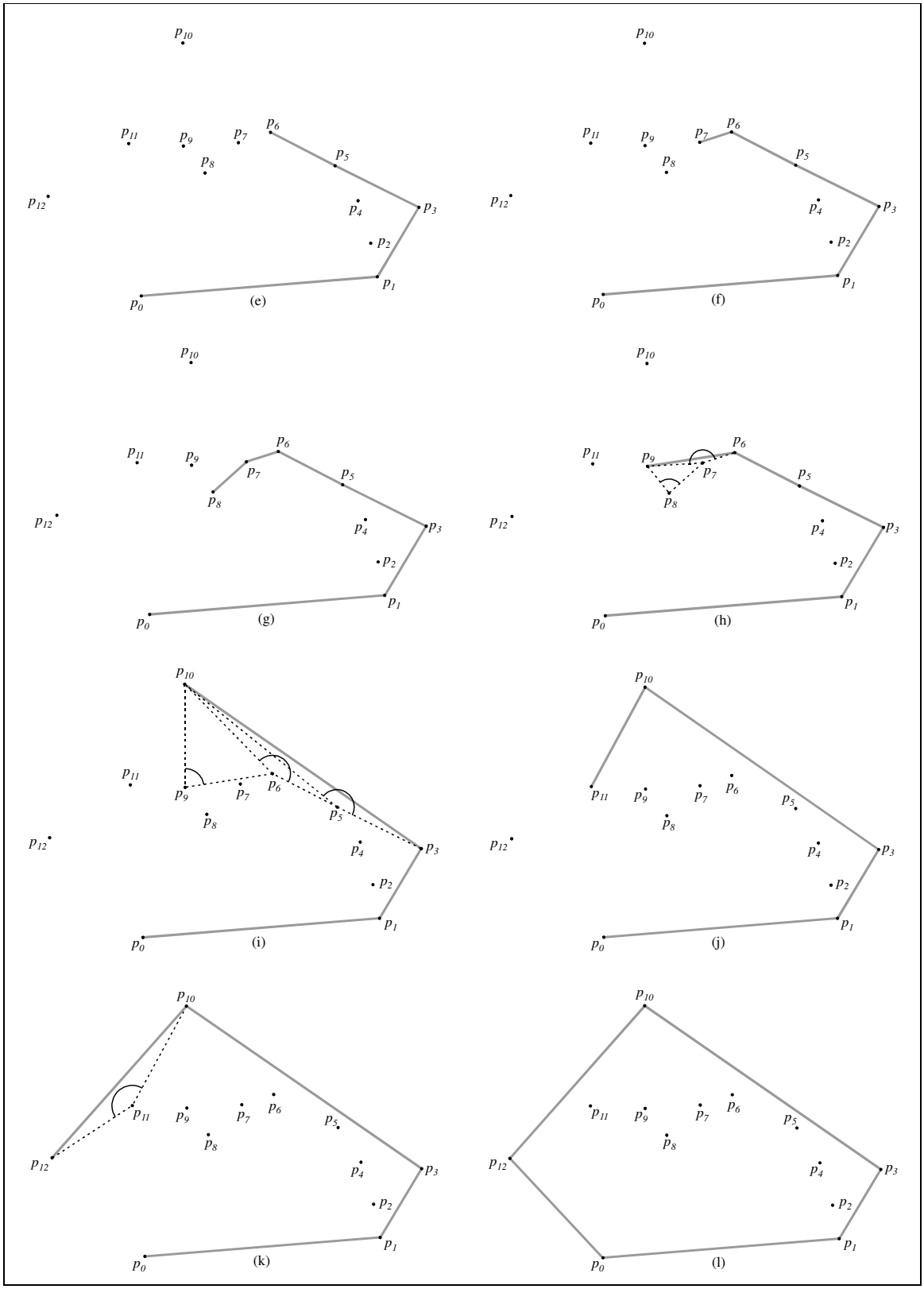


Figura 2.6 – Pontos ordenados pelo ângulo

O próximo passo é testar os pontos ordenados um a um para saber se o ponto faz parte dos vértices do fecho convexo. É usada a estrutura de pilha para armazenar os vértices/pontos que fazem parte do fecho convexo. Pode-se empilhar o P_0, que já faz parte do fecho convexo e o P_1, que pode ou não fazer parte do fecho convexo. A adição e remoção de outros pontos na pilha depende do posicionamento dos dois últimos elementos inseridos, por esse motivo a pilha é inicializada com P_0 e P_1.

Para verificar se o ponto P_k $(0 > k < n)$ faz parte do fecho convexo, é necessário saber se ele está a direita ou a esquerda da reta formada pelos dois últimos elementos inseridos na pilha. Se o ponto P_k estiver a direita da reta, significa que ele é mais externo que o ponto que se encontra no topo da pilha. Por esse motivo, o ponto que se está no topo da pilha precisa ser removido. A remoção do topo da pilha é realizado enquanto o ponto P_k estiver a direita do segmento formado pelos dois últimos pontos inseridos. O conteúdo da pilha está sendo atualizado, ou seja, estão sendo removidos os pontos que são internos ao fecho convexo. Quando o ponto P_k estiver do lado esquerdo do segmento de reta formado pelos dois últimos pontos inseridos na pilha, esse ponto faz parte do fecho convexo "parcial" e por isso também deve ser inserido na pilha. O procedimento é repetido para o próximo ponto a ser analisado P_{k+1} $(0 > k+1 < n)$ contido no conjunto de pontos ordenados pelo ângulo. A fig. 2.7 mostra o algoritmo sendo executado passo a passo.





Fonte: Cormen (2000, p. 752-753)

Figura 2.7 – Ilustração da seqüência de passo utilizados no algoritmo Graham

Na fig. 2.7a mostra a seqüência de pontos já nomeadas em ordem crescente de acordo o ângulo polar formado em relação ao P_0 que é o ponto com menor coordenada y, mostra também a verificação do ponto P_2 em relação ao segmento formado pelos pontos P_0 e P_1. Como o ponto P_2 está à esquerda do segmento, ele é inserido na pilha. Na fig.2.7b é feito o mesmo teste para o ponto é P_3. Como o ponto se encontra à direita do segmento de reta formado pelos pontos P_1 e P_2, isso implica na remoção do ponto P_2 da pilha, ou seja, o ponto P_2 não faz parte do fecho convexo porque o ponto P_3 é mais externo que ele. Na fig.2.7c o ponto P_4 é inserido na pilha visto que a sua localização está à esquerda do segmento de reta formado pelos pontos P_1 e P_3. Na fig. 2.7d o ponto P_4 é removido, pois o ponto P_5 é mais externo que P_4. Da fig.2.7e a fig. 2.7k mostra-se o conjunto de pontos sendo avaliados de acordo com os dois últimos pontos da pilha. Na fig. 2.7l destaca o conjunto de pontos que fazem parte do fecho convexo e o conjunto de pontos inicial. No quadro 2.2 é apresentado um pseudo-código onde o algoritmo é descrito.

```
Graham(S){
      P0 = ponto com o menor ou maior x, ou menor ou maior y contido em S;
      P = Ordenação do conjunto S de acordo com o ângulo formado por P0Pi;
      Empilha P0 e P[1] em convexHull;
      topo = 2;
      i = 3;
      Enquanto (i < qtdeDePontosDeS) faça
            enquanto ( P[i] está a direita da linha convexHull[topo] e
      convexHull[topo-1]) faça
                  retira o topo da pilha convexHull;
                  topo = topo -1;
            empilha P[i] em convexHull;
            topo = topo+1;
      return convexHull;
}
```

Quadro 2.2 – Pseudo-código que representa o algoritmo proposto por Graham

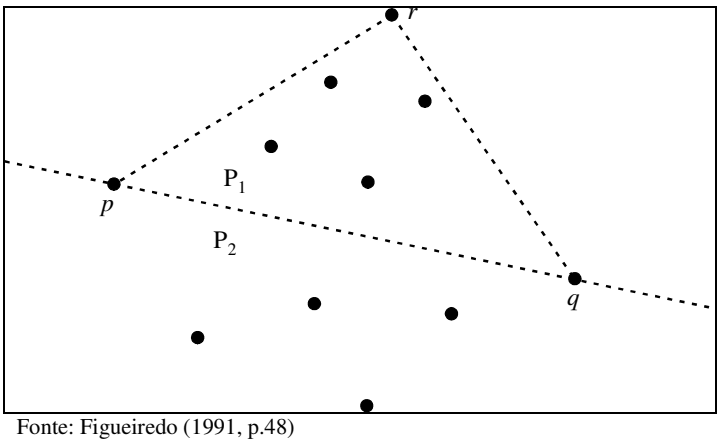
Segundo Cormen (2000) e Figueiredo (1991) o tempo gasto para a execução do algoritmo proposto por Graham é de O(n log n).

2.1.3 QuickHull

O algoritmo Quickhull divide recursivamente o conjunto *P* em subconjuntos *P1* e *P2* de forma que o fecho convexo é obtido pela concatenação dos fechos convexos dos subconjuntos.

Dados dois vértices, *p* e *q* que fazem parte do fecho convexo de *P*, esses vértices podem ser obtidos encontrando-se os pontos contidos em *P* que tenham a menor e a maior

abscissa ou a menor e a maior ordenada. Os subconjuntos P_1 e P_2 são obtidos através dos pontos P que estão contidos em um plano formado pela reta p e q. Para a obtenção do fecho convexo do subconjunto P_1, é selecionado o ponto r cuja a área formada pelo triângulo p,q r seja máxima ou seja, o ponto r que está mais distante em relação a reta p e q, fazendo isso é garantido que o ponto r é um ponto que faz parte do fecho convexo de P_1 (Figueiredo, 1991). Na fig. 2.8 essa idéia é ilustrada.



Fonte: Figueiredo (1991, p.48)

Figura 2.8 – Ilustração de um passo do algoritmo QuickHull

Um novo subconjunto P_{11} é formado a partir dos pontos situados a esquerda da reta p,r e um outro P_{12} a partir dos pontos situados a direita da reta r,q.

O Resultado do fecho convexo de P_1 é a combinação dos fechos convexos P_{11} e P_{12}. Esse algoritmo é chamado recursivamente, dividindo o conjunto em dois subconjuntos de pontos e buscando o mais externo em relação aos que já fazem parte do fecho convexo.

No quadro 2.3 mostra um pseudo-código do algoritmo resultante da idéia descrita acima. O método tem a seguinte assinatura *quickHull(pts, p, q)*, onde o *pts* é um conjunto de pontos, p e q são os pontos extremos do conjunto de pontos.

```
QuickHull(pts, p, q){
      Se pts = {p,q} então
         retorne o segmento orientado pq;
      else
         r = ponto pts tal que a distancia seja máxima;
         p1 = pontos a esquerda de de qr;
         p2 = pontos a direita de rp;
```

Quadro 2.3 – Pseudo-código que representa o algoritmo QuickHull

Segundo Figueiredo (1991) o desempenho do algoritmo é de $O(n^2)$.

2.2 ANIMAÇÃO

Animação de personagens é a modificação de parâmetros em um determinado tempo, de tal forma que possamos perceber o movimento de forma natural. Em outras palavras, animação significa dar vida a personagens. Geralmente é usado o mapeamento dos movimentos de um objeto real para um personagem virtual, tentando reproduzir de forma natural os movimentos no personagem.

Segundo Silva (1998) o cartunista Winsor McKey obteve movimentação de um personagem com o desenho deste, com pequenas variações, em múltiplos pedaços de papéis, que foram visualizados em uma certa taxa de amostragem, gerando assim o conceito de animação no ano de 1911.

Para um personagem ser animado, é preciso conhecer a sua geometria, a maneira como as partes do personagem estão relacionadas e os movimentos que o personagem pode executar, para que animação seja o mais natural possível.

Nas próximas sessões, serão apresentadas técnicas para construção de personagens que representam a forma humana. Na sessão 2.2.1 são descritos modelos geométricos para criação de personagens que podem ser animados. Na sessão 2.2.2 são abordadas as proporções entre os membros que compõem o personagem.

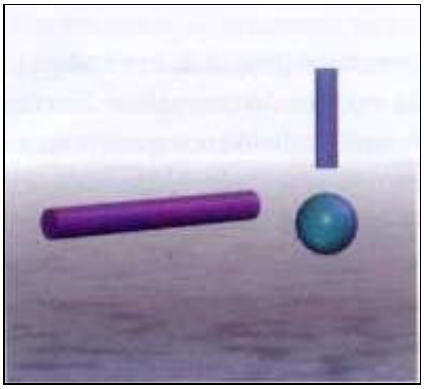
2.2.1 Modelos de representação do corpo humano

Apesar do corpo humano assumir vários tipos de postura, ele possui uma forma bem definida o que não permite que a alteração do posicionamento relativo entre os membros que o compõem. No contexto de captura de movimento, a criação de um modelo de representação do corpo humano ajuda no rastreamento do movimento do ator e conseqüentemente a sua localização.

A criação de um modelo para o um personagem animado é uma maneira de simplificar manipulação dos movimentos do personagem pelo computador. Segundo Maestri (1996), construir e configurar o personagem adequadamente antes do inicio da animação pode evitar inúmeros problemas.

