CLEITON LUIZ ROCHA TEODORO

ANIMAÇÃO DE OBJETOS ARTICULADOS: PROPOSTAS PARA O ACTIONBUILDER

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Orientador:
Prof. Bruno de Oliveira Schneider

LAVRAS
MINAS GERAIS - BRASIL
2009

CLEITON LUIZ ROCHA TEODORO

ANIMAÇÃO DE OBJETOS ARTICULADOS: PROPOSTAS PARA O ACTIONBUILDER

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 26 de Novembrro de 2009

Ana Paula Piovesan Melchiori

Cristiano Leite de Castro

Prof. Bruno de Oliveira Schneider
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL
2009

Dedico este trabalho à minha família e a todos aqueles que estiveram comigo neste período da minha vida.

Agradecimentos

À Deus e aos meus pais Sinval e Marta por todo o apoio e paciência durante esta longa jornada, e aos meus irmãos Danielle e Celton. Ao professor Bruno de Oliveira Schneider por compartilhar seu conhecimento comigo.

Sumário

1 Introdução **1**

2 Referencial Teórico **3**

2.1 Um resumo da história da animação por computador 3

2.2 Representação de Objetos Articulados 5

2.3 Métodos de Animação 7

2.3.1 Animação por Quadro-Chave 7

2.3.2 Métodos de Interpolação 8

2.3.3 Animação Comportamental 11

2.3.4 Captura de Movimento 12

2.4 Softwares para Animação por Computador 13

2.4.1 Blender 13

2.4.2 Art of Illusion 14

2.5 V-ART 14

3 Metodologia **17**

3.1 Animação no Blender 17

3.2 Animação no Art of Illusion 20

4 Proposta de Recursos para o ActionBuilder **22**

4.1 Mudar o Controle de Tempo 22

4.2 Utilização de quadros-chave 23

4.3 Utilização de uma Interface Gráfica 24

4.4 Mudar o tipo de interpolação 26

4.5 Editor de Curvas 27

4.6 Sistema de Cinemática Inversa 27

5 Discussões Finais **30**

A Nova DTD para a animação **35**

B Algoritmo para executar a animação **36**

Lista de Figuras

2.1 Exemplo de uma árvore representando o corpo humano 7

2.2 Animação com os quadro-chave destacados em cinza 8

2.3 Interpolação linear (a) e interpolação senoidal (b) 9

2.4 Curva de Hermite (a) e Curva de Bézier (b) 10

2.5 Modelo humano e dados capturados 13

3.1 Preparando personagem para animação 18

3.2 NLA Editor e quadros-chave destacados 19

3.3 Editor de curvas 20

3.4 Trilhas e linha de tempo 21

3.5 Editor de Curvas do Art of Illusion 21

4.1 Classe Frame 23

4.2 Classe Animação 24

4.3 Classe ActionBuilder 25

4.4 Classe MainFrame 25

4.5 Classes GlCanvas e EditArea 26

4.6 Cinemática Direta e Cinemática Inversa 28

4.7 Calculando o ângulo entre os dois vetores 29

Resumo

Este trabalho é um estudo dos aspectos que formam a base para um sistema de animação de objetos articulados. Foram estudadas as estruturas para representar esses objetos, os métodos de animação, com ênfase na animação por quadro-chave e também os métodos de interpolação aplicados a esse tipo de animação, além da pesquisa das técnicas utilizadas para implementar um sistema de cinemática inversa. Foram estudados também editores de animação focando a influência das interfaces no processo de animação. Com base nesses estudos foram feitas propostas de melhorias no ActionBuilder, um software para criação de animações para o framework V-ART.

Palavras-Chave: Computação Gráfica, Animação, V-ART, Objetos Articulados

Abstract

This work is a study of the elements that form the base for an animation system for articulated objects. It is a review of the data structures used for representing articulated objects, the animation methods (with emphasis on key-frame animation), the interpolation methods for in-betweening and the techniques for inverse kinematics. Animation software has been put to use, in order to observe the influence of their interfaces on the animation creation process. These reviews provided a base for proposing enhancements to ActionBuilder, an animation software for creating animation using the V-ART framework.

Keywords: Computer Graphics, Animation, V-ART, Articulated Figures

1. Introdução

A animação de um modo geral pode ser definida como o processo de geração de uma série de quadros representando um conjunto de objetos no qual cada quadro constitui uma alteração do quadro anterior.

A forma através da qual uma animação é feita pode ser dividida em animação convencional, feita sem o auxílio do computador, e a animação por computador, aquela em que os quadros são gerados pelo computador.

A animação convencional é baseada em uma técnica quadro-a-quadro onde animadores definem e ilustram os quadros principais da seqüência que será animada, após isto, animadores assistentes têm a tarefa de desenhar os quadros intermediários inferindo mentalmente as ações entre os quadros.

O computador começou a ser utilizado na animação primeiramente como ferramenta auxiliar nos processos de composição, edição e colorização de filmes criados através do processo de animação convencional.

Com a evolução dos computadores e das pesquisas, o computador passou a ser utilizado para fazer as animações. Muitos dos editores de animação desenvolvidos utilizam o método de animação por quadro-chave citada anteriormente. A quantidade de informação que um animador tem que controlar nestes editores é muito grande.

Por isso é importante que os editores de animação facilitem ao animador o tratamento de toda essa informação. O *framework* V-ART (V-ART)[1] tem um editor de animação o ActionBuilder, ambos serão abordados com mais detalhe no Capítulo 2, que não oferece algumas dessas funcionalidades se comparado aos editores convencionais.

O objetivo deste trabalho é propor recursos para um editor de animações que facilite o processo de modificação de uma animação.

[1] http://www.codeplex.com/vart

Para desenvolver este editor será necessário especificar um sistema de animação de objetos articulados, focando nas estruturas de dados necessárias à criação, representação e modificação da animação. A especificação será embasada em soluções encontradas na literatura e nas soluções já implementadas no V-ART. Para isso foram utilizados os editores de animação Blender[2] e Art of Illusion[3] para conhecer as dificuldades que um animador pode encontrar ao realizar uma animação.

Este trabalho está organizado da seguinte maneira, no Capítulo 2 são discutidos os conceitos básicos de animação e como eles contribuem para gerar a animação, no Capítulo 3 é apresentada a metodologia, no Capítulo 4 são apresentadas e discutidas as propostas para o editor e, finalmente, no Capítulo 5 a conclusão e a discussão.

[2]http://www.blender.org
[3]http://www.artofillusion.org

2. Referencial Teórico

Nesta seção será fornecido um histórico da animação por computador, depois, serão descritas as formas de se representar objetos articulados no computador e, finalmente, a descrição de técnicas utilizadas pela animação por computador.

2.1 Um resumo da história da animação por computador

As primeiras animações feitas por computador no fim dos anos 60 e início dos anos 70 foram produzidas por uma mistura de pesquisadores em laboratórios nas universidades e artistas visionários (PARENT, 2002). Naquela época, monitores com *frame buffe*r estavam começando a ser desenvolvidos e a saída digital para televisão ainda estava em estágio experimental. Os monitores em uso eram *storage tubes*[1] e *vector displays*. Entretanto, devido à dificuldade para modificar as imagens, os *storage tubes* eram usados principalmente para desenhar modelos estáticos. *Vector displa*ys usam uma lista de exibição de instruções para desenhar linhas e arcos que um processador interno usa para desenhar repetidamente uma imagem que de outra forma desapareceria rapidamente da tela. *Vector displays* podem desenhar imagens em movimento pela mudança da lista de exibição entre as atualizações.

