Animação de um Personagem Virtual Utilizando Captura Óptica de Movimento com Marcações Especiais

Giovane Roslindo Kuhn (FURB)
brain@netuno.com.br

Paulo César Rodacki Gomes (FURB)
rodacki@inf.furb.br

Resumo. Este trabalho utiliza os conceitos para a captura óptica de movimento humano (MoCap), utilizando marcações especiais sobre o corpo do ator, para a animação de um personagem virtual 3D. É apresentando o desenvolvimento de um protótipo de software que implementa as etapas de um sistema MoCap, gerando arquivos no formato BVH contendo a animação. Para isso, são aplicadas técnicas de processamento de imagens, no intuito de segmentar os marcadores, em seguida são aplicadas técnicas de predição para rastrear os marcadores ao longo de vídeo. Após isto, um modelo articulado é encaixado aos marcadores, para então ser rastreado ao longo vídeo e gerar o arquivo BVH com a animação do personagem virtual. Para o rastreamento do modelo é proposto um algoritmo que utiliza relações matemáticas entre as articulações do modelo para efetuar os seus posicionamentos.

Palavras-chave: Animação, BVH, Captura de movimento, Estruturas articuladas, MoCap, Personagem virtual

1 Introdução

O conceito de animação é antigo e analisando as suas origens, nota-se que os primeiros trabalhos nesta área são datados muito antes de invenção dos computadores. Segundo Parent (2002), a definição mais simples para animação é a geração de uma seqüência de imagens que retrata o movimento relativo de objetos de uma cena sintética, e possivelmente, do movimento de uma câmera virtual.

O início das experiências com animação por computadores, representou um grande avanço nas técnicas de animação. Uma das técnicas mais utilizadas na atualidade é a animação através de captura de movimento (SILVA, 1998), também chamada de *Motion Capture* (MoCap). O foco de estudo deste trabalho é especificamente a captura óptica de movimento, que consiste em colocar marcadores especiais no objeto durante o processo de captura do seu movimento e capturar as coordenadas destes marcadores através do uso de técnicas de processamento de imagens.

A alta qualidade do movimento gerado pela captura óptica de movimento, desperta interesse nas mais diversas áreas, como a indústria cinematográfica, jogos, ergonomia, desempenho desportivo e análise de movimento (PARENT, 2002). Porém utilizar um sistema MoCap ainda é uma realidade distante da grande maioria das empresas, universidades e grupos de pesquisa em nosso país, devido o alto custo dos *softwares* e *hardwares* envolvidos.

O grupo de pesquisa de ergonomia da Universidade Regional de Blumenau (FURB) desenvolve projetos utilizando análise de postura e movimento de pessoas, que necessitam ser enviados para a Universidade do Sul de Santa Catarina (UNISUL), localizada em Florianópolis, para serem analisados pelo sistema existente naquela instituição. O presente trabalho pretende contribuir neste sentido, dando continuidade as pesquisas em captura de movimento humano no Departamento de Sistemas e Computação.

Para atender esta necessidade, é implementado um protótipo de *software* para animar um personagem virtual utilizando a captura óptica de movimento. O protótipo utiliza o conjunto de

vídeos disponibilizados pelo grupo de pesquisa de ergonomia da FURB. Cada vídeo contém a gravação em plano sagital de uma criança caminhando, com marcadores especiais nas principais articulações do seu corpo. O plano sagital corresponde a uma visão lateral da pessoa a ser filmada.

A Seção 2 apresenta conceitos de animação e captura de movimento utilizados no trabalho, e um detalhamento do formato de arquivo BVH. A seguir, a Seção 3 apresenta aspectos sobre desenvolvimento do protótipo, detalhando as etapas especificadas e implementadas do sistema MoCap. Por fim, na Sessão 4 são apresentadas as conclusões deste trabalho.

2 Captura de movimento e animação

Esta seção tem por objetivo apresentar os conceitos de animação e captura de movimento utilizados neste trabalho, através de modelos para animar personagens, as etapas no processo de captura de movimento e o formato de arquivo BHV, utillizado para armazenar dados da animação e captura de movimento.

2.1 Personagem virtual

Em sistemas de animação, uma das primeiras etapas do ciclo de desenvolvimento é definir o modelo do personagem virtual, isto é, definir como o ator a ser animado é representado no computador e que parâmetros serão informados para animar este personagem (SILVA, 1998).

Geralmente o personagem virtual é modelado com estruturas articuladas, que consiste em um conjunto de objetos rígidos conectados por articulações. Estas articulações formam o vínculo geométrico entre os objetos, permitindo o movimento relativo entre eles.