2.2.1.1 Personagem segmentado/articulado

Personagem segmentado é uma das maneiras mais simples para a construção de personagens, onde as peça do corpo (segmentos) são criadas separadamente e em seguida são conectadas através de articulações que dão ao personagem flexibilidade na movimentação. A construção das peças individuais é formada por figuras geométricas, geralmente cilindros, caixas e esferas. A fig. 2.9 mostra peças geométricas utilizadas para construção de personagens segmentados.



Fonte: Maestri (1996, p.30)

Figura 2.9 – Cilindros e esferas

As articulações são responsáveis por conectar os segmentos. Para que o modelo tenha um movimento realístico, é necessário informar o grau de liberdade que cada articulação possui, ou seja, é necessário informar o ângulo e a direção que os segmentos podem se mover.

Na Tabela 1, são mostrados uma lista de articulações (junções) do corpo humano e seus limites de rotação. O parâmetro "tipo" indica o(s) eixo(s) que a junta pode ser movimentada.

Tabela 1 – Limites rotacionais das juntas humanas

Segmento	**Junta**	**Tipo**	**X**	**Y**	**Z**
Pé	Tornozelo	Rotacional	65º	30º	0º
Perna	Joelho	Articulada	135º	0º	0º
Coxa	Quadril	Esfera/Órbita	120º	20º	10
Espinha	Quadril/Espinha	Rotacional	15º	10º	0º
Ombro	Espinha	Rotacional	20º	20º	0º
Bíceps	Ombro	Esfera/Órbita	180º	105º	10º
Antebraço	Cotovelo	Articulada	150º	0º	0º
Mão	Pulso	Esfera/Órbita	180º	30º	120º

O personagem cuja geometria consiste em segmentos e articulações, não pode se locomover de forma separada. Ele deve se mover como unidade que é a representação do

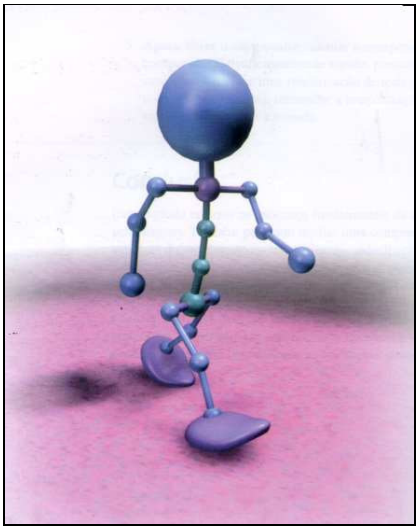
corpo humano. Para isso pode ser utilizada a estrutura de hierarquia do corpo humano, onde é definida a ordem de posicionamento e a ligação entre os membros.

Essa técnica é usada para indicar ao computador como o personagem foi montado e conseqüentemente, quais as peças que estão subordinadas a outras. Essa estrutura cria uma relação de parentesco entre os membros do corpo, modelando a movimentação do corpo de forma natural. Cada segmento pode ter um pai e pode ter zero, um ou mais filhos. Na fig. 2.10 é mostrada a estrutura do corpo humano de maneira hierárquica.

Fonte: Maestri (1996, p. 29)

Figura 2.10 – Modelo hierárquico do corpo

No caso do corpo humano, o quadril é o elemento raiz da estrutura. Se o quadril se movimentar, todos os demais segmentos serão movimentados. A fig. 2.11 mostra o resultado de um personagem segmentado.



Fonte: Maestri (1996, p. 204)
Figura 2.11 – Personagem segmentado

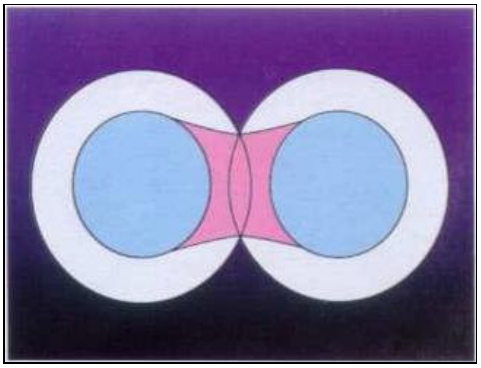
A movimentação das peças que formam os personagens pode ser feita da seguinte forma:

a) cinemática direta: a movimentação das peças do corpo se dá a partir do topo da hierarquia. A movimentação do pai faz movimentar o filho;
b) cinemática inversa: é o inverso da direta, onde a movimentação do filho faz a movimentação do pai. Se uma mão for movimentada, o braço (pai) tem que se movimentar junto. O limite de movimentação dos membros do corpo do personagem é imposto pelas articulações e estão descritos na Tabela 1.

Segundo Maestri (1996), a criação das partes do corpo separadas torna mais simples a construção do personagem, e mais fácil o seu manuseio ao ser animado.

2.2.1.2 Personagens modelados com metaballs

É uma técnica para criação de personagens arredondados, consiste em esferas que juntam como glóbulos quando são renderizadas. A cada metaball é associado um peso e um raio de influência. A distancia entre os metaballs e os valores das variáveis (peso e influência) de cada metaball é responsável pela junção das esferas. Na fig. 2.12, são mostradas duas esferas e seus raios de influência. Na fig. 2.13, são mostradas as esferas renderizadas.



Fonte: Maestri (1996, p. 37)
Figura 2.12 – Esferas e raio de influencia

Fonte: Maestri (1996, p. 37)
Figura 2.13 – Esferas renderizadas

A fusão de muita esferas acarreta na construção de figuras complexas. Para construção de um corpo humano é necessário criar esferas de acordo com o formato do corpo e ajustar o tamanho, o peso, e o raio de influência de cada uma. O resultado da criação de um personagem usando essa técnica é mostrado na fig. 2.14.

Fonte: Maestri (1996, p. 38)
Figura 2.14 – Corpo humano modelado com *metaballs*

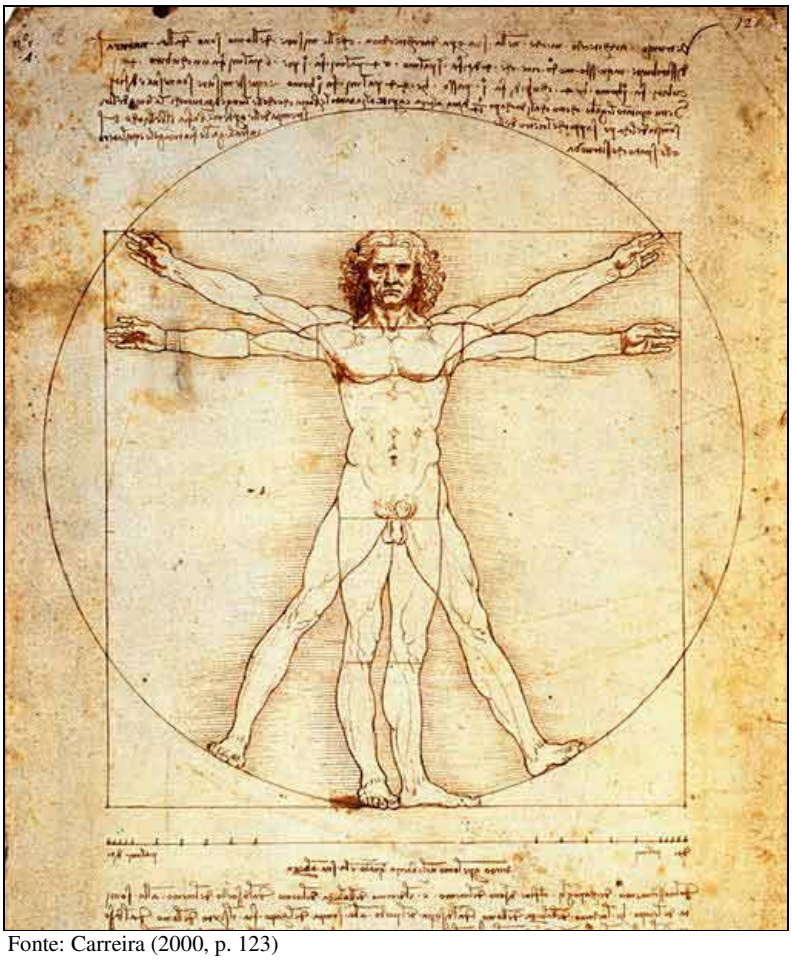
A animação de personagens usando essa técnica pode ser feita animando cada esfera individualmente, o que permite alterar a forma do personagem sem alterar a ligação entre as esferas, ocasionando uma ligeira deformação no personagem.

Para animação de um personagem com características humanas, é preciso ter várias esferas, o que torna inviável o método citado no parágrafo anterior. Neste caso, pode ser construído um esqueleto que seria responsável pela movimentação do personagem revestido pelo conjunto de metaballs. O esqueleto pode ter uma estrutura hierárquica e ser animado

usando cinemática direta ou inversa, a movimentação do esqueleto faz os metaballs se moverem também.

2.2.2 Proporção entre os membros do corpo

Leonardo da Vinci em sua obra denominada “uomo vitruviano” traduzindo para português significa “homem Vitroviano” é um dos seus desenhos mais conhecidos e perfeitos, mostrado na fig. 2.15.



Fonte: Carreira (2000, p. 123)

Figura 2.15 – Uomo vituviano

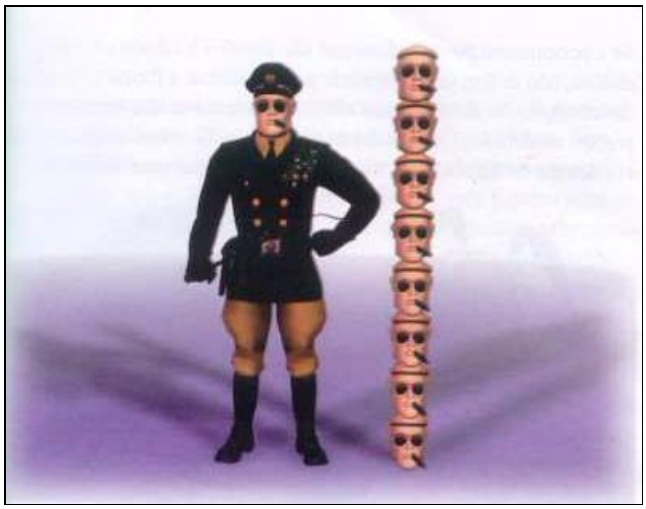
Segundo Carreira (2000) a obra foi baseada no livro *De architetura, III, 1* escrito pelo arquiteto Vitruvio em 70 AC. Nesta obra, Leonardo da Vinci escreveu o seguinte comentário sobre as proporções do corpo humano.

> **Ven**. S/□um
>
> O arquiteto Vitrúvio diz, em sua obra sobre a arquitetura, que as medidas de um homem estão por natureza, distribuídas desse modo: quatro dedos fazem um palmo e quatro palmos fazem um pé, seis palmos fazem um cúbito, quatro cúbitos fazem um

> homem. E que, quatro cúbitos fazem um passo e, 24 palmos fazem um homem. Estas medidas estão em seus edifícios. Se você abrir tanto as pernas a ponto de sua altura diminuir em 1/14, e estender ou levantar os braços até alcançar, com os dedos médios a linha que delimita o extremo superior da cabeça; você verá que o centro dos membros estendidos será o umbigo e que o espaço compreendido entre as pernas formará um triângulo eqüilátero.
>
> O comprimento dos braços estendidos de um homem é igual a sua altura.
>
> Da raiz dos cabelos até a base da barba, temos 1/10 da altura do homem; da base da barba até o extremo superior da cabeça, 1/6 do homem; do extremo superior do peito até a raiz dos cabelos, 1/7 de todo o homem; dos mamilos até o extremo superior da cabeça, 1/4 do homem. A largura máxima dos ombros contém em si 1/4 do homem; do cotovelo até a ponta da mão, 1/5 do homem; desse mesmo cotovelo até o termino do ombro, 1/8 desse homem. Toda a mão 1/10 do homem; o membro viril nasce no centro do homem; o pé é 1/7 do homem; da planta do pé até a parte inferior do joelho temos 1/4 do homem; da parte inferior do joelho até a base do membro, 1/4 do homem. Os espaços compreendidos entre o queixo e o nariz, e entre a raiz dos cabelos e as sobrancelhas são iguais e equivalem à orelha, a saber, 1/3 do rosto. (CARREIRA, 2000, p. 122).

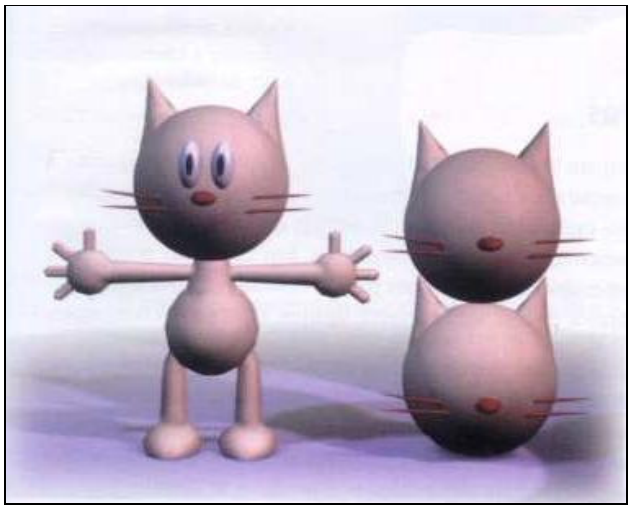
Leonardo da Vinci em seu comentário estabelece uma relação de proporção entre os membros do corpo humano, no entanto essas proporções não perfeitas, foram baseadas em antigas tradições de modularização do corpo humano, portanto devem ser entendidas como proporções aproximativas (CARREIRA, 2000).