As primeiras pesquisas em animação por computador ocorreram no MIT em 1963 quando Ivan Sutherland desenvolveu um sistema interativo para satisfação de restrições, o usuário podia construir um conjunto de linhas especificando restrições entre vários elementos gráficos. Se um dos elementos gráficos se movia, o sistema calculava a reação dos outros elementos para esta mudança satisfazer as restrições especificadas. Pela manipulação interativa de um desses elementos gráficos o usuário podia produzir movimentos complexos no resto da montagem.

[1] *storage tube* é um tipo de tubo de raios catódicos (CRT) cuja tela é capaz de reter imagens durante um longo tempo, e no qual o feixe proveniente do canhão de elétrons pode ser movimentado livremente na superfície da tela.

Mais tarde, na Universidade de Utah, Sutherland ajudou David Evans a estabelecer o primeiro programa de pesquisa significativo em computação gráfica e animação.

No início dos anos 70, a animação por computador em laboratórios de universidades se tornou mais comum. A computação gráfica, assim como a animação por computador, recebeu um importante impulso através do financiamento governamental na Universidade de Utah. Como resultado foram produzidos muitos trabalhos pioneiros em animação: uma mão e uma face animadas por Ed Catmull (*Hand/Face*, 1972) (CATMULL, 1972); uma figura humana que falava e caminhava, feita por Bary Wessler (*Not just reality*, 1973); e uma face que falava feita por Fred Parke (*Talking Face*, 1974) (PARKE, 1972). Apesar das imagens primitivas pelos padrões atuais, estas apresentações foram demonstrações bem a frente de seu tempo.

Em meados dos anos 70, Norm Badler na Universidade da Pensilvânia conduziu investigações para colocar em pose uma figura humana. Ele desenvolveu um sistema de restrições para mover a figura de uma posição para outra. Ele continuou esta pesquisa e estabeleceu o Centro para Modelagem Humana e Simulação em na Universidade da Pensilvânia. *Jack* é um software desenvolvido neste centro que suporta o posicionamento e a animação de figuras humanas em um mundo virtual.

No fim dos anos 70, o Instituto de Tecnologia de Nova Iorque embarcou em um ambicioso projeto de produzir um filme completamente feito no computador, chamado The Works. O projeto nunca foi completado, mas trechos mostrados em muitas conferências da SIGGRAPH (*Special Interest Group on GRAPHics and Interactive Techniques*) mostraram renderização de alta qualidade, figuras articuladas e objetos interagindo.

Em 1974, a primeira animação por computador indicada para o Oscar, *Hunger*, foi produzida por René Jodoin. Esta obra usou um sistema de 2½D[2] que dependia de técnicas de modificação de forma de objetos e interpolação de linhas.

[2]um sistema $2^1/_2$D ou pseudo-3D é um termo usado para se referir a gráficos 2D que simulam a aparência de gráficos 3D

Nos anos 80 se viu um movimento mais sério de empresários em direção da animação comercial. O hardware avançou significativamente com a introdução do IBM PC no inicio dos anos 80. Surgia a *Silicon Graphics*. Simuladores de vôo baseados em tecnologia digital se tornaram realidade.

Algumas animações que produzidas pela LucasFilms e Pixar que ganharam prêmios:

- The Adventures of André and Wally B. (1984);
- Luxo Jr. (1986) – indicada para o Oscar;
- Red's Dream (1987);
- TinToy (1988) – primeira animação feita no computador a ganhar um Oscar;
- Knick Knack (1989);
- Geri's Game (1999) – ganhador do Oscar;

Existe uma polêmica sobre qual foi o primeiro filme produzido totalmente por computador, os dois candidatos são o filme americano *Toy Story* e o brasileiro Cassiopéia (SUPPIA, 2006). A diferença de lançamento entre os dois filmes é de meses, mas este não é o principal motivo da polêmica. Enquanto em Toy Story alguns personagens mais complexos foram feitos primeiramente em modelos de argila e depois digitalizados, em Cassiopéia todos os personagens foram totalmente criados no computador resultando num longa metragem feito de maneira 100% digital.

2.2 Representação de Objetos Articulados

Encontrar formas adequadas para representação de objetos articulados é de importância fundamental para animação de pessoas e animais. Uma estratégia muito utilizada é representar estes objetos através de hierarquias.

Em (GOMES; VELHO, 2003) uma hierarquia é definida da seguinte forma:

> ”no sentido mais amplo uma hierarquia é um grafo onde os vértices são objetos gráficos. Temos portanto um conjunto de pares(O_i, O_j) de objetos que constituem as arestas. Quando dois objetos constituem uma aresta dizemos que eles possuem uma relação de vínculo na hierarquia. Os objetos definidos por uma hierarquia podem ser classificados em objetos compostos ou objetos articulados.”

Nos objetos compostos não existe movimento relativo entre os componentes da hierarquia. Os objetos articulados são formados por partes rígidas conectadas por juntas que permitem a movimentação relativa entre elas.

Existem dois tipos de juntas, as juntas de revolução e as juntas esféricas. Nas juntas de revolução os objetos estão ligados através de um eixo em torno do qual eles podem girar, já as juntas esféricas são representadas por uma esfera que é livre para executar qualquer movimento de rotação. As juntas de revolução possuem apenas um grau de liberdade, enquanto que as juntas esféricas possuem três (GOMES;VELHO, 2003). Os graus de liberdade descrevem a quantidade de movimentos independentes de translação e rotação que uma junta pode assumir no espaço. A animação de um objeto articulado se dá através da aplicação de movimentos de translação e rotação em suas juntas ao longo do tempo.

Uma forma comum para representar objetos articulados é a estrutura em árvore. O nó mais alto da árvore é o nó raiz cuja posição é conhecida no sistema de coordenadas global e a posição de todos os outros nós da hierarquia está localizada em relação à posição do nó raiz. Se entre dois nós existe uma ligação, aquele que está mais alto na hierarquia é chamado de nó pai e aquele mais baixo é chamado de nó filho. A Figura 2.1 é um exemplo de objeto articulado representado por uma estrutura em árvore.

As hierarquias são mais adequadas para a representação de objetos articulados porque facilitam as transformações de movimento e posicionamento. Isto se deve às regras de hereditariedade das transformações na hierarquia. Uma transformação aplicada à um objeto de nível mais alto na hierarquia é aplicada também aos seus filhos mantendo a integridade da estrutura. Estruturas de dados hierárquicas aplicadas a animação são discutidas em (BOULIC; RENAULT,1991). Um padrão

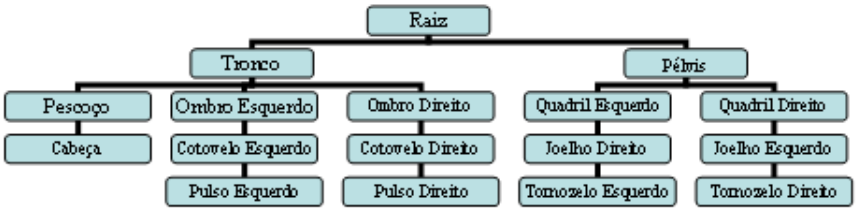

Figura 2.1: Exemplo de uma árvore representando o corpo humano

para a representação de objetos articulados é proposto por (H-ANIM, 1999).