No computador estruturas articuladas são representadas por estruturas hierárquicas (árvores), onde a posição de cada articulação é definida através da composição em seqüência das transformações das articulações anteriores. Com isto, apenas a primeira articulação da estrutura precisar ser posicionada no espaço, enquanto o resto da estrutura é posicionado apenas pelos ângulos entre as articulações, chamados de ângulos relativos. São estas informações, posição da articulação e ângulo relativo entre o restante das articulações, que são os parâmetros da animação do personagem virtual .

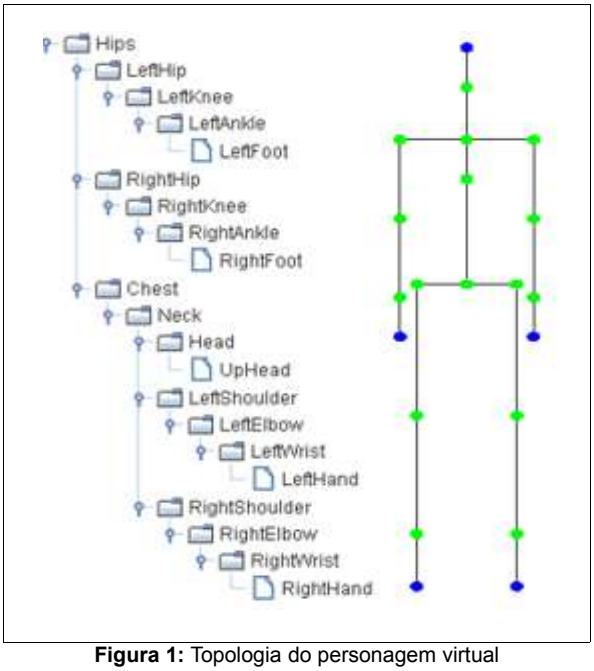

Figura 1: Topologia do personagem virtual

A fig. 1 apresenta a topologia do personagem virtual utilizado neste trabalho, que possui 16 articulações e 20 segmentros rígidos. A raíz da estrutura é o quadril do personagem e qualquer transformação nesta articulação tem efeito sobre toda a estrutura hierárquica.

2.2 Captura de movimento

A primeira etapa na captura de movimento, consiste em vestir o ator com marcadores reflexivos nos locais do corpo que se deseja capturar e posteriormente gravar o movimento do

ator, utilizando preferencialmente câmeras digitais de alta resolução e taxa de amostragem. Esta etapa não é realizada pelo presente trabalho, pois a captura do movimento do ator foi efetuada pelo grupo de pesquisas de ergonomia da FURB.

A etapa seguinte é efetuar processamentos nos dados capturados pelas câmeras para extrair o posicionamento 3D dos marcadores, para isto, técnicas de processamento de imagens são aplicadas em cada quadro do vídeo para realçar e segmentar os marcadores. Depois de segmentados os marcadores, são utilizadas técnicas de triangulação para efetuar o posicionamento 3D dos marcadores, com base nas informações 2D de cada câmera. O presente trabalho não precisou utilizar técnicas de triangulação, pois os vídeos disponibilizados gravam em plano sagital o movimento do ator.

A ultima etapa consite em gerar a trajetória dos marcadores entre os quadros do vídeo e encaixar o modelo do personagem virtual a estes marcadores rastreados. Os *softwares* de mercado não mencionam as técnicas computacionais utilizadas nesta etapa, o que se menciona, é que quando marcadores chaves não são rastreados, devido uma oclusão do marcador, por exemplo, é necessário que o usuário auxilie o *software* no rastreamento do marcador.

2.3 Padrão BVH

O BVH é um formato de arquivo para armazenar dados da captura de movimento e/ou animação. A escolha deste tipo de arquivo para o presente trabalho, deve-se ao fato do formato ser um padrão entre *softwares* de captura de movimento e suportado pelas principais ferramentas de animação do mercado, como SoftImage e 3D Studio MAX (BAERLE e THINGVOLD, [2005?]).