Segundo Maestri (1996) a proporção de altura entre corpo e a cabeça de um ser humano mediano é de aproximadamente oito cabeças, para que o personagem se mostre realístico, como mostra a fig. 2.16. Para personagens em quadrinhos e desenhos animados essas relações podem ser bem diferentes, quanto maior a cabeça mais graciosa se torna o personagem, como é mostrado na fig. 2.17.



Fonte: Maestri (2000. p. 15)

Figura 2.16 – Proporção entre a altura do personagem realístico e a altura da cabeça



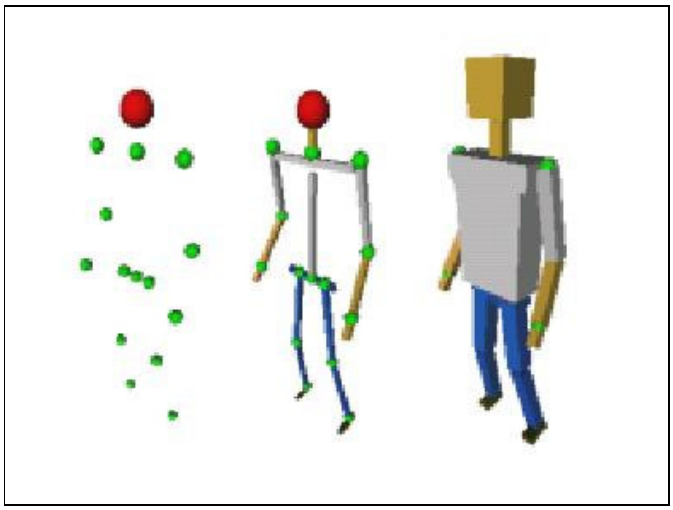
Fonte: Maestri (2000. p. 15)

Figura 2.17 – Proporção entre a altura do personagem animado e a altura da cabeça

Neste capítulo foram apresentados alguns conceitos sobre geometria computacional, necessários para a extração da localização 2D das principais partes do corpo do ator. Também foram abordadas técnicas de criação e animação de personagens virtuais. O próximo capítulo está reservado para a descrição de técnicas usadas para a estimação do movimento humano usando captura óptica de movimento, que foram empregadas em trabalhos correlatos.

3 TRABALHOS CORRELATOS

Silva (1998) descreve um sistema de animação baseado em movimento capturado. O sistema proposto analisa, modifica e reutiliza os dados capturados. Utiliza um modelo articulado com 16 segmentos rígidos conectados por articulações com um total de 33 graus de liberdade, conforme mostra a figura 3.1.

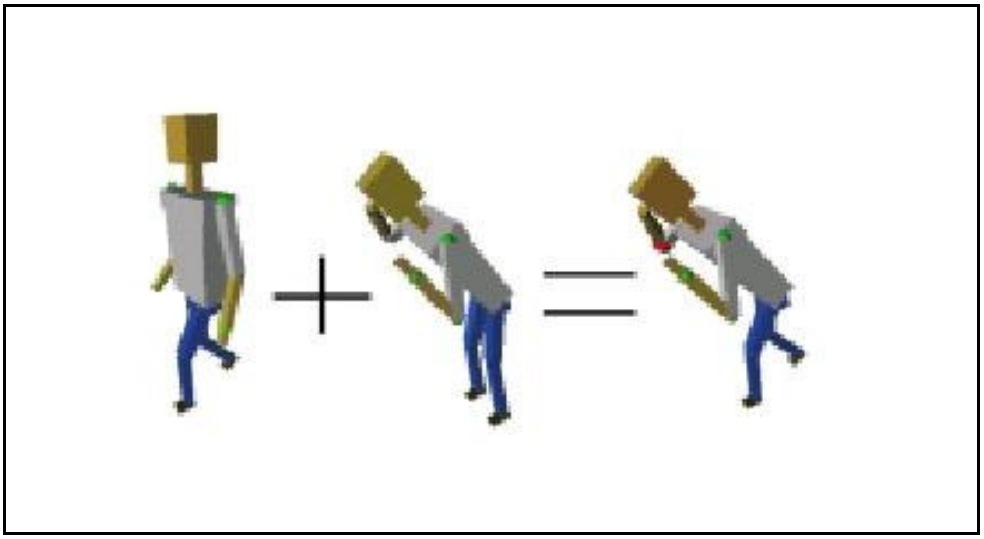


Fonte: Silva (1998 p. 3)

Figura 3.1 – Modelo ator virtual

Para animação do ator foi usado o modelo de estrutura hierárquica. Um sistema de captura de movimento humano armazena as informações sobre a posição e orientação global das marcações posicionadas sobre o corpo do ator real. Esses dados servem como entrada para o sistema de animação transformando as posições globais dos marcadores para ângulos relativos associando aos segmentos do modelo hierárquico.

Depois que os dados foram mapeados para o modelo, o sistema permite concatenar posturas através da junção do movimento de cada articulação para obter uma animação mais realística, conforme mostra a fig. 3.2.



Fonte: Silva (1998 p. 11)

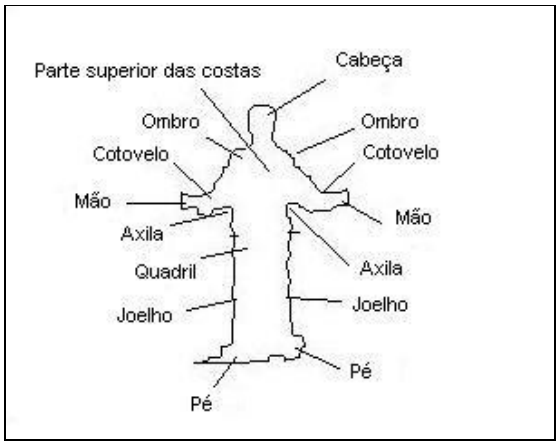
Figura 3.2 – Combinação de diferentes posturas

Este procedimento é útil para ajustar os dados obtidos com captura de movimento, visto que muitas informações sobre a localização das partes do ator podem ser omitidas devido a falhas no processo de captura.

Haritaoglu, Harwood e Davis (1998) propuseram um sistema que se baseia no modelo do corpo humano para descrever a estimação da postura e localização das partes do corpo para posturas genéricas em tempo real usando imagens monocromáticas.

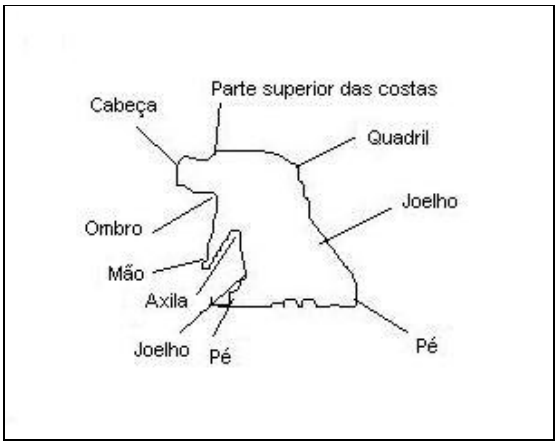
Este trabalho procura identificar as partes do corpo humano através da combinação da topologia do corpo com geometria computacional, primeiro identificando as partes primárias do corpo (cabeça, mãos, pés e torso) e depois as partes secundárias (cotovelos, joelhos, ombros, axilas, quadril e a parte superior das costas).

Para a identificação das partes do corpo humano, são analisadas quatro posturas corporais: em pé, sentando, engatinhando, deitando. Para cada postura é determinada a ordem de localização dos membros. Na fig. 3.3, mostra a ordem de localização dos membros para a postura em pé. Na fig. 3.4 mostra a ordem de localização dos membros para a postura em que a pessoa está engatinhando.



Fonte: adaptado de Haritaoglu, Harwood e Davis (1998, p. 2)

Figura 3.3 – Pessoa em pé



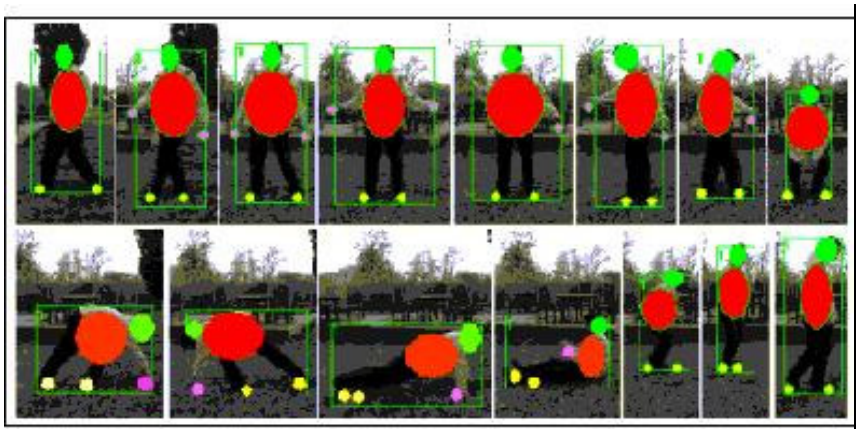
Fonte: adaptado de Haritaoglu, Harwood e Davis (1998, p. 2)

Figura 3.4 – Pessoa engatinhando

O processo de identificação das partes que compõe o corpo humano está divido em quatro etapas:

a) é feita uma análise para descobrir a postura em que a silhueta se encontra, computando as similaridades entre as projeções verticais e horizontais com os histogramas que armazenam as projeções para as quatro principais posturas que o corpo humano pode assumir;
b) um algoritmo de fecho convexo é executado recursivamente para a obtenção das partes do corpo que se encontram no contorno da silhueta detectada;
c) a localização da cabeça é predita usando o maior eixo da silhueta;
d) a partir da localização da cabeça são impostas regras topológicas dependendo da postura estimada para um mapeamento dos vértices encontrados e os membros do corpo.

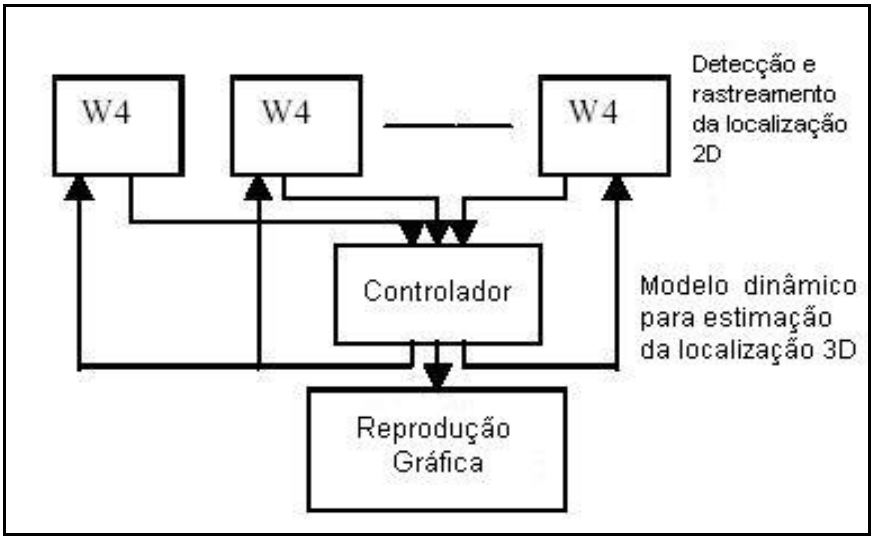
Exemplos de resultados obtidos pelo sistema são mostrados na fig. 3.5, onde a área pintada da cor verde corresponde à cabeça do ator, a área rosa corresponde às mãos, a amarela aos pés e a área vermelha corresponde ao torso.



Fonte: Haritaoglu (1998, p. 8)

Figura 3.5 – Reconhecimento das partes do corpo

O trabalho de Horprasert et al (2000) descreve um sistema de visão computacional 3D em tempo real para detecção e rastreamento do movimento humano. A pessoa é vista por múltiplas câmeras. O sistema é executado em rede onde, para cada câmera o processamento, é executado em um micro-computador detectando o rastreamento e o posicionamento 2D das partes do corpo para aquele ponto de vista. Os dados são enviados para um outro computador onde é reconstruída a localização 3D, e um modelo dinâmico é reproduzido em tempo real em um ambiente gráfico 3D. A fig. 3.6 mostra a visão geral da estrutura utilizada no trabalho.