2.3 Métodos de Animação

2.3.1 Animação por Quadro-Chave

A animação por quadro-chave (em inglês *key-frame*) é uma das técnicas mais utilizadas na animação por computador. Esta técnica é baseada nos procedimentos de animação feitas à mão, na qual um animador define e desenha os quadros principais da sequência que será animada e depois animadores assistentes desenham os quadros intermediários inferindo mentalmente as ações entres os quadros principais.

Os primeiros sistemas de animação por computador utilizavam a animação por quadro-chave (BURTNYK;WEIN, 1976), (CATMULL,1978). No computador o animador define as poses principais e o computador é encarregado de calcular os quadros intermediários utilizando algum método de interpolação. A Figura 2.2 mostra uma sequência definida através da animação por quadro-chave.

A animação por quadro-chave requer muito esforço manual do animador. Para assegurar uma boa interpolação é necessário um alto nível de detalhes. Outra desvantagem da animação por quadro-chave é a dificuldade para modificar uma sequência de ações definidas entre dois quadros-chave. Por isso existe a necessi-

Figura 2.2: Animação com os quadro-chave destacados em cinza

dade de especificar quadros mais próximos uns dos outros para situações onde se deseja um movimento mais complexo (NEDEL, 2000).

2.3.2 Métodos de Interpolação

O fundamento da animação por quadro-chave é a interpolação de valores. Nesta seção serão descritos os principais métodos de interpolação usados na animação por computador.

Ao escolher um método de interpolação para animação por computador devem ser levados em conta 3 fatores: a complexidade da função interpoladora, continuidade e controle global versus controle local.

A complexidade da função interpoladora se traduz em eficiência computacional, quanto mais simples a função mais rápido o computador irá realizar os cálculos. Funções polinomiais de terceiro grau são as mais utilizadas porque oferecem flexibilidade suficiente sem muita complexidade computacional.

Na verdade, na animação por computador são utilizadas curvas compostas de segmentos menores de polinomiais cúbicas como as curvas de Hermite e de Bézier. Usar polinômiais de ordem maior que três não trazem maiores benefícios e aumentam o custo computacional para calculá-las (PARENT, 2002).

A continuidade se refere à quantas derivadas da função são contínuas, em animação isso se traduz em uma animação em que os movimentos são mais suaves

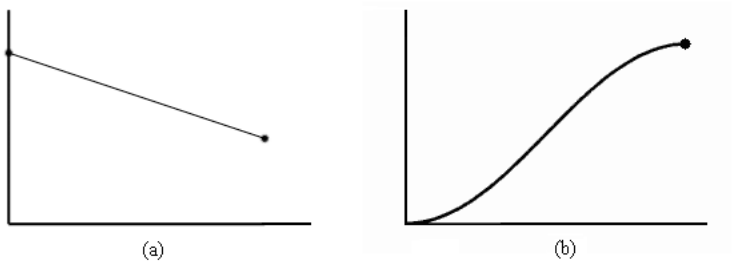

Figura 2.3: Interpolação linear (a) e interpolação senoidal (b)

e realistas.

Quando se trabalha com curvas o animador pode desejar reposicionar apenas alguns pontos que controlam a forma da curva para modificar apenas uma parte dela. É dito que uma curva que tem esta propriedade possui controle local. Ao contrário, quando uma alteração em um único ponto da curva provoca uma modificação em toda a curva, é dito que esta curva possui controle global.

A interpolação linear é o tipo de interpolação mais simples que existe. Matematicamente, a interpolação linear gera um segmento de reta entre dois pontos, ver Figura 2.3(a).

Dados dois pontos a e b a equação utilizada para encontrar um ponto intermediário entre eles é:

$$a(1-t)+bt, 0 \leq t \leq 1 \tag{2.1}$$

Uma das desvantagens da interpolação linear é que a animação fica com um aspecto pouco realista, além de causar distorções em movimentos que envolvem rotações (KOCHANEK; BARTELS, 1984). Este método também pode causar problemas quando usado para interpolar posições no espaço (FOLEY *et al.*, 1990).

A interpolação de curvas evita alguns problemas da interpolação linear. Na animação por computador os tipos de interpolação de curvas mais utilizados são a interpolação senoidal, a interpolação Hermite e a interpolação Bézier.

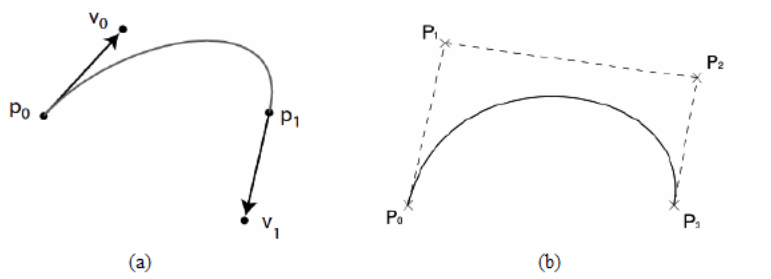

Figura 2.4: Curva de Hermite (a) e Curva de Bézier (b)

A interpolação senoidal resulta em uma curva mais suave se comparada à interpolação linear. A função interpoladora é definida por:

$$c(t) = \frac{sin(t\pi - \frac{\pi}{2}) + 1}{2}, 0 \leq t \leq 1 \tag{2.2}$$

Esta função é usada para implementar um conceito conhecido como *Ease-in/Ease-out*, muito usado por permitir maior controle da velocidade da animação (PARENT, 2002). A curva resultante de uma interpolação senoidal é mostrada na Figura 2.3 (b).

Uma curva de Hermite é uma curva de terceiro grau definida pelos dois pontos que serão interpolados e pelas tangentes à estes pontos, como mostra a Figura 2.4.

Os pontos p_0 e p_1 são, respectivamente, os pontos inicial e final da curva. O vetor v_0 indica como a curva deixa o ponto inicial, ou seja, direção e velocidade e o vetor v_1 indica como a curva encontra o ponto final.

A função interpoladora é dada por:

$$c(t) = (1 - 3t^2 + 2t^3)p_0 + (t - 2t^2 + t^3)v_0 + (-t^2 + t^3)v_1 + (3t^2 - 2t^3)p_1, 0 \leq t \leq 1 \tag{2.3}$$

Uma curva de Bézier é semelhante à uma curva de Hermite, com a di-

ferença de que na última, os vetores tangentes são calculados indiretamente pela definição de dois pontos auxiliares que não pertencem a curva ver Figura 2.4.

A função interpoladora é:

$$c(t) = (1-t)^3 p_0 + 3(1-t)^2 t p_1 + 3(1-t)t^2 p_2 + t^3 p_3, 0 \leq t \leq 1 \qquad (2.4)$$

Onde p_0 e p_3 são os pontos a serem interpolados e p_1e p_2 são os pontos de controle.

2.3.3 Animação Comportamental

O movimento de personagens não é simplesmente uma questão de mecânica. O movimento baseado em mecânica é muito formal porque não leva em conta o estado emocional do personagem. É pouco realista pensar que somente as características físicas de duas pessoas realizando as mesmas ações as fazem diferentes para um observador. O comportamento e a personalidade dos seres humanos também são uma causas essenciais das diferenças de movimento entre os humanos.

A animação comportamental corresponde à modelagem do comportamento dos personagens e das interações emocionais complexas entre eles. Em uma implementação ideal da animação comportamental, é quase sempre impossível executar a mesma cena de maneira exatamente igual (THALMANN; THALMANN, 1993). Na animação comportamental os personagens são projetados para serem agentes autônomos, inteligentes o suficiente para animar a si mesmos.