Um arquivo BVH é dividido em duas partes principais: dados da estrutura hierárquica do modelo e dados da animação deste modelo. O quadro 1 apresenta um exemplo de arquivo BVH, onde é possível notar a distinção entre os dados do modelo e da animação. Este exemplo está baseado na topologia do personagem virtual utilizado neste trabalho.

```
HIERARCHY
ROOT Hips
{
  OFFSET 0 0 0
  CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
  JOINT LeftHip
  {
    OFFSET 15 0 -0
    CHANNELS 3 Zrotation Xrotation Yrotation
    ...
        End Site
        {
          OFFSET 0 -20 0
          CHANNELS 0
        }
    ...
}
MOTION
Frames: 95
Frame Time: 0.0400
0.0 195.0 35.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.0 0.0 0.0 0.0 0.0
...
```

Quadro 1: Exemplo de arquivo BVH

A palavra `HIERARCHY` define o início da estrutura hierárquica, `ROOT` define a articulação raiz desta estrutura, `End Site` indica os end-effectors da estrutura e `JOINT` o restante das articulações. O `OFFSET` indica a posição relativa (X, Y e Z) da articulação em relação ao seu pai, `CHANNELS` indica os parâmetros que serão animados da articulação, o BVH suporta seis tipos de parâmetros:

- `Xposition`, `Yposition` e `Zposition`: indicam a posição relativa desta coordenada em relação ao seu pai na hierarquia;

- `Xrotation`, `Yrotation` e `Zrotation`: indicam o ângulo de rotação sobre o respectivo eixo, a ordem em que aparecem na definição deve ser respeitada no momento da animação.

A palavra `MOTION` indica o início dos dados da animação, `Frames` a quantidade de quadros da animação, `Frame Time` o tempo de duração de um quadro em segundos e depois cada linha contém os valores dos parâmetros de um quadro da animação.

Os parâmetros apresentados neste exemplo são aqueles comumente utilizados em arquivos BVH, onde apenas a raiz da estrutura contém dados translacionais e o restante das articulações contém apenas dados rotacionais. Porém, o formato permite definir parâmetros translacionais para todas as articulações, um problema do formato translacional é que a distância entre as articulações pode variar durante a animação.

3 Desenvolvimento do protótipo

O projeto recebe o nome de *Apollo* (APOLLO, 2005) e utiliza a metodologia de desenvolvimento em espiral, na qual especificação e implementação são construídas em conjunto. Para especificação é utilizada a notação *Unified Modeling Language* (UML) em conjunto com a ferramenta *Java UML Modeling Tool* (JUDE, 2005) e conceitos de análise orientada a objetos.

Na implementação do protótipo de *software* é utilizada a linguagem de programação *Java* (JAVA, 2005) e o ambiente de desenvolvimento *Eclipse* (ECLIPSE, 2004). Para trabalhar com vetores é utilizada a biblioteca *Vecmath* do *Java3D* (JAVA3D, 2003).

Na manipulação dos vídeos é utilizada a biblioteca *Java Media Framework* (JMF, 2003). Os visualizadores utilizam a biblioteca *Java bindings for OpenGL* (JOGL, 2005) que segue a especificação *OpenGL 2.0* (Akeley e Segal, 2004).

3.1 Diagrama de atividades

O usuário do protótipo exerce dez atividades básicas para poder capturar o movimento de um ator em um vídeo e então animar um personagem virtual 3D. A fig. 2 apresenta o diagrama com as principais atividades efetuadas durante o processo.

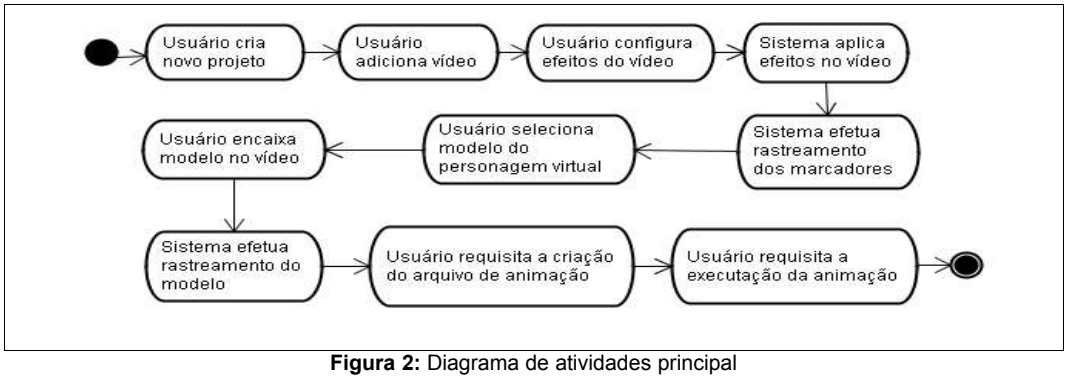

Figura 2: Diagrama de atividades principal

O fluxo de atividades principal consiste no usuário criar um novo projeto, neste projeto o usuário adiciona vídeos a serem analisados durante o processo de captura de movimento. O usuário tem condições de configurar os efeitos a serem aplicados no vídeo, no intuito de realçar os marcadores no corpo do ator, com isto, facilitar o segmentação destes marcadores.