Fonte: Horprasert et al (2000, p.1)

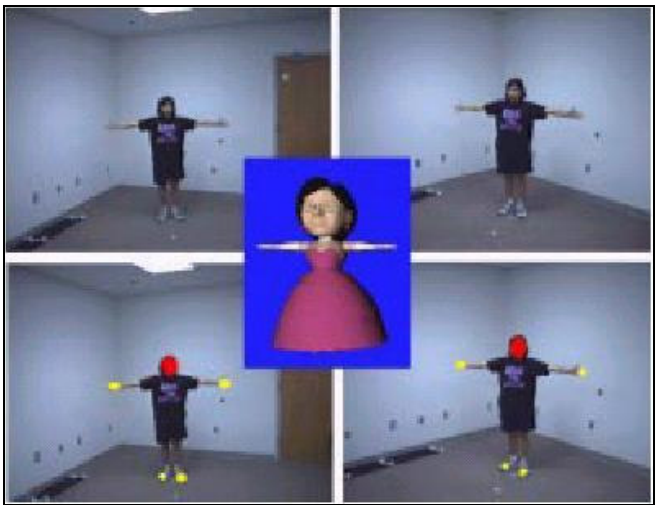
Figura 3.6 – Visão geral da estrutura do sistema

Para ajudar no processo de rastreamento do corpo e na localização das articulações 3D são usados modelos cinemáticos dinâmicos e o filtro de Kalman (*Kalman filter*).

O filtro de Kalman é um algoritmo recursivo, utilizado para prever os estados de um sistema dinâmico, baseado na medição de confiabilidade dos dados observados a partir de uma matriz denominada matriz de covariâncias, calculada a cada novo processo de estimação em que o filtro é utilizado. A partir de uma posição inicial conhecida, este algoritmo combina todos os dados disponíveis adicionados ao conhecimento anterior do sistema e de seus dispositivos, para produzir uma estimativa das variáveis desejadas de tal forma que o erro seja reduzido estatisticamente ao longo do tempo. Consiste em duas etapas: a fase de propagação e a fase de atualização.

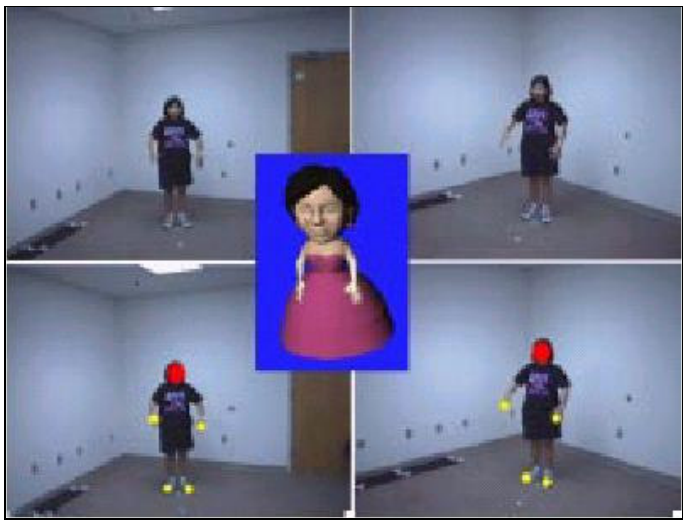
Este método tem por natureza recursiva e não necessita armazenar as medidas previamente em grandes matrizes, tornando ideal para ser usado em tempo real.

O exemplo do resultado do trabalho é mostrado nas fig. 3.7, 3.8 e 3.9.

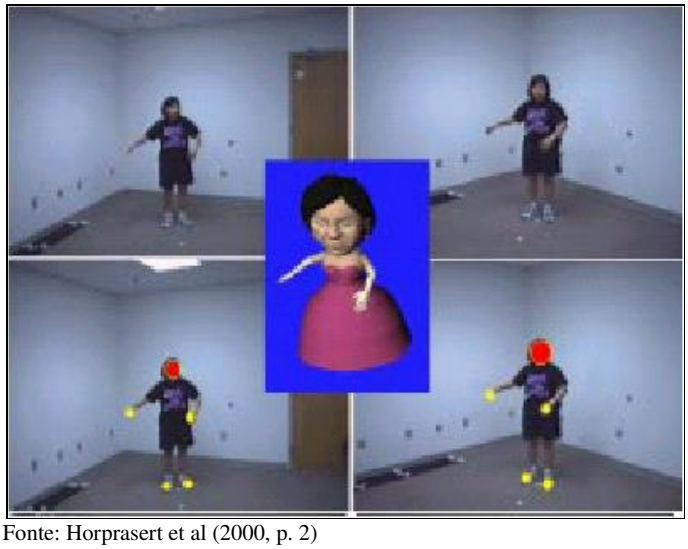


Fonte: Horprasert et al (2000, p. 2)

Figura 3.7 – Reconhecimento da postura e animação do ator virtual



Fonte: Horprasert et al (2000, p. 2)
Figura 3.8 – Reconhecimento da postura e animação do ator virtual



Fonte: Horprasert et al (2000, p. 2)
Figura 3.9 – Reconhecimento da postura e animação do ator virtual

O trabalho proposto por Fernandes (2002) utiliza a técnica de captura de movimento através de um sistema óptico. No trabalho são implementadas as primeiras etapas de um sistema de animação de personagens virtuais sem a utilização de marcações especiais. Foi implementada a etapa de captura dos dados de origem, vindos de câmeras e de arquivos de vídeo. Para detecção dos objetos em movimento foi utilizada a técnica de remoção do fundo da cena. Existe também a etapa de análise, onde os objetos em movimento detectados são analisados para localização da provável figura humana.

O presente trabalho tem por objetivo dar continuidade ao trabalho iniciado por Fernandes (2002) implementando as etapas que faltam para a reconstrução dos movimentos do ator, que consiste em segmentar a área que corresponde a localização da provável figura humana e estimando a localização 2D das principais partes do corpo humano, para cada frame do arquivo de vídeo. Em seguida é gerada uma animação com o intuito de reconstruir os movimentos do ator real.

4 DESENVOLVIMENTO DO PROTÓTIPO

Com base nos conceitos apresentados nos capítulos anteriores, tornou-se possível o desenvolvimento do protótipo. Neste capítulo serão abordados requisitos identificados, a especificação, implementação do protótipo.

4.1 REQUISITOS IDENTIFICADOS

Para o desenvolvimento do protótipo foram identificados os seguintes requisitos:

a) executar tratamento sobre o arquivo de vídeo (RF), que será usado como entrada de dados. Este arquivo contém imagens do ator se movimentando;
b) identificar o posicionamento das principais partes do corpo do ator (cabeça, mãos, pés e torso) (RF), focando a área delimitada pelo polígono que contém o contorno da silhueta, levando em consideração a topologia do corpo humano;
c) determinar as coordenadas 2D das principais partes do corpo do ator, para cada quadro do arquivo de vídeo (RF);
d) gerar de um arquivo contendo as coordenadas 2D do ator (RF);
e) reproduzir os movimentos (RF) do ator em uma marionete virtual, tendo como entrada o arquivo gerado na etapa anterior.

4.2 ESPECIFICAÇÃO

Para construção do protótipo, foi utilizado o conceito de programação orientada a objetos (AMBLER, 1997). A notação UML (*Unified Modeling Language*) foi usada para a modelagem das classes, utilizando-se a ferramenta *Enterprise Architect*.

O sistema especificado está dividido em duas partes, a primeira é responsável por analisar os pontos referentes a silhueta humana, obter o posicionamento 2D das principais partes do corpo do ator e animar uma marionete com a localização obtida. Na segunda parte do trabalho é feita uma correção dos dados encontrados na etapa anterior, e é gerada uma animação 2D suavizada a partir dos dados.

Os pontos referentes à silhueta humana foram obtidos através de um sistema óptico de captura de movimento. Para a obtenção da silhueta humana, deve-se obedecer alguns requisitos: a câmera deve ter posicionamento estático e não pode sofrer alteração no foco; o cenário pode ser qualquer ambiente estático e não reflexivo, sob qualquer tipo de iluminação,

desde que a iluminação esteja presente e não sofra variação de intensidade nem alteração no posicionamento da fonte; a única movimentação no vídeo deve ser a do ator; o formato do arquivo deve ser *Audio Video Interleaved* (AVI), utilizando o compressor Indeo® video 5.04. A imagem deve ter o tamanho de 320x240 *pixels*, e os *pixels* 24bits de profundidade de cor, utilizar o sistema de cores RGB e trabalhar a uma freqüência de 30 quadros/s. Para garantir o funcionamento do sistema, a trilha de vídeo deve conter apenas o fundo da cena durante o tempo de aprendizagem, que varia de computador para computador. Após esse tempo de aprendizagem o ator está habilitado a entrar em cena. Mais detalhes sobre o sistema óptico de captura de movimento humano pode ser conferidas em Fernandes (2002).

A primeira parte do desenvolvimento do protótipo teve os seguintes passos:

a) tratamento do arquivo de vídeo, dados de entrada;
b) remoção do cenário para identificação da figura humana;
c) segmentação da figura humana;
d) identificação das principais partes da figura humana, e geração de um arquivo contendo o posicionamento das partes do corpo;
e) animação das partes encontradas.

A Segunda parte do desenvolvimento protótipo tem as seguintes etapas:

a) leitura do arquivo e correção da altura do ator baseado na média das alturas identificadas;
b) interpolação dos movimentos do ator, quanto não é identificado alguma parte do ator devido a oclusão;
c) ajuste no pontos;
d) animação da marionete com os dados atualizados.

A fig. 4.1 mostra o diagrama de classes responsáveis pela a primeira parte do desenvolvimento do trabalho.

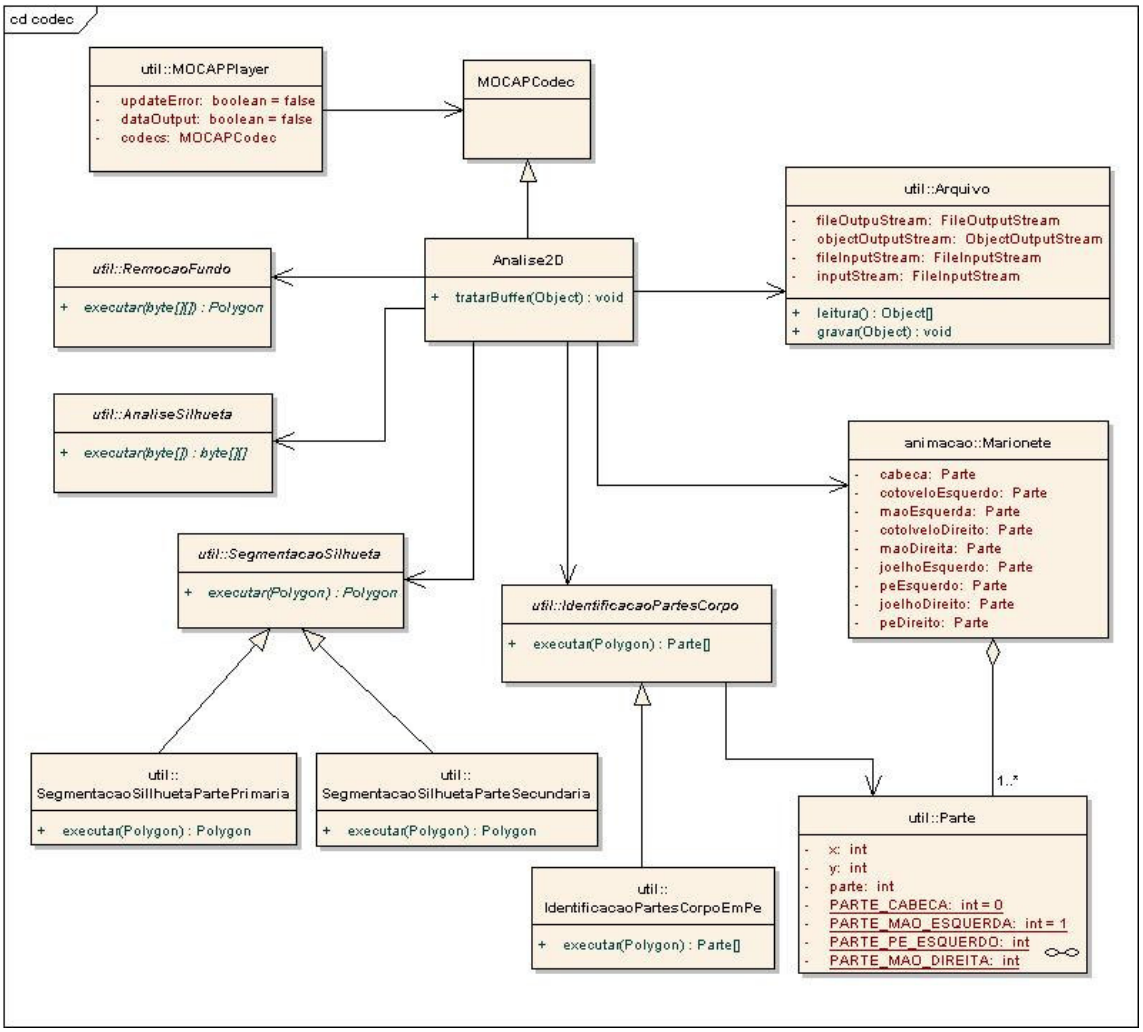


Figura 4.1 – Diagrama de classes

A fig. 4.2 corresponde ao diagrama de classes da segunda parte do trabalho.