São duas as principais abordagens para a animação comportamental, o modelo comportamental e o modelo cognitivo. O modelo comportamental descreve como o personagem deve agir ao estado atual do ambiente que o cerca. Essa abordagem é usada em (REYNOLDS, 1987) para simular bandos de pássaros e cardumes de peixes. O comportamento de cada pássaro ou peixe em relação aos seus vizinhos é governado por um conjunto de regras e o movimento do grupo é

orientado por um vetor de direção global. É impossível fazer uma animação deste tipo usando técnicas tradicionais como a animação por quadro-chave.

O modelo cognitivo descreve o processo de pensamento do personagem permitindo que ele decida sobre possíveis ações (FUNGE; TU; TERZEPOULOS, 1999), conseqüentemente esse modelo é considerado mais poderoso do que o modelo comportamental, mas requer mais capacidade de processamento. (MILLAR; HANNA; KEALY, 1999) descrevem as técnicas utilizadas na animação comportamental.

2.3.4 Captura de Movimento

Outra técnica utilizada na animação por computador é a captura de movimento. Esta técnica envolve a captura da posição e orientação de um objeto real no espaço físico e a gravação dessa informação de forma a ser utilizada em um modelo computacional (NEDEL, 2000). A Figura 2.5 ilustra os dados extraídos de um modelo em um sistema de captura de movimento.

A captura de movimento inicialmente foi desenvolvida para aplicação na medicina, por exemplo, para estudo do caminhar de pessoas com problemas de locomoção. No entanto a qualidade do movimento tornou esta técnica atraente para a indústria cinematográfica.

As vantagens de usar a captura de movimento são claras. Ela produz imediatamente dados para todos os graus de liberdade com um alto nível de detalhe. Se o movimento é incomum ou se muito realismo é necessário, pode ser difícil para o animador criar o movimento utilizando animação por quadro-chave. Como resultado, é mais fácil simplesmente capturar as ações de um ator executando os movimentos e mapear os dados para um modelo no computador (PULLEN, 2002).

Entre as desvantagens estão as diferenças geométricas entre os modelos humanos e os objetos modelados no computador, as seções de captura de movimento ainda são caras e trabalhosas, logo é melhor não repeti-las. Ainda assim é difícil saber exatamente quais são os movimentos desejados antes de uma sessão.

Um assunto que é objeto de pesquisa é a captura de movimento através de

Figura 2.5: Modelo humano e dados capturados

vídeo (BREGLER; MALIK, 1997), (HERDA; URTASUN; FUA, 2004). Estudos também estão sendo feitos no sentido de adaptar dados já capturados à modelos de vários tamanhos ou para combinar movimentos (GLEICHER, 1998). Isto pode resultar no desenvolvimento de bibliotecas de movimentos digitalizados que podem ser adaptados e combinados para produzir movimento em qualquer modelo.

2.4 Softwares para Animação por Computador

Nesta seção serão descritos alguns programas utilizados para modelagem e animação por computador, estes programas também podem ser chamados de sistemas de animação por computador.

2.4.1 Blender

O Blender é um sistema de animação por computador multiplataforma de código aberto desenvolvido pela Blender Foundation (BLENDER, 2009) e possui suporte para modelagem e renderização.

O Blender tem recursos similares aos de outros programas proprietários

que incluem ferramentas de simulação tais como dinâmica de corpos rígidos e dinâmica de fluídos.

A princípio a interface do Blender é confusa para iniciantes, no entanto, o blender possui uma comunidade grande e atuante em todo o mundo inclusive no Brasil e conta com uma grande quantidade de tutoriais e manuais para o iniciante.

2.4.2 Art of Illusion

Art of Illusion (ART, 2009) é um programa usado para modelagem 3D e renderização de imagens estáticas ou animações. O objetivo deste programa é ser um sistema de animação com uma interface melhor se comparada à de outros programas do gênero. Apesar da interface simples, este programa contém muitas características encontradas em softwares gráficos comerciais.

Ao contrário do Blender, o Art of Illusion possui uma interface mais simples de usar, o que torna mais o fácil o processo para se fazer uma animação. Essa simplicidade também é conseqüência do menor número de recursos do Art of Illusion. A comunidade em torno do Art of Illusion não é tão grande quanto a do Blender, mas a página do projeto do Art of Illusion possui documentação completa e tutoriais.

2.5 V-ART

O V-ART (*Virtual Articulations for Virtual Reality*) é um *framework* multiplataforma e independente de API gráfica desenvolvido em C++ para facilitar a criação de programas com ambientes 3D, em especial os que contêm humanóides. Ele se distingue de outros *frameworks* semelhantes por ser inteiramente orientado a objetos e possuir um sistema de suporte a animações, além de permitir a representação de articulações biologicamente corretas.

Fazer animações que se baseiam em movimentos humanos é uma atividade complicada visto que o corpo humano é uma estrutura complexa geralmente representada por um objeto articulado.

Para tratar a complexidade envolvendo movimentação semelhante à dos seres humanos, o V-ART foi desenvolvido pelo grupo de Computação Gráfica da UFRGS. A base para a criação do V-ART foi o trabalho de Maciel (MACIEL; NEDEL; FREITAS, 2002).

No V-ART, um corpo articulado é representado através de um arquivo XML (*eXtensible Markup Language*) que é um padrão mundial para troca de qualquer tipo de dado estruturado. Embora muito usado na troca de documentos pela Web, esse padrão foi desenvolvido para armazenar qualquer tipo de dado estruturado, tendo se mostrado ideal para dados hierarquicamente estruturados, como por exemplo o corpo humano. Em um arquivo XML, os dados são gravados em formato texto, com etiquetas de marcação para separar cada seção, atributo ou objeto.

No V-ART os graus de liberdade trabalham em um espaço normalizado, ou seja, quando um grau de liberdade é criado e seu intervalo de movimentação definido, sua configuração é um valor no intervalo entre 0 e 1, onde 0 é o limite mínimo e 1 o limite máximo.

Uma junta no V-ART pode possuir até três graus de liberdade:

- Flexão: alterar este grau de liberdade implica na alteração do ângulo entre dois membros.
- Adução: alterar este grau de liberdade produz um movimento de aproximação ou afastamento do membro em relação ao tronco.
- Torção: o movimento de um membro em torno de seu eixo longitudinal.

O ActionBuilder é uma aplicação usada para fazer animações para o V-ART. É através dele que são criados e carregados os arquivos que representam uma animação.

No V-ART as animações, também chamadas de ações, são conjuntos de posições e tempos alvo para determinados graus de liberdade. Estas ações também são descritas através de arquivos XML e trabalham com um tempo absoluto e passam tempos normalizados de acordo com a velocidade definida. Quando uma

ação é definida, o tempo total da animação é especificado em segundos. Sendo 0 o tempo inicial da ação e 1 o tempo final, cada grau de liberdade tem seus tempos determinados neste intervalo.

3. Metodologia

De acordo com Jung (2004), este trabalho pode ser classificado em relação à sua natureza como pesquisa tecnológica pois tem como objetivo modificar um *framework* já existente, no caso o V-ART. Quanto aos objetivos ela pode ser classificada como exploratória e quanto aos procedimentos como experimental.

Foi feito inicialmente um levantamento de artigos científicos clássicos e atuais referentes ao tema da animação por computador. As principais bases utilizadas para encontrar estes artigos foram a ACM SIGGRAPH (SIGGRAPH, 2009) e a IEEE Computer Graphics and Applications (CGA, 2009).