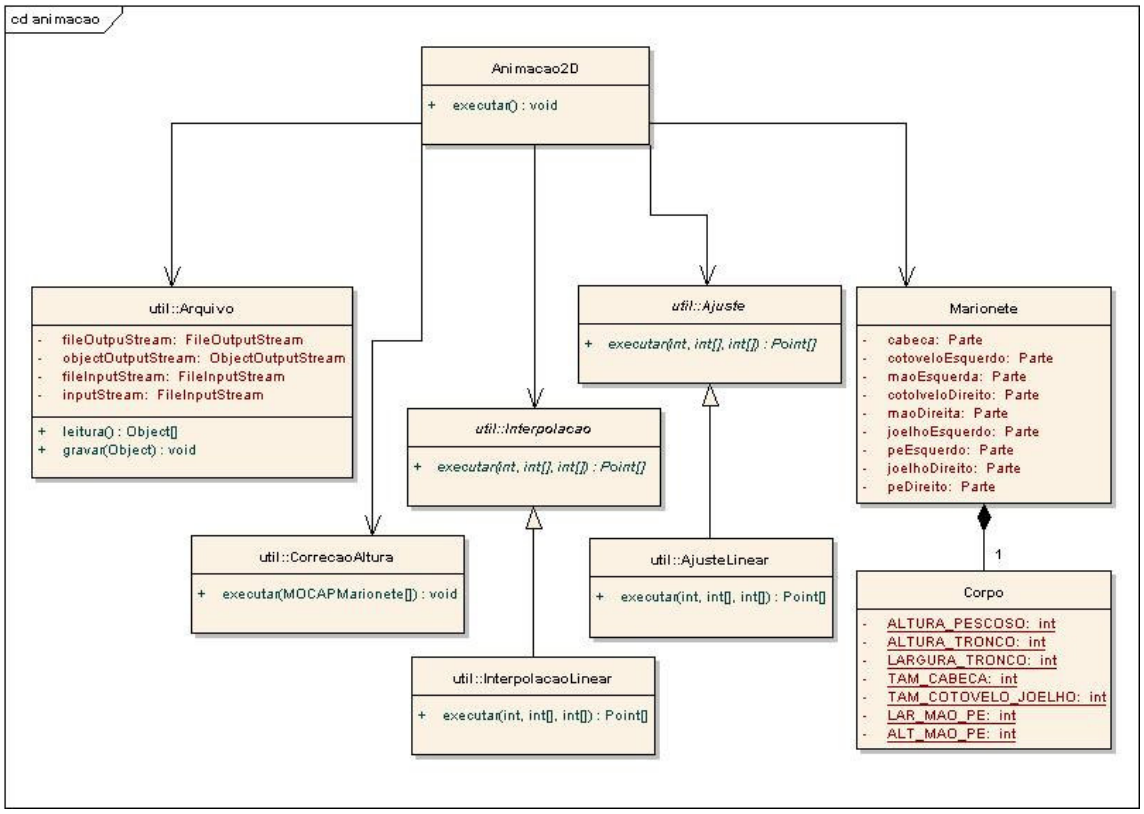


Figura 4.2 – Diagrama de classes

As classes responsáveis pelas rotinas de interface com usuário não foram abordadas no diagrama.

A classe `MOCAPPlayer` é responsável por abrir reproduzir e fechar o arquivo de vídeo. As classes `RemocaoFundo`, `AnaliseSilhueta`, `SegmentaçãoSilhueta`, `IdentificacaoPartesCorpo`, `Interpolacao e Ajuste` são abstratas para permitir a implementação de mais métodos para resolver o mesmo problema. A seguir é descrito o funcionamento de cada uma delas:

a) para `RemocaoFundo` é passado como parâmetro o conteúdo do *buffer* do quadro que está sendo analisado e o resultado é uma matriz numérica com dimensões idênticas às do vídeo e que contém a classificação dos *pixels* como “fundo da cena” (valor igual a zero) ou “em movimento” (valor igual a um);

b) para `AnaliseSilhueta` é passado como parâmetro a matriz resultante de `RemocaoFundo`. O resultado é um objeto da classe `java.awt.Polygon`, que representa a silhueta da figura humana identificada;

c) para `SegmentacaoSilhueta` é passado como parâmetro o polígono resultante de `AnaliseSilhueta` e o resultado é um objeto da classe `java.awt.Polygon` que representa a silhueta humana com os pontos candidatos a representação as principais partes do corpo;
d) para a classe `IdentificacaoPartesCorpo` é passado como parâmetro o polígono resultante de `SegmentacaoSilhueta` e o resultado é uma lista com objetos da classe `Parte` que representa a localização 2D das partes identificadas da figura humana;
e) para a classe `Interpolacao` é passado como parâmetro um conjunto de pontos, e a quantidade de pontos que se deseja realizar a interpolação. Como retorno é enviada uma lista de pontos interpolados;
f) para a classe `Ajuste` é passado como parâmetro um conjunto de pontos. O retorno dessa classe é um conjunto de pontos ajustados.

As classes `MOCAPPlayer, MOCAPCodec`, responsáveis por realizar o tratamento no arquivo de vídeo, e as classes `RemocaoFundo` e `AnaliseSilhueta` foram especificadas e implementadas por Fernandes (2002).

4.3 IMPLEMENTAÇÃO

O protótipo do software desenvolvido aborda a captura e identificação das principais partes do corpo do ator a partir de um arquivo de vídeo. Foi utilizada a idéia desenvolvida por Haritaoglu, Harwood e Davis (1998), discutida no capítulo 3, para a segmentação das imagens e identificação das principais partes do corpo humano. É gerada uma animação com os movimentos obtidos, em uma marionete virtual e salvo um arquivo contendo a localização 2D dos dados relacionados a marionete. A partir deste arquivo, é feito um tratamento sobre os dados e a gerada uma animação mais suave. Para a construção do protótipo foi utilizada a linguagem de programação Java (SUN, 2004) e o ambiente de desenvolvimento Eclipse (ECLIPSE, 2004).

Na sessão 4.3.1 aborda a extração, identificação e animação das informações sobre as principais partes do corpo humano, na sessão 4.3.2 é tratada a correção dos dados encontrados que foram armazenados em um arquivo e a animação com os dados corrigidos.

4.3.1 Tratamento dos dados obtidos do arquivo de vídeo

Nesta sessão será abordada a segmentação da imagem, identificação das partes do corpo do ator e animação da marionete virtual.

4.3.1.1 Segmentação

A segmentação de imagens procura extrair as informações importantes em um conjunto de *pixel*s, agrupando-os por regiões homogêneas e descartando as informações desnecessárias. As informações relevantes são utilizadas para uma análise mais detalhada cujo objetivo é reconhecer, representar ou classificar os objetos de interesse contidos na imagem.

No protótipo, a etapa de segmentação é usada para estimar o posicionamento 2D das principais partes do corpo do ator, utilizando geometria computacional para obter as informações que estão contidas em um conjunto de *pixels*/pontos que caracterizam o movimento da silhueta humana em cada *frame* do arquivo de vídeo.

Foi utilizado o algoritmo para obtenção do fecho convexo proposto por Graham, pois entre os algoritmos pesquisados foi o de menor custo computacional, está disponível em Cormen (2000) e foi apresentado na sessão 2.1.3. A sua utilização tem o objetivo de achar os pontos extremos da silhueta, levando em consideração que esses pontos têm uma grande probabilidade de corresponderem às principais partes do corpo (cabeça, mãos e pés). Na fig. 4.3 é mostrada uma imagem capturada do protótipo, onde o primeiro quadro corresponde ao ator em movimento. O segundo quadro é o resultado da etapa de remoção de fundo aplicada na imagem do ator em movimento. O terceiro quadro corresponde ao fecho convexo obtido com a aplicação do algoritmo proposto por Graham, na silhueta detectada exposto no quadro 2.

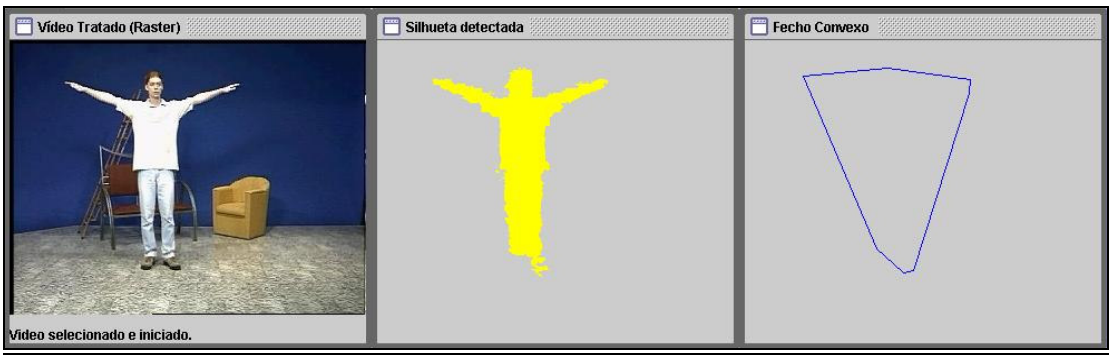


Figura 4.3 – Fecho convexo da silhueta detectada

Com a aplicação do algoritmo para a obtenção do fecho convexo, a silhueta perde os detalhes contidos no seu contorno, ficando apenas com informações sobre os pontos mais externos. Em alguns casos, as informações sobre as partes primárias do corpo humano estão no contorno da silhueta como é mostrada na fig. 4.4. A informação sobre as mãos do ator estão sendo perdidas.

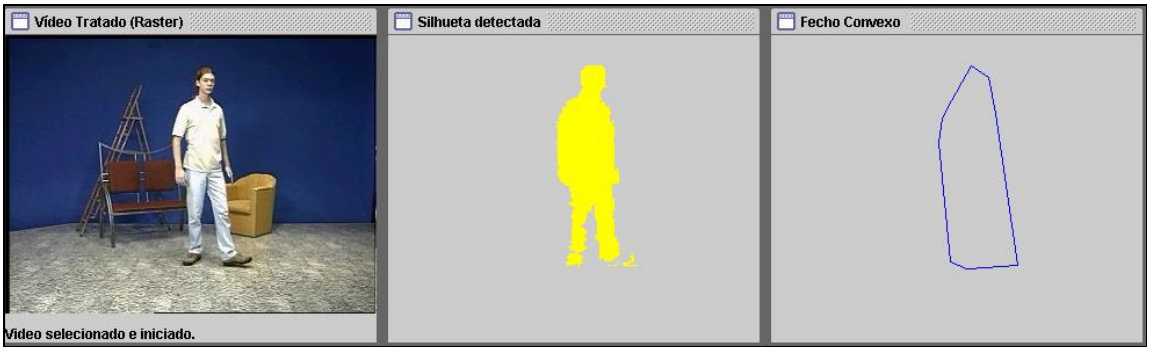


Figura 4.4 – Fecho convexo da silhueta detectada

Para contornar esse problema, foi detectado dentro do conjunto de pontos que representam a silhueta do ator o fecho convexo e fecho côncavo, ou seja, o conjunto de pontos que tem a máxima e mínima curvatura.

Para extração da curvatura mínima da silhueta detectada, são criados novos polígonos, a partir da junção do fecho convexo e da silhueta detectada na etapa de remoção de fundo.

Os polígonos são criados a partir de dois vértices que fazem parte do fecho convexo, e do sub-conjunto de pontos que estão delimitados pelos vértices que fazem parte da silhueta do ator. Conforme ilustra a fig. 4.5.

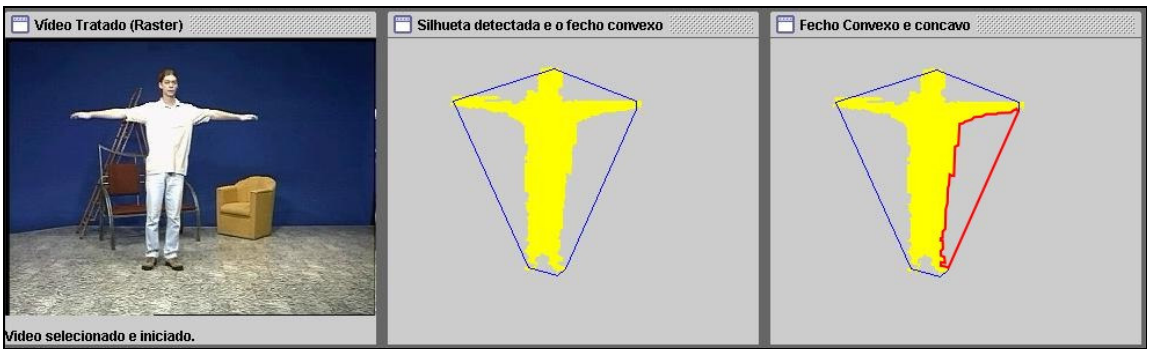


Figura 4.5 – Polígono para obtenção do fecho côncavo

No novo polígono é aplicado o algoritmo de fecho convexo, o retorno do algoritmo corresponde aos fechos côncavo e convexo de uma parte do conjunto de pontos correspondentes à silhueta do ator. Este procedimento deve ser repetido para cada par de vértices do fecho convexo da silhueta. O resultado obtido é mostrado na fig 4.6.

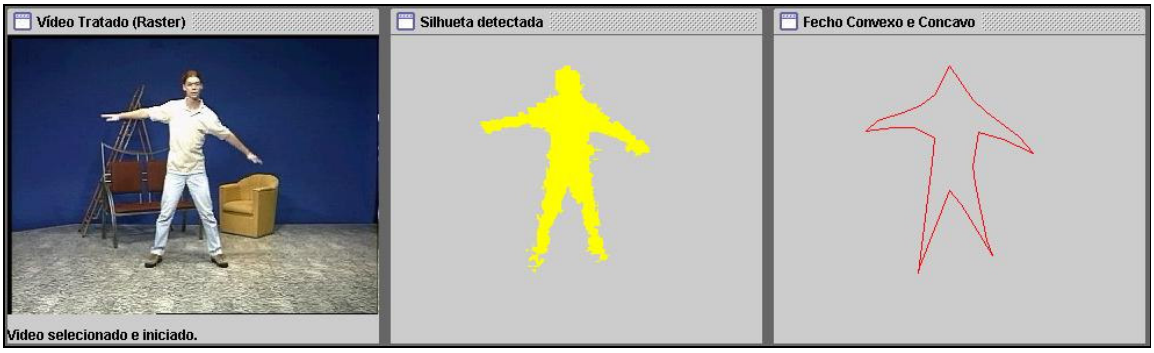


Figura 4.6 – Fecho convexo e côncavo

Na implementação do protótipo a classe `SegmentacaoSilhueta` é uma classe abstrata que tem as classes `SegmentacaoSilhuetaPartePrimaria` e `SegmentacaoSilhuetaParteSecundaria` como filhas. Estas classes são responsáveis pela obtenção das partes primárias e secundárias que fazem parte da silhueta detectada.

A classe `SegmentacaoSilhuetaPartePrimaria` implementa o algoritmo de fecho convexo de um polígono que é passado como parâmetro. A classe `SegmentacaoSilhuetaParteSecundaria` implementa a obtenção do fecho côncavo do polígono que é passado como parâmetro.

4.3.1.2 Identificação das partes do corpo

Na sessão anterior foi descrita uma maneira de filtrar os pontos para a estimação da localização 2D das principais partes do corpo. Nesta sessão será descrito a maneira utilizada para nomear os pontos obtidos na sessão anterior.

Para a identificação das principais partes do corpo foi explorada a topologia do corpo humano, suas restrições de limites e posicionamento entre os membros. O corpo humano tem uma topologia bem definida e uma ordem de localização dos membros que é invariável para uma determinada postura. Isto diminui a dificuldade da localização 2D das partes do corpo, já que não são utilizadas marcações especiais para a sua identificação.

Baseado na idéia de proporções do corpo descritas na sessão 2.2.2, Leonardo da Vinci divide o corpo humano de forma simétrica estabelecendo proporções de tamanho entre os membros que compõem o corpo. A estimativa de identificação das partes do corpo é feita impondo limites de localização e tamanho dos membros, mapeando a proporção do corpo com os pontos encontrados.

Como no protótipo a identificação das partes é feita de forma automática, ou seja, não existe nenhuma marcação especial sobre o corpo do ator e não há interação humana para inicializar variáveis ou obter alguma informação antes do processamento, a altura do ator é estimada como sendo a diferença entre o ponto mais alto e o ponto mais baixo, ou seja, entre o ponto que tem maior e o menor y. O ponto mais alto no conjunto de pontos recebido é considerado a cabeça do ator.

A partir da cabeça são definidos raios mínimo e máximo para os membros dos quais se deseja descobrir a localização, é verificado qual o ponto é mais adequado para corresponder àquela parte. A fig. 4.7, 4.8, 4.9 e 4.10 mostram o raio mínimo e o raio máximo adotados para identificação das mãos do ator.

Figura 4.7 – Identificação mãos ao longo da silhueta

Figura 4.8 – Identificação mãos levantando

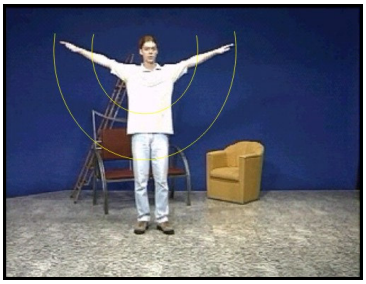


Figura 4.9 – Identificação mãos no alto

Figura 4.10 – Limites para localização das mãos

No protótipo, a classe `IdentificacaoPartesCorpo` é uma classe abstrata. A classe `IdentificacaoPartesCorpoEmPe` herda da classe `IdentificacaosParteCorpo`. As regras para a localização 2D das partes do corpo do ator que foram implementadas correspondem a um ator que está com a postura ereta e de frente para câmera. Para cada postura que o ator pode assumir deve ser implementada uma classe correspondente àquela postura, com os limites de localização de cada um dos membros do corpo do ator.

O resultado da execução da classe `IdentificacaoPartesCorpo` é uma lista de objetos referentes as partes do corpo que foram encontradas com base no polígono passado como parâmetro. Os objetos correspondem à classe `Parte`, que contém o identificador da parte localizada e sua coordenada 2D. Nesta classe, também são armazenadas as distâncias entre os membros do corpo humano, usadas para classificar os pontos encontrados como sendo ou não uma parte principal do corpo humano.

4.3.1.3 Criação da marionete animação

A classe `Marionete` é responsável pela criação da marionete virtual para reprodução dos movimentos 2D do ator. Esta classe recebe como parâmetro um conjunto de objetos da

classe `Parte` e reproduz os movimentos do ator com a localização 2D de cada parte. A fig. 4.11 ilustra a marionete criada, em amarelo se encontra a figura humana detectada.

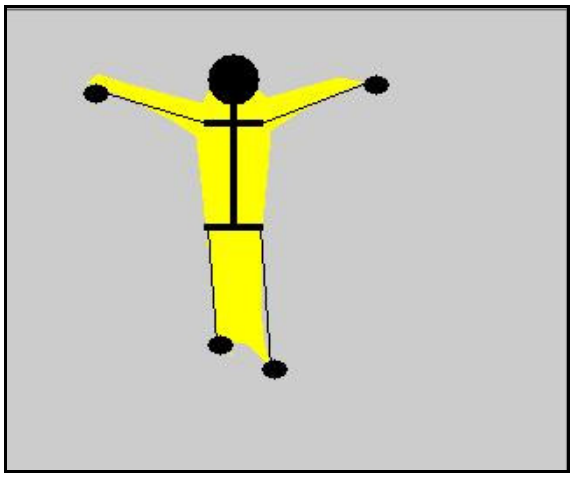


Figura 4.11 – Imagem marionete virtual

A marionete virtual é composta por uma estrutura fixa (pescoço, ombros, tronco e quadril) e por estruturas móveis (cabeça, mãos e pés). A fig. 4.12 destaca a estrutura fixa da marionete e a fig.4.13 destaca as estruturas móveis.

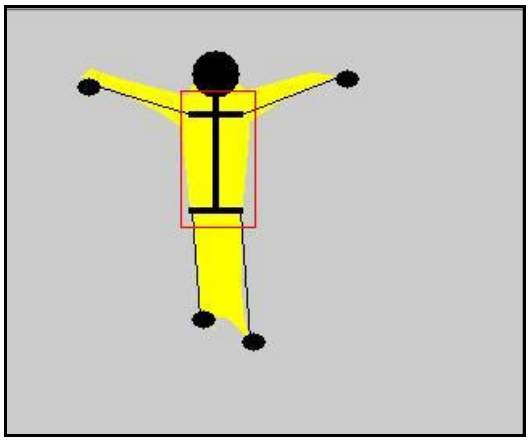


Figura 4.12 – Estrutura fixa da marionete

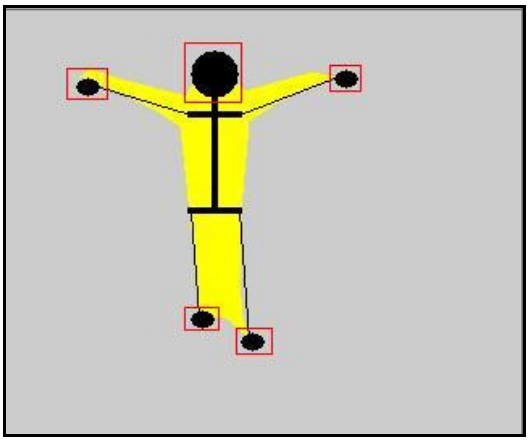


Figura 4.13 – Estrutura móvel da marionete

A estrutura fixa corresponde a um objeto do tipo `Corpo` que contém as partes do corpo com tamanho e proporções fixas. Esta estrutura é rotacionada de acordo com ângulo formado pela reta que tem inicio na cintura até o centro da cabeça. As partes móveis são objetos do tipo `Parte` que são criados na classe `IdentificacaoPartesCorpo` e são agregadas à marionete. A fig. 4.14 apresenta o diagrama de classes referente à criação da marionete virtual.

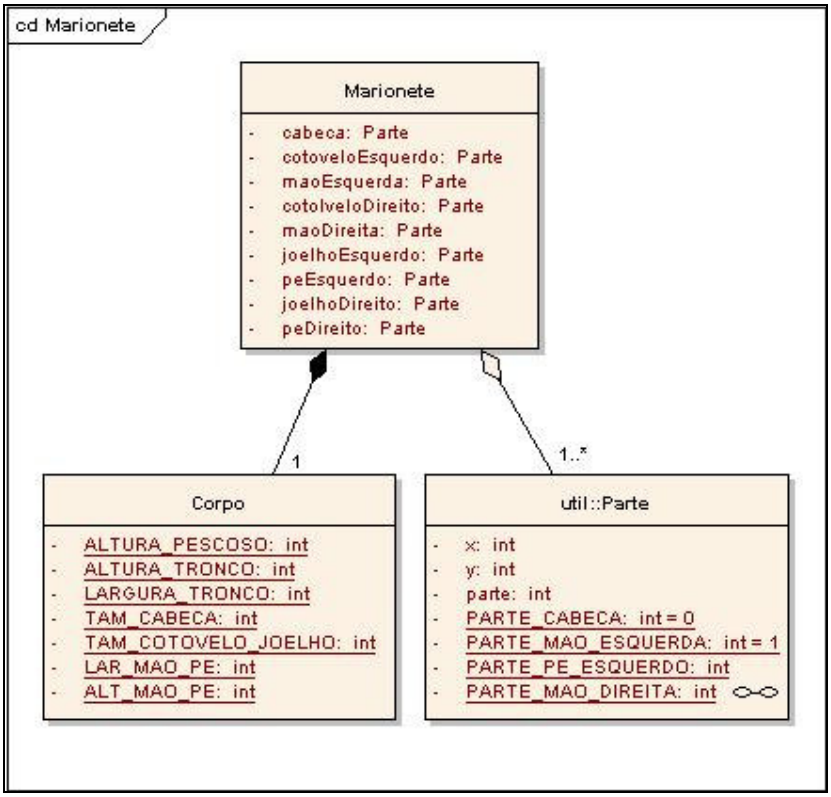


Figura 4.14 – Diagrama de classes referente à marionete virtual

A última etapa da primeira parte da implementação consiste na geração de um arquivo contendo as localizações das partes do corpo para cada *frame* do arquivo de vídeo.

A classe `Arquivo` tem a função de criar um arquivo para armazenamento dos pontos que correspondem à estimação da localização das principais partes do corpo do ator contidos no objeto da classe `Marionete`. Depois que a marionete é criada e animada, ela é salva no arquivo. Essa classe também tem a incumbência de abrir, ler e fechar o arquivo.

4.3.2 Tratamento dos dados obtidos da marionete

Nesta sessão serão relatadas as soluções implementadas para que a marionete virtual tenha uma movimentação mais suave. Na sessão 4.3.2.1 é descrita a forma de correção da altura, na sessão 4.3.2.2 a interpolação entre os dados, na sessão 4.3.2.3 o ajuste na movimentação e na 4.3.2.4 a animação da marionete.

4.3.2.1 Correção da altura do ator

Como o ator não possui nenhuma marcação especial em seu corpo e o protótipo não possui nenhum tipo de inicialização para identificação de parâmetros. A altura do ator é estimada como sendo a ponto mais alto da silhueta encontrada. Esse ponto é considerado cabeça do ator. No entanto essa afirmação não é verdadeira em alguns momentos no decorrer do vídeo o ator levanta as mãos acima da cabeça. Neste caso o protótipo entende como se a mão fosse a cabeça e a partir disso tenta achar uma relação de proporção entre os outros membros, fazendo assim com que as outras partes do corpo seja identificadas erroneamente. A fig. 4.15 mostra uma imagem onde o ator levanta a mão acima da cabeça, mostra também a identificação errada da marionete.

Figura 4.15 – Estimação da cabeça

Esse comportamento se dá de maneira brusca, no instante em que a altura da mão ultrapassa a altura da cabeça, o posicionamento da cabeça passa a ser o posicionamento onde era a mão e o posicionamento da mão é calculado baseado nas proporções do corpo levando em consideração a nova posição da cabeça. Na fig. 4.16 mostra uma seqüência de imagens retiradas do protótipo sem a correção da cabeça do ator, demonstrando esse comportamento.