Foram utilizados também dois editores de animação semi-profissionais e de código aberto, o Blender e o Art of Illusion com o objetivo de conhecer as interfaces e recursos mais comuns à editores.

As atividades executadas foram: realização de animações em ambos os editores, realização de animações utilizando o ActionBuilder, a proposição de recursos a serem implementados no V-ART com o objetivo de melhorar a realização das animações.

Optou-se por realizar a animação de uma caminhada porque este movimento envolve praticamente todos os membros do corpo, o que resulta em uma animação com grau de dificuldade maior para fazer e para editar.

3.1 Animação no Blender

Inicialmente a tela para animar um personagem no Blender é como a exibida na Figura 3.1

Para iniciar a animação é preciso escolher o "*Pose Mode*" que é onde o esqueleto que foi colocado no personagem fica destacado e pronto para que o animador comece a definir as poses.

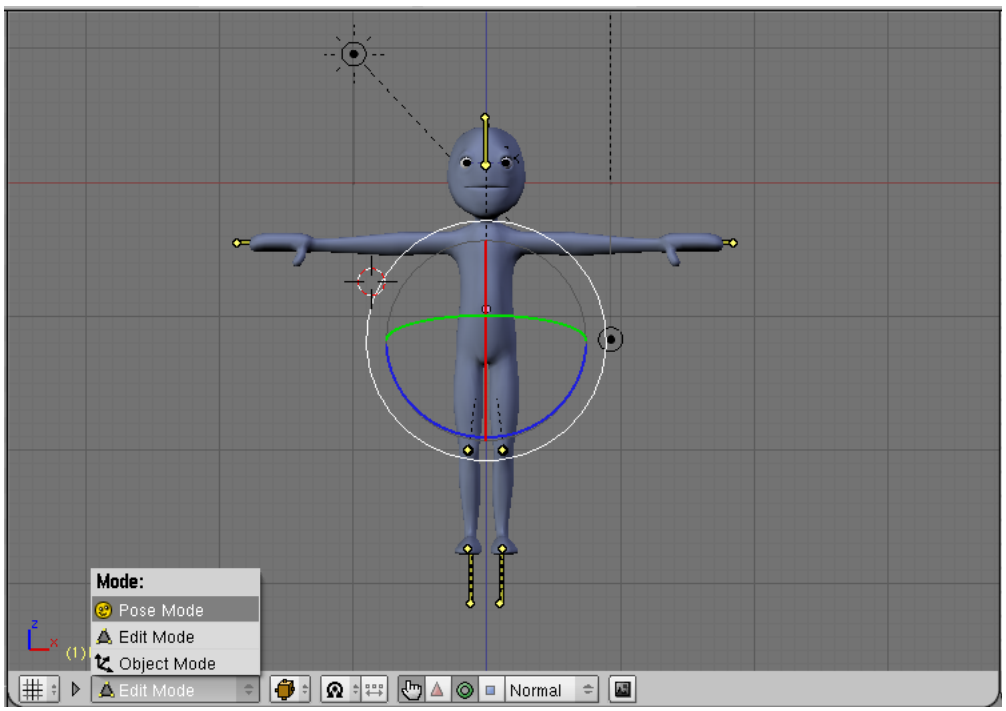

Figura 3.1: Preparando personagem para animação

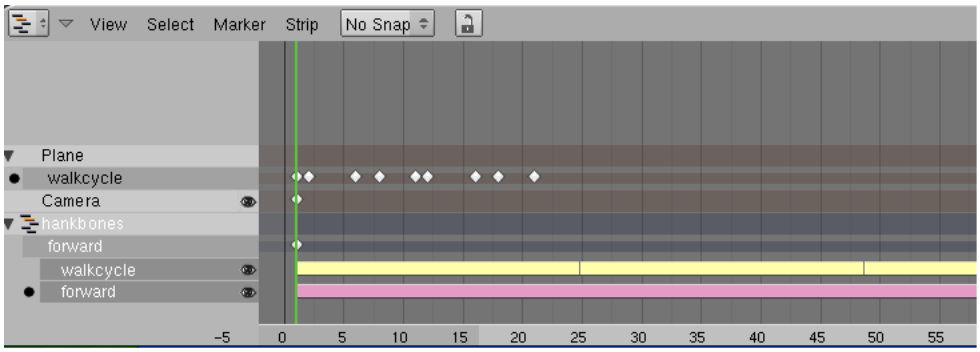

Figura 3.2: NLA Editor e quadros-chave destacados

A forma utilizada para fazer a animação no Blender foi através de *actions*. Uma action é criada através do *ActionEditor* e pode ser combinada com outras através do NLA Editor (*non-linear animation* editor, vide Figura 3.2).

Existe uma área na interface que representa os quadros da animação. É preciso posicionar a linha no quadro desejado, posicionar o personagem e adicionar a pose-chave pressionando a tecla "i". Cada quadro-chave é exibido como um losango (vide Figura 3.2).

Deve-se repetir este processo para cada uma das posições chave do personagem.

Após terminar de definir as posições chave da animação, é preciso fazer fazer correções que vão tornar a animação mais realista.

Para fazer ajustes em uma animação o Blender oferece o IPO Curve Editor que permite editar as curvas que representam a animação. O eixo Y representa o valor editado (posição, rotação) e o eixo X a linha de tempo, (vide Figura 3.3).

O uso deste editor permitiu um melhor controle sobre a variação dos movimentos entre um quadro e outro.

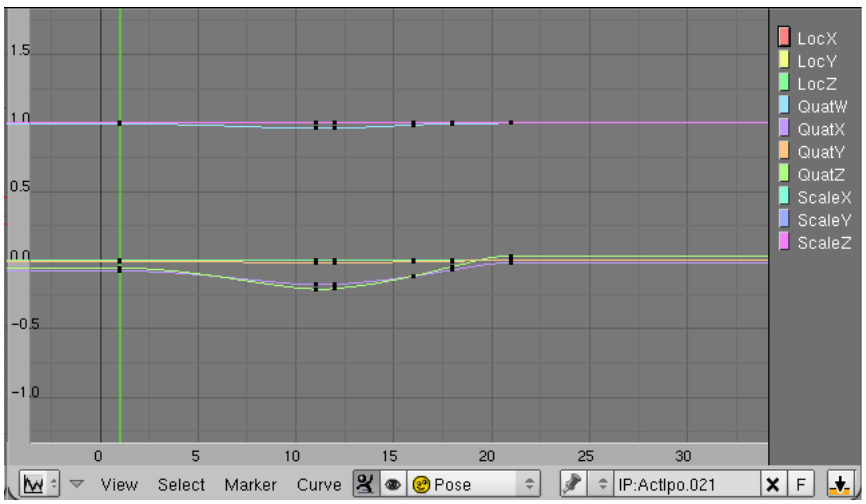

Figura 3.3: Editor de curvas

3.2 Animação no Art of Illusion

O processo de animar um personagem no Art of Illusion é semelhante ao do Blender. Ele possui uma linha de tempo onde se pode adicionar os quadro-chave, como mostra a Figura 3.4.

No Art of Illusion é possível definir movimentos independentes para o personagem, cada movimento independente pode ser definido em uma trilha diferente como mostra a Figura 3.4. Existem as trilhas para as poses, e outras três para posições dos quadros-chave e os quadros intermediários também são destacados como losangos.