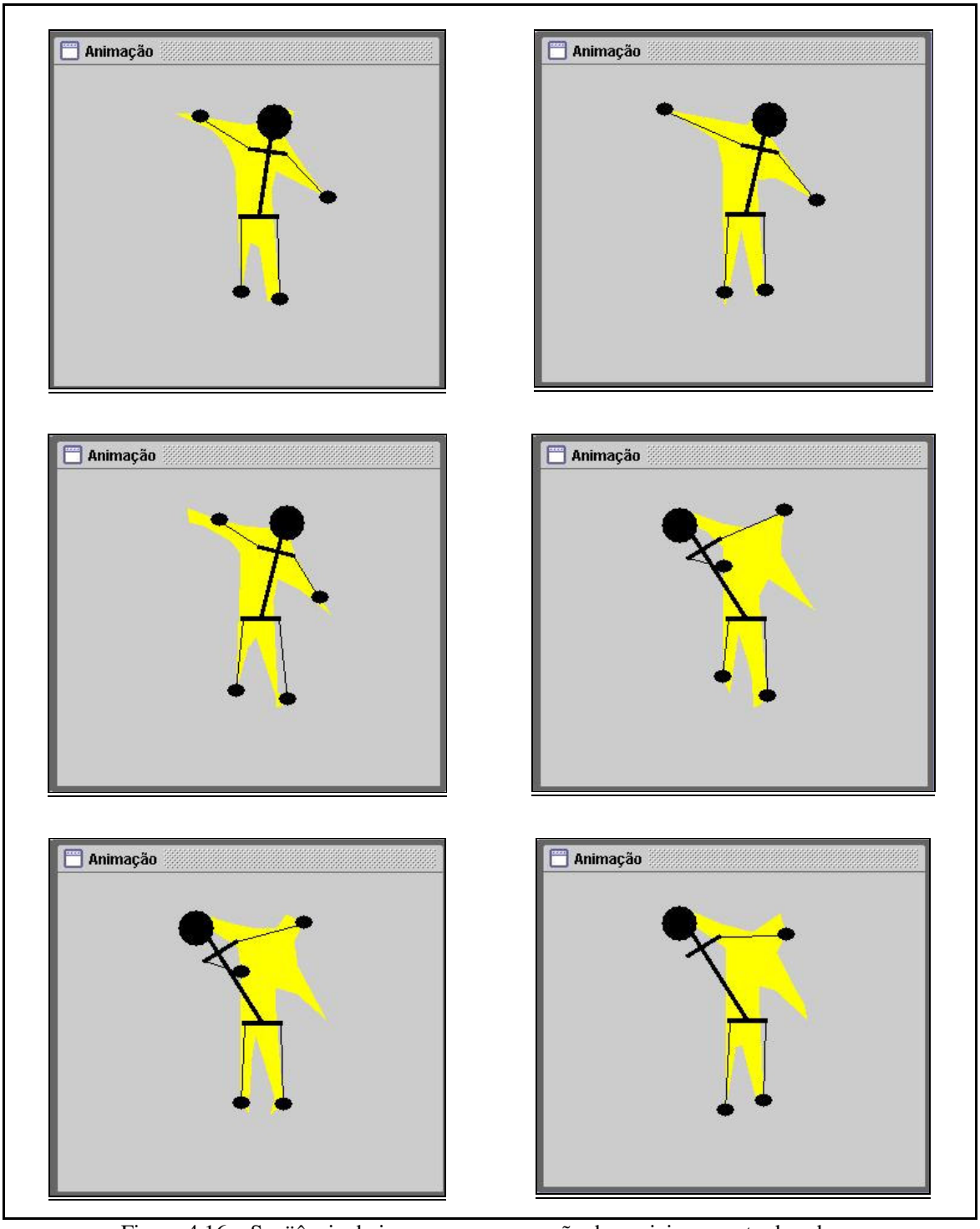


Figura 4.16 – Seqüência de imagens sem correção do posicionamento da cabeça

Para solucionar este problema foram analisadas as coordenadas correspondentes ao posicionamento da cabeça do ator, nos dois últimos *frames* analisados. Se a variação for brusca, o posicionamento das partes que compõem a marionete é estimado novamente, levando em consideração o ponto mais próximo ao ponto que foi considerado a cabeça do ator

no frame anterior. Na fig 4.17 é mostrada a mesma seqüência de *frames* da fig. 4.16, mas fazendo a correção quando houver uma variação brusca do posicionamento da cabeça.

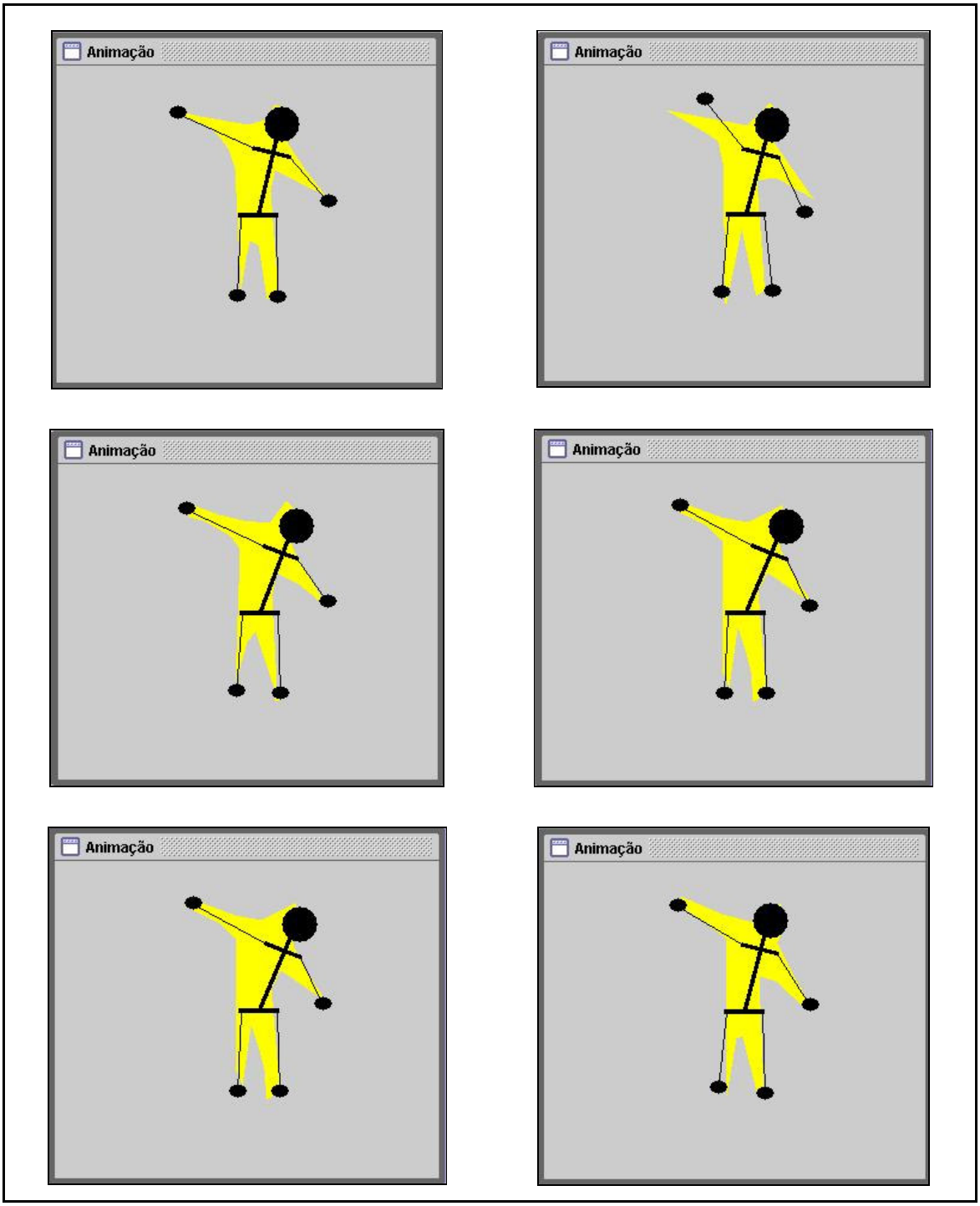


Figura 4.17 – Seqüência de imagens com correção do posicionamento da cabeça

Na Fig. 4.18 é mostrada a imagem do ator real, na fig. 4.19 a imagem da estimação das partes do corpo do ator sem a correção da altura e na fig. 4.20 a estimação das partes com a estimação da altura do ator.

Figura 4.18 – Imagem do ator

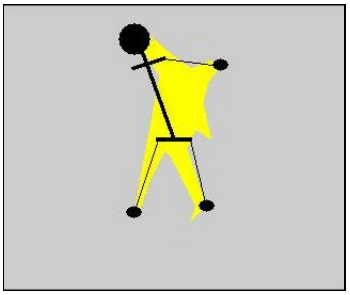


Figura 4.19 – Sem correção do posicionamento da cabeça

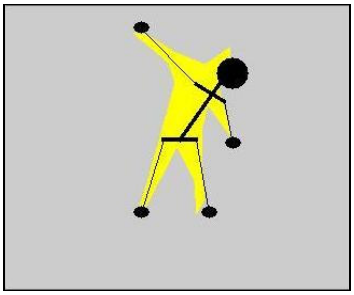


Figura 4.20 – Correção do posicionamento da cabeça

A classe `CorrecaoAltura` é responsável pela correção do posicionamento da cabeça do ator.

4.3.2.2 Interpolação

Devido o ator estar em constante movimento, e os dados serem recebidos de apenas uma câmera, algumas partes do corpo são omitidas durante alguns *frames* de acordo com a postura em que o ator se encontra. A fig. 4.21 mostra um *frame* onde a mão esquerda é omitida.

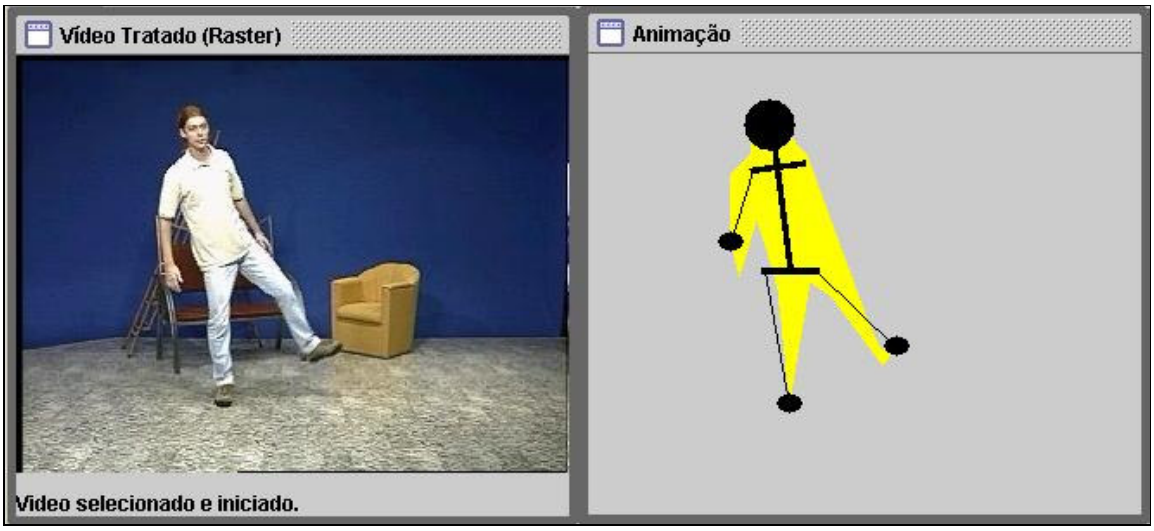


Figura 4.21 – *Frames* onde foi omitida a mão esquerda do ator.

Foi utilizada interpolação para estimar os valores referentes aos *frames* que foram omitidos. O resultado é a estimação da parte de acordo com *frames* mais próximos que foram

encontrados durante a etapa de identificação das partes do corpo. Na animação têm-se a impressão de continuidade no movimento do ator.

Como a função que deu origem ao movimento do ator real não é conhecida os valores para os *frames* onde as partes não foram identificadas são estimados através de uma interpolação linear baseada nos pontos anterior e posterior ao(s) *frame*(s) na qual à parte do corpo não foi encontrada. Através da resolução do sistema linear das equações que são mostradas a seguir obtém-se os valores de a_0 e do a_1.

$$y_i = a_0 + a_1 \cdot x_i$$

$$y_j = a_0 + a_1 \cdot x_j$$

Deve-se estimar um valor para a coordenada que esteja entre os valores x_i e x_j e a equação determina o valor para a abscissa através dos valores obtidos por a_0 e a_1.

A seguir é exposto na fig. 4.22 um gráfico onde uma suposta parte do corpo não é encontrada durante dois *frames* do arquivo. Na fig. 4.23 é mostrado o gráfico resultante da etapa de interpolação, com a estimação dos pontos que foram omitidos, baseados nos pontos que estão mais próximos.