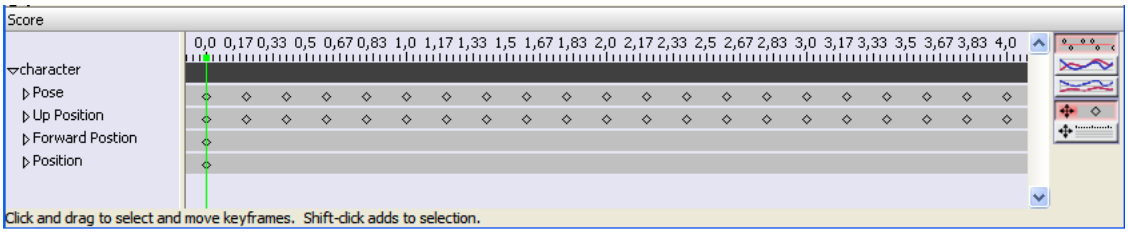

Figura 3.4: Trilhas e linha de tempo

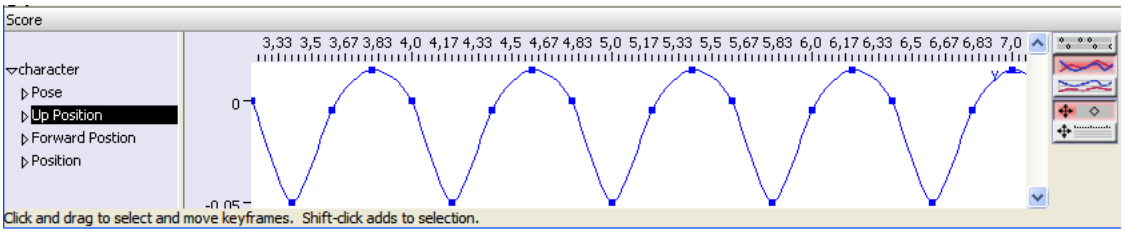

Figura 3.5: Editor de Curvas do Art of Illusion

O Art of Illusion também possui um editor de curvas com as mesmas funcionalidades do Blender. A Figura 3.5 mostra as curvas que definem os ciclo da caminhada para o personagem. Os quadrados na curva representam as posições definidas pelo animador. Elas podem ser adiantadas ou atrasadas, o valor pode ser alterado e estes quadros também pode ser removidos.

4. Proposta de Recursos para o ActionBuilder

Idealmente, o animador atuaria como um diretor que diz ao personagem (o "ator") o que ele deve fazer e como ele deve fazer. Embora muitas pesquisas estejam sendo realizadas com este objetivo a tecnologia atual ainda está longe de produzir os resultados desejados pelos animadores.

Enquanto estes resultados não são alcançados é preciso que os programas utilizados para fazer as animações ofereçam meios para que os animadores realizem e modifiquem as animações de maneira menos trabalhosa e que permita maior expressividade à animação.

O V-ART tem a capacidade de alterar animações em tempo de execução, característica que os dois editores de animação utilizados não possuem, por isto é necessário desenvolver um editor que reuna as características do V-ART e dos editores convencionais.

4.1 Mudar o Controle de Tempo

A velocidade de uma ação é um princípio importante porque ela dá significado ao movimento definindo o impacto da animação no observador. Ela pode definir o estado emocional de um personagem, por exemplo, ao variar a velocidade de uma ação estes movimentos podem indicar se o personagem está triste, nervoso ou relaxado.

O ActionBuilder determina automaticamente a duração do movimento de um grau de liberdade, isto não dá ao animador flexibilidade suficiente para fazer a animação.

Esta modificação na maneira de tratar o tempo na animação também será

```
class Frame
{
private:

    float time;
    VART::Collector<VART::Joint> pose;
}
```

Figura 4.1: Classe Frame

fundamental para outros recursos propostos neste trabalho, por exemplo, o editor curvas que será descrito seção 4.5.

Um melhor controle de tempo permitirá ao animador adiantar ou atrasar o início do movimento de um grau de liberdade.

4.2 Utilização de quadros-chave

Um quadro-chave pode ser considerado como o valor de todos os graus de liberdade de um objeto articulado em um determinado instante. Então, uma estrutura básica para representar um quadro-chave deve guardar os graus de liberdade e o tempo (vide Figura 4.1).

A variável pose irá guardar o valor de todos os graus de liberdade do objeto.

Considerando esta representação, uma animação pode ser pensada como um vetor de poses (vide Figura 4.2) que tem o número de posições igual ao número total de quadros da animação. Cada quadro-chave seria um conjunto de poses definidas pelo animador, enquanto os quadros intermediários do vetor seriam obtidos através da interpolação.

Considerando as alterações propostas será necessário modificar também os arquivos xml que representam uma animação: eles deveram agora ter tags que descrevem um quadro e seus parâmetros e também a animação com seu tempo de duração e a quantidade de quadros ver Apêndice A.

```
class Animacao
{
private:

    float tempoTotal;
    Frame animacao[qtdQuadros];
}
```

Figura 4.2: Classe Animação

4.3 Utilização de uma Interface Gráfica

Para que a realização de um animação seja facilitada é preciso que o editor disponha de uma interface intuitiva e fácil de usar. Interfaces gráficas facilitam a interação do animador com o programa além da capacidade exibir mais informações. O Blender e o Art of Illusion possuem alguns aspectos em comum em suas interfaces gráficas.

Uma interface gráfica para um editor de animação deve ter como funcionalidades mínimas:

- uma área para mostrar o objeto que está sendo animado: esta área deve permitir que o animador veja o objeto de diferentes ângulos e a possibilidade se aproximar e afastar a câmera.
- uma área para mostrar as propriedades da junta selecionada: devem ser mostrados os valores correspondentes à posição da junta selecionada, valores do graus de liberdade.
- uma área que mostre a linha de tempo da animação: os instantes em que existem posições chave definidas pelo animador devem ser destacados, o animador deve ter a possibilidade de adiantar, atrasar, acrescentar ou remover posições chave através desta área.

Para a interface do editor é sugerido o wxWidgets (WXWIDGETS), que é um *framework* multiplataforma de código aberto para desenvolvimento de interfaces gráficas.

```
#include <wx/wx.h>

class ActionBuilder : public wxApp
{
public:
    virtual bool OnInit();
};
```

Figura 4.3: Classe ActionBuilder

```
#include <wx/mdi.h>

class MainFrame : wxMDIParentFrame
{
private:

    TimeLine *linhadetempo;
    EditArea *editarea;
}
```

Figura 4.4: Classe MainFrame

O primeiro passo será converter o ActionBuilder que é uma aplicação console para uma wxApp. Uma aplicação que use o wxWidgets deve ter a seguinte estrutura (vide Figura 4.3)

Cada um dos itens listados acima deverá ser uma janela. A janela principal da aplicação será do tipo wxMDIParentFrame, o MDI (*multiple document interface*) será responsável por gerenciar as múltiplas áreas da aplicação.

O *frame* que irá conter a área de edição irá herdar a classe wxMDIChildFrame que será criada a partir do frame principal da aplicação.

O frame principal deverá conter os ponteiros para a linha de tempo e a área para edição do objeto (vide Figura 4.4) .