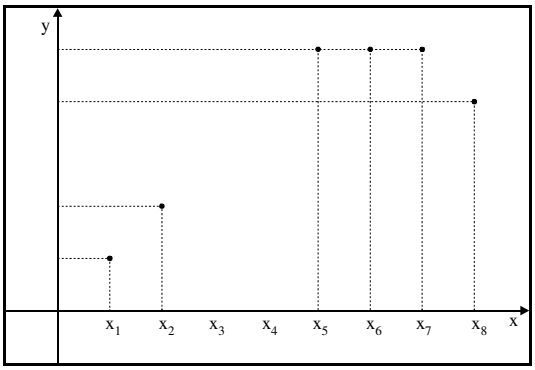


Figura 4.22 – Gráfico "obtido"

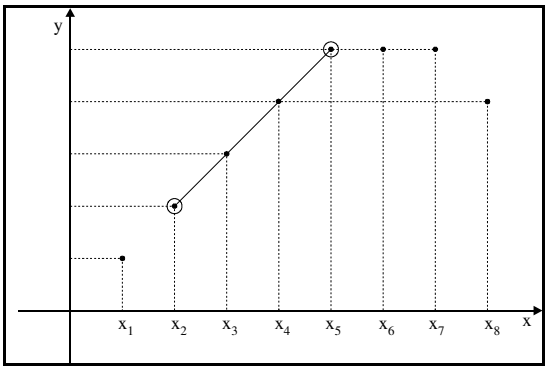


Figura 4.23 – Gráfico gerado

Na fig. 4.24 mostra o resultado da interpolação do *frame* no qual a mão esquerda tinha sido omitida.

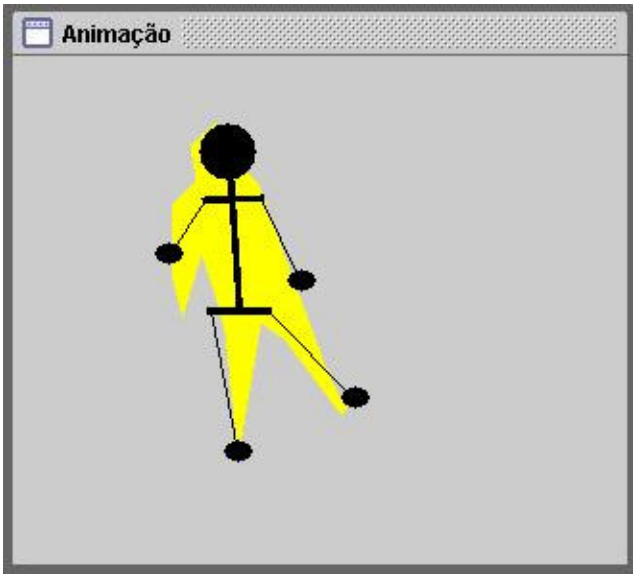


Figura 4.24 – Interpolação da mão esquerda

A classe abstrata `Interpolacao` e a classe filha `InterpolacaoLinear` é usada para esse propósito.

4.3.2.3 Ajuste

A técnica usada para encontrar os *pixels* que correspondem à silhueta do ator foi a de remoção do fundo da cena. Para isso o sistema de captura de movimento óptico tem um tempo de aprendizagem no inicio do vídeo. Isso é necessário para que o sistema reconheça o fundo da cena e posteriormente, quando o ator entra em cena, identifique quais *pixels* fazem parte da silhueta do ator e quais os que fazem parte do cenário (subtração do fundo da cena).

Esta técnica não é muito precisa. Existem pequenas variações na localização das coordenadas 2D do contorno do corpo do ator devido a alguns ruídos, sombras. Durante a fase de animação da marionete notou-se que esse comportamento tem um efeito visual ruim, com uma ligeira trepidação, fazendo necessária uma etapa de ajuste entre os pontos encontrados para suavizar a animação da marionete. As classes `Ajuste` e `AjusteLinear` são responsáveis por tal funcionalidade. A classe `Ajuste` é uma classe abstrata que tem a classe `AjusteLinear` como classe filha, nesta foi implementada a técnica de ajuste linear entre os pontos passados.

O ajuste é uma técnica de aproximação de funções que se aplica a um conjunto de dados experimentais $\{(x_1,y_1), (x_2,y_2), ..., (x_n,y_n)\}$ e deseja-se obter a lei $y = f(x)$ estabelecendo

uma relação entre a variável independente x e a variável dependente y da forma que melhor se adapte aos pontos dados. O ajustamento traduz um comportamento médio.

Segundo Cláudio (2000) para o ajuste linear, sabe-se que existe uma relação linear entre os pontos e procura-se estabelecer uma reta que melhor se ajuste aos pontos dados, conforme mostra a fig. 4.25.

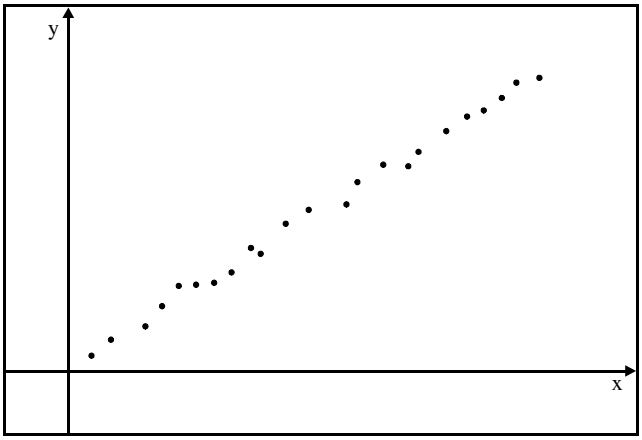


Figura 4.25 – Ajuste linear de um conjunto de pontos

Dada a equação da reta onde $reta = a + b.x$, a fórmula do ajuste linear é dada por $\hat{y} = \beta_0 + \beta_1.x$ onde $\hat{y}$ é o ajuste, β_0 eqüivale ao parâmetro linear e o β_1 ao parâmetro angular e são obtidos pelas seguintes equações respectivamente:

$$\beta_0 = \frac{\sum y_i - \sum x_i \cdot \beta_1}{n}$$

$$\beta_1 = \frac{n \cdot \sum x_i y_i - \sum x_i \cdot \sum y_i}{n \cdot \sum \mathrm{x_i}^2 - \left(\sum \mathrm{x_i}\right)^2}$$

Na implementação a quantidade de *frames* a serem ajustados foi feita de forma empírica. O ajuste a cada 15 *frames* dos pontos identificados para cada uma das partes que compõem o corpo do ator virtual foi a que obteve o melhor resultado. Este método amenizou a instabilidade da movimentação do personagem tornando a movimentação mais suave.

4.3.2.4 Animação

Nesta etapa, é feita a animação da marionete com os dados atualizados. A altura da marionete corrigida, as partes que não foram identificadas em alguns *frames* foram

interpolados com os pontos mais próximos e houve um ajuste nos dados de forma que a marionete tenha uma movimentação mais suave. A animação acontece da mesma forma que foi descrita na sessão 4.3.1.3.

5 CONCLUSÕES

O presente trabalho discutiu o desenvolvimento de um protótipo de software para animação de personagens humanos virtuais com captura ótica de movimento sem utilização de marcações especiais sobre o corpo do ator. No protótipo, foi implementada a estimação das principais partes do corpo humano a partir de um arquivo de vídeo que contém uma pessoa em movimento. O ator deve está em pé, preferencialmente de frente para câmera. A principal dificuldade da estimação da posição das partes do corpo do ator se dá devido à grande variedade de posturas que o corpo humano pode assumir, e à falta de marcação no corpo do ator, que possa servir como sinal na identificação.

Em um dos arquivos de vídeo usado como entrada de dados notou-se a ausência da estimação do posicionamento da mão esquerda do ator em 15% dos *frames*, o posicionamento da mão direita e o pé esquerdo em aproximadamente 10%, e o posicionamento do pé direito em 5% dos *frames* que compõe o arquivo. Nos *frames* onde não foi possível estimar a localização, foram utilizados métodos de interpolação para gerar uma posição intermediária, utilizando os *frames* anterior e posterior ao *frame* no qual a posição não foi encontrada. Esta etapa muda o efeito visual na animação gerada, pois em alguns *frames* uma das partes desaparecia da animação.

Na etapa de animação da marionete virtual foi observada uma instabilidade nos movimentos do personagem, este problema foi amenizado através do uso de métodos de ajustes de função.

O protótipo conseguiu realizar o processamento da localização das partes do corpo em tempo real, gerando uma animação com os pontos encontrados e sem nenhum tipo de tratamento. O resultado obtido com esta animação foi usado para extração de dados estatísticos. A reconstrução dos movimentos do ator não foi feita em tempo real, visto que os métodos utilizados para diminuir a instabilidade da animação são executados depois de ter todos os pontos estimados. Para o desenvolvimento do protótipo foi utilizado um microcomputador AMD Atlhon[TM]XP1800+ com 1.54Ghz de freqüência e 512Mb de memória RAM.

5.1 EXTENSÕES

Para trabalhos futuros sugere-se:

a) realizar a identificação das várias posturas que o corpo humano pode assumir. A partir da postura identificada, podem ser implementadas classes para estimação do posicionamento 2D das partes do corpo do ator, levando em consideração a ordem de localização das principais partes do corpo e os limites que a postura oferece. Detalhes podem ser obtidos em Haritaoglu, Harwood, Davis (1998);
b) dar continuidade ao desenvolvimento do protótipo para a localização de mais partes do corpo do ator e reconstrução dos movimentos de forma mais realística;
c) criação e animação de uma marionete 3D. Para isso é necessário ter mais de uma câmera obtendo a movimentação do ator de ângulos diferentes. A partir dos pontos obtidos, fazer cálculo de triangulação quadrática para a obtenção da coordenada 3D daquele ponto.

REFERÊNCIAS BIBLIOGRÁFICAS

AMBLER, S. W. **Análise e projeto orientados a objeto**: um guia para o desenvolvimento de aplicações orientadas a objeto. 4. ed. Rio de Janeiro: Infobook, 1997.

BERG, Mark et al. **Computational geometry:** algorithms and applications. Berlin: Spring, 2000.

CARREIRA, Eduardo. **Os escritos de Leonardo da Vinci sobre a arte da pintura**. São Paulo: Ed. Universidade de Brasília, 2000.

CLÁUDIO, Dalcidio Morais, MARINS, Jussara Maria. **Cálculo numérico computacional**: Teoria e prática. São Paulo: Ed. Atlas, 2000.

CORMEN, Thomas H. et al. **Algoritmos**: teoria e pratica. Rio de Janeiro: Campus, 2000.

ECLIPSE. Version 3.0, 2004. Disponível em: <http://www.eclipse.org>. Acesso em: 13 nov. 2004.

FERNANDES, Leandro A. F. **Protótipo de um sistema óptico de captura do movimento humano, sem a utilização de marcações especiais**. 2002. 115 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

FERNANDES, Marco Antonio. **Notas da produção:** final fantasy. SãoPaulo, [2004?]. Disponível em: <http://www.webcine.com.br/notaspro/npfinalf.htm>. Acesso em: 10 abr. 2004.

FIGUEIREDO, Luiz Henrique de, CARVALHO, Paulo Cézar Pinto. **Introdução à geometria computacional**. Rio de Janeiro: IMPA, 1991.

HARITAOGLU, Ismail; HARWOOD, David; DAVIS, Larry S. **Ghost**: a human body part labeling system using silhouettes. Germantown, [1998?]. Disponível em: <http://www.umiacs.umd.edu/users/hismail/Ghost_outline.htm>. Acesso em: 02 abr. 2004.

HORPRASERT, Thanatat et al. **Real-time 3D motion capture**. San Fransisco, 2000. Disponível em: <http://www.umiacs.umd.edu/users/hismail/Publications/PUIPaper.pdf>. Acesso em: 31 mar. 2004.

MAESTRI, George. **Animação [digital] de personagens**. São Paulo: Quark do Brasil Ltda, 1996.

PREPARATA, Franco P.; SHAMOS, Michael I. **Computacional geometry**: an introduction, New York: Springer, 1988.

SILVA, Fernando Wagner Serpa Vieira da; CAVALCANTI, Paulo Roma**.** Animações complexas em tempo real utilizando movimentos capturados. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING (SIBGRAPI), 1., 1996, Rio de Janeiro. **Anais eletrônicos...** Rio de Janeiro: 1996. Disponível em: <http://mirror.impa.br/sibgrapi96/programa/com.html>. Acesso em: 01 abr. 2004.

SILVA, Fernando Wagner Serpa Vieira da.**Um sistema de animação baseado em movimento capturado**. 1998. 101 f. Tese (Mestrado em Computação Gráfica) - Laboratório de Computação Gráfica COPPE/Sistemas, UFRG, Rio de Janeiro. Disponível em: <http://www.visgraf.impa.br/Projects/mcapture/publ/thesis-letter.pdf >. Acesso em: 21 nov. 2004.

SUN MICROSYSTEMS INC. **Java™ media framework API guide**. Mountain View: Sun, 2004.