A área onde o objeto está sendo editado deve conter um objeto do tipo wxGLCanvas que permite a exibição de gráficos 3D nas aplicações feitas usando o wxWidgets (vide Figura 4.5).

```
#include <wx/glcanvas.h>

class GLCanvas : public wxGLCanvas
{
public:

    wxGLContext* MyContext;
    GLCanvas(wxFrame* parent,const wxSize& size, int* args);
};

#include <wx/mdi.h>
#include "glcanvas.h"

class EditArea : public wxMDIChildFrame
{
public:

    GLCanvas* glWindow;
    void resized(wxSizeEvent& event);
    EditArea(const wxString& title,const wxPoint& pos, const wxSize& size);
};
```

Figura 4.5: Classes GlCanvas e EditArea

4.4 Mudar o tipo de interpolação

Para implementar um editor de curvas semelhante aos dos editores utilizados é preciso introduzir uma nova forma de representar e interpolar as curvas no V-ART visto que ele implementa dois tipos de interpolação, a interpolação linear e a interpolação senoidal.

O método de interpolação a ser introduzido é a interpolação Hermite. A vantagem de utilizar curvas de Hermite para interpolar os valores dos graus de liberdade ao longo do tempo é que podem ser adicionados vários pontos de controle, que podem ser interpretados como novos quadros-chave, com a garantia de que a curva sempre irá passar por eles.

4.5 Editor de Curvas

Nos dois editores utilizados um recurso que facilitou a realização das animações foi a edição de curvas. Estes editores permitem que animador observe o comportamento das curvas que definem a animação ao longo do tempo com os quadros-chave destacados.

Por comportamento da curva entende-se que são os valores pelos quais a curva que representa o grau de liberdade passa ao longo do tempo da animação, a forma da curva.

Na animação estes pontos de controle serão os valores que os grau de liberdades devem assumir naquele instante.

A primeira função de um editor de curvas é mostrar o comportamento das curvas que definem a animação. Como dito no Capítulo 2, no V-ART cada junta possui até três graus de liberdade. O editor de curvas deve mostrar o comportamento de cada um desses graus de liberdade ao longo da animação no mesmo gráfico e em cores diferentes.

Devem ser destacados o tempo inicial e o tempo final de movimento de cada grau de liberdade.

Além de mostrar as curvas que representam a animação o editor de curvas deve permitir um ajuste da animação através do controle da velocidade e da alteração de valores referentes aos graus de liberdade das juntas.

4.6 Sistema de Cinemática Inversa

Tradicionalmente, na animação, a posição de um membro de um objeto articulado é determinada em função dos ângulos das juntas as quais ele está direta ou indiretamente ligado. Esta forma de obter a posição de um membro é chamada de Cinemática Direta.

Quando um animador está fazendo uma animação pode ser muito cansativo

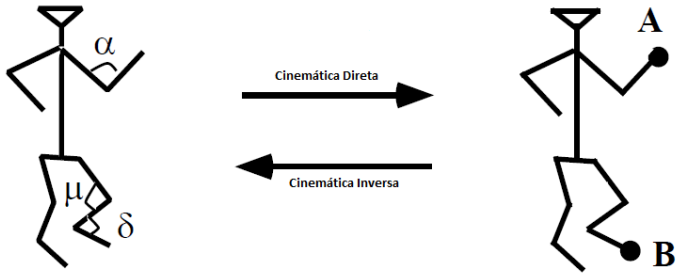

Figura 4.6: Cinemática Direta e Cinemática Inversa

ter que definir todos os ângulos das juntas para definir a posição de um membro.

Para evitar as dificuldades de ter que especificar o ângulo de todas as juntas, a Cinemática Inversa é usada. Na Cinemática Inversa o animador especifica apenas a posição dos membros extremos (mão, pés) e os ângulos das juntas internas são calculados automaticamente.

A Figura 4.6 [1] ilustra o princípio da Cinemática Direta e da Cinemática Inversa, na primeira, o animador irá definir os ângulos α,μ e δ e o computador encontrará as posições A e B e na última o usuário irá apenas posicionar os membros nas posições A e B e o computador irá encontrar os ângulos α,μ e δ.

A Cinemática Inversa é um aspecto muito importante na animação e no desenvolvimento de jogos, onde ela assegura que os personagens caminhem de maneira correta pelo cenário, por exemplo.

Existem vários métodos para resolver este problema na Cinemática Inversa, os métodos analíticos e os métodos numéricos.

Os métodos analíticos reduzem o problema através da resolução de uma série de sistemas de equações não-lineares, produzindo uma solução exata. Este método é de rápida resolução para sistemas com poucos graus de liberdade, porém, quanto mais complexos forem os objetos articulados mais difícil fica obter uma

[1] Extraída de (Thalmann; Thalmann, 1993)

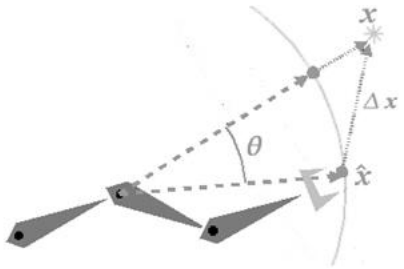

Figura 4.7: Calculando o ângulo entre os dois vetores

solução de maneira analítica.

Os métodos numéricos resolvem o problema através de várias iterações e são geralmente baseados em inversões de matrizes. Um método muito usado é a Jacobiana Inversa em que uma matriz chamada Jacobiana é usada para mapear as mudanças nas variáveis das juntas para mudanças na posição e orientação dos membros extremos.

O método proposto para cinemática inversa é o CCD (*Cyclic Coordinate Descent*) desenvolvido por (WELMAN, 1993).

O princípio deste método é o seguinte: as juntas são percorridas do final para o início, isto é, iniciam no *end-effector* e terminam na base da cadeia. Em cada iteração, todas as juntas são visitadas e a configuração de cada uma é atualizada de maneira a minimizar o erro.

Inicialmente é preciso criar dois vetores, um que vai da posição da junta ao ponto extremo do membro e outro que vai da posição da junta até a posição alvo ver Figura 4.7, depois calcula-se o produto escalar entre estes dois vetores para encontrar o ângulo entre eles.

As iterações são repetidas até que o erro fique menor que um valor pré-determinado ou atinja um número máximo de iterações.

5. Discussões Finais

Primeiramente foi estudada a estrutura mais adequada para representar estes objetos, chegando à conclusão que estruturas hierárquicas, por manterem a integridade das estruturas, são as mais adequadas para esta representação. Entre as estruturas que apresentam esta característica a mais utilizada é a estrutura em árvore. Pois o entendimento das estruturas de dados influência o entendimento da capacidade de realização de operações sobre os dados (a animação).

Depois foram estudados os métodos de animação por quadro-chave, animação comportamental e captura de movimento. Considerando que a forma de se fazer animações no ActionBuilder é a animação por quadro-chave foi feito um estudo dos principais métodos de interpolação, ver Capítulo 2.3, e o que deve ser levado em conta quando se escolhe um deles para uso em um sistema de animação e de como as propriedades das curvas influenciam no resultado final da animação.

Foi observado que o V-ART não implementa um tipo de interpolação cúbica, que é um tipo de curva que permite maior flexibilidade ao animador para fazer uma animação.

Além disso o ActionBuilder não possui uma estrutura adequada para representar um quadro-chave, para que alguns recursos possam ser implementados foi necessário a proposição de uma classe que representasse este conceito, como conseqüência disto é necessário mudar a forma como a animação é representada. Por isso foi preciso também mudar a forma como a animação é descrita em um arquivo XML.

A utilização do Blender e do Art of Illusion apontou funcionalidades que podem ser implementadas para aumentar a capacidade e a facilidade de uso do ActionBuilder. Estes editores foram descritos dando ênfase às interfaces e a dificuldade para se fazer e modificar uma animação, a interface gráfica facilita a interação do animador com o programa. A utilização de um editor de curvas possibilita ao animador observar as curvas que definem a animação, isso aliado ao uso

de uma estrutura para representar um quadro chave permite uma capacidade maior para que o animador modifique a animação.

Por fim, um sistema de cinemática inversa poupa o trabalho do animador de definir todos os ângulos para posicionar um membro. Dentre os métodos disponíveis para resolver este problema, foi escolhido o CCD que é um método eficiente e de fácil implementação.

Com estes recursos a realização de animações no V-ART irá se tornar mais simples e formará uma base para implementação de mais funcionalidades no futuro.

Referências Bibliográficas

ART of Illusion. Disponível em www.artofillusion.org/, acessado em 10/2009.

BLENDER. Disponível em www.blender.org, acessado em 09/2009.

BOULIC, R.; RENAULT, O. 3d hierarchies for animation. In: *New Trends in Animation and Visualization*. Nova Iorque, NY, EUA: Wiley Professional Computing, 1991. p. 59–77.

BREGLER, C.; MALIK, J. *Video Motion Capture*. Disponível em: http://www.debevec.org/IBMR99/75paper.pdf, acessado em 10/2009, 1997.

BURTNYK, N.; WEIN, M. Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Comunications of the ACM*, v. 19, n. 10, p. 564–569, 1976.

CATMULL, E. A system for computer generated movies. In: *ACM'72: Proceedings of the ACM annual conference*. Nova Iorque, NY, EUA: ACM, 1972. v. 1, p. 422–431.

CATMULL, E. The problems of computer assisted animation. *SIGGRAPH Computer Graphics*, v. 12, n. 3, p. 348–353, 1978.

CGA, Computer Graphics and Applications. Disponível em http://www.computer.org/portal/web/cga/home.

FOLEY, J. D. et al. *Computer Graphics, Principles and Practice*. 2. ed. Nova Iorque, NY, EUA: Addison-Wesley, 1990.

FUNGE, J.; TU, X.; TERZOPOULOS, D. Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In: *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. Nova Iorque, NY, EUA: ACM Press/Addison-Wesley Publishing Co., 1999. p. 29–38.

GLEICHER, M. Retargeting motion to new characters. In: *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. Nova Iorque, NY, EUA: ACM, 1998. (Annual Conference Series), p. 33–42.

GOMES, J.; VELHO, L. *Fundamentos da Computação Gráfica*. Rio de Janeiro, RJ, Brasil: IMPA, 2003.

H-ANIM, HUMANOID ANIMATION WORKING GROUP. *Specification for a Standard Humanoid*. Disponivel em http://h-anim.org/Specifications/H-Anim1.1/, acessado em 09/2009.

HERDA, L.; URTASUN, R.; FUA, P. Implicit surface joint limits to constrain video-based motion capture. In: *Proceedings 8th European Conference on Computer Vision*. Praga, República Tcheca: Springer, 2004. p. 405–418.

JUNG, C. F. *Metodologia para pesquisa & desenvolvimento: aplicada a novas tecnologias, produtos e processos*. Rio de Janeiro, RJ, Brasil: Axcel Books do Brasil, 2004.

KOCHANEK, D. H. U.; BARTELS, R. H. Interpolating splines with local tension, continuity, and bias control. *SIGGRAPH Computer Graphics*, v. 18, n. 3, p. 33–41, 1984.

MACIEL, A.; NEDEL, L. P.; FREITAS, C. M. Anatomy-based joint models for virtual human skeletons. In: *Computer Animation*. Genebra, Suíça: IEEE Computer Society, 2002. p. 220–224.

MILLAR, J.; HANNA, J.; KEALY, S. A review of behavioural animation. *Computer Graphics*, v. 23, n. 1, p. 127–143, 1999.

NEDEL, L. P. *Animação por Computador: Evolução e Tendências*. 2000. Disponível em www.inf.ufrgs.br/cg/publications/eri2000–texto.pdf, acessado em 06/2009.

PARENT, R. *Computer Animation, Algorithms and Techniques*. São Francisco, CA, EUA: Morgan Kaufmann Publishers Inc., 2002.

PARKE, F. Computer generated animation of faces. In: *ACM'72: Proceedings of the ACM annual conference*. Nova Iorque, NY, EUA: ACM, 1972. v. 1, p. 451–457.

PULLEN, K. A. *Motion Capture Assisted Animation: Texturing and Synsthesis*. Tese (Doutorado) — Stanford University, 2002.

REYNOLDS, C. W. Flocks, herds, and schools: A distributed behavioral model. In: *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. Nova Iorque, NY, EUA: ACM, 1987. v. 21, n. 4, p. 25–34.

SIGGRAPH, Special Interest Group on Graphics and Interactive Techniques. Disponivel em http://www.siggraph.org/.

SUPPIA, A. L. P. de O. História de dez anos de produção digital inclui brasileiros. *Ciencia e Cultura*, v. 58, n. 3, p. 56–57, 2006.

THALMANN, N. M.; THALMANN, D. Computer animation. *Encyclopedia of Computer Science and Technology, Marcel Dekker Inc.*, v. 29, p. 91–117, 1993.

V-ART, Virtual Articulation for Virtual Reality. Disponível em http://www.codeplex.com/vart, acessado em 03/2009.

WELMAN, C. *Inverse Kinematics and Geometric Constraints for Articulated Figure Manipulation*. Tese (Doutorado) — Simon Fraser University, 1993.

WXWIDGETS. Disponível em http://www.wxwidgets.org/, acessado em 09/2009.

A. Nova DTD para a animação

As alterações propostas para representar a animação devem ser estendidas às DTD's (*Document Definition Type*) que definem a animação no V-ART.

```
<!ELEMENT action (frame+)>
<!ATTLIST action numframes CDATA #REQUIRED>

<!ATTLIST action action_name  CDATA #REQUIRED
                 speed        CDATA #REQUIRED
                 cycle        CDATA #REQUIRED
                 model        CDATA #IMPLIED>

<!ELEMENT interpolation EMPTY>
<!ATTLIST interpolation type  CDATA #REQUIRED>

<!ELEMENT dof EMPTY>
<!ATTLIST dof dofID     CDATA #REQUIRED
              position CDATA #REQUIRED>

<!ELEMENT joint (dof, dof, dof) >

<!ELEMENT frame (joint+) >
<!ATTLIST frame time CDATA #REQUIRED>
```

B. Algoritmo para executar a animação

```
// constói a animação
para cada quadro (Q) entre Qi e Qi+1 faça
    tempo(Q) = posição do quadro no vetor de animação x 1 / quantidade de quadros por segundo
    para cada junta(J) em Q
        para cada grau de liberdade em J faça
                valor = Interpola( valor do Dof em Qi, valor do Dof em Qi+1, tempo(Q) )


// a função Interpola se utiliza de um metodo de interpolação para
//calcular o valor de um grau de liberdade com base nos graus de
// liberdade correspondentes nos quadros Qi e Qi+1

// executa a animação
 Início
    posicaoAtual = 0;
    enquanto frameAtual < numero de quadros
        quadroAtual = animacao.GetQuadro(posicaoAtual)
        SetFrame(quadroAtual)
        espera( tempo por quadro )
        posicaoAtual = posicaoAtual + 1
   fim enquanto
Fim

/* a função GetQuadro(i) retorna a posição i
 do vetor de quadros da animação
 SetFrame é uma função que recebe um quadro como parâmetro
 e faz com que as juntas do objeto articulado que está sendo animado
 assumam valores iguais às juntas correspondentes do
 quadro que está sendo passado como parâmetro */
```