Após configurado os efeitos, o usuário requisita para o sistema aplicar estes efeitos no vídeo, é neste momento que os marcadores são segmentados, isto é, a coordenada de cada marcadores é

extraída em cada quadro do vídeo. Com isto, o sistema pode efetuar o rastreamento destes marcadores, que consiste em analisar a trajetória dos marcadores ao longo do vídeo.

Feito o rastremanto dos marcadores, o usuário seleciona o modelo do personagem virtual a ser utilizado na animação e enaixa este modelo nos marcadores segmentados na etapa anterior. Após encaixar o modelo, o usuário requisita para rastrear este modelo ao longo do vídeo, enfim, definir a posição de cada articuação para cada quadro do vídeo.

Depois de rastrado o modelo, o usuário requisita a criação do arquivo BVH com a animação do personagem virtual e por fim requisita ao sistema para executar e então poder visualizar a animação do personagem virtual 3D.

3.2 Módulos

Com base nos requisitos levantados e no fluxo de atividades do projeto, o protótipo é dividido em cinco módulos: núcleo do protótipo, módulo de processamento do vídeo, módulo de rastreamento dos marcadores, módulo de rastreamento do modelo e módulo de animação.

No núcleo do protótipo é mantida a tela principal do aplicativo, onde o usuário cria novos projetos, adiciona e remove vídeos destes projetos. Neste módulo também são mantidos os executores e visualizadores de vídeo e animação, que são um conjunto de classes utilizadas por todos os outros módulos do protótipo.

No módulo de processamento do vídeo é mantida a tela e o conjunto de classes para o usuário manipular os vídeos do projeto e configurar os efeitos a serem aplicados nestes vídeos. Neste módulo também são mantidas as classes para segmentar os marcadores em cada quadro do vídeo.

No módulo de rastreamento dos marcadores é mantida a tela e o conjunto de classes para rastrear os marcadores em cada quadro do vídeo, isto é, definiar a trajetória de cada marcador ao longo do vídeo.

O módulo de rastreamento do modelo é responsável por manter a tela e o conjunto de classes para o usuário encaixar o modelo no primeiro quadro do vídeo, para então, efetuar o rastreamento deste modelo por todo o vídeo.

No módulo de animação é mantido o conjunto de classes para a criação do arquivo de BVH com a animação do personagem virtual e a tela para a execução e visualização desta animação.

3.3 Tela principal

Na tela principal do protótipo o usuário tem opção de criar um novo projeto e/ou abrir um projeto existente. Nesta tela o usuário mantêm os vídeos deste projeto, que posteriormente são analisados para capturar o movimento e animar o personagem virtual. Ao lado esquerdo da fig. 3 é apresentada a tela para criar e abrir projeto e ao lado direito a tela para manter os vídeos de um projeto.

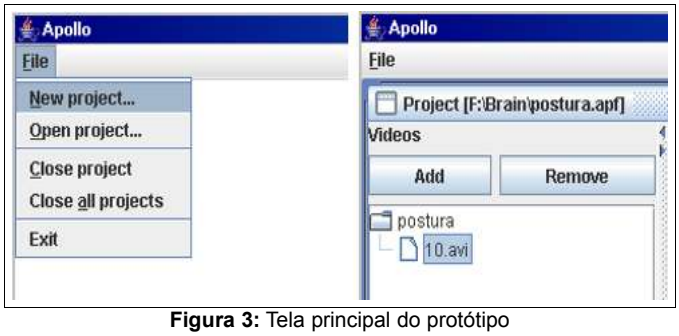

Figura 3: Tela principal do protótipo

3.4 Processamento do vídeo

Na tela de processamento do vídeo o usuario pode adicionar e/ou remover efeitos que são aplicados no vídeo, com o objetivo de realçar e segmentar os marcadores do ator. Esta etapa

envolve algum conhecimento do usuário em processamento de imagens, escolhendo os efeitos adequados a serem aplicados no vídeo.

O quadro 2 apresenta os onze efeitos implementados no protótipo que o usuário pode utilizar e configurar para segmentar os marcadores no processamento do vídeo.

A fig. 4 apresenta a tela para manipulação dos efeitos de um vídeo. Ao lado esquerdo da tela o usuário pode visualizar o vídeo original e logo abaixo o vídeo com os efeitos aplicados. Do lado direito da tela são mantidos os efeitos do vídeo e na parte inferior da tela são configuradas as propriedades de cada efeito.

Efeito	***Objetivo***
Escala de cinza	Converter a imagem para escala de cinza
Brilho e contraste	Aumentar ou diminuir a intensidade da imagem
Filtro por limiar	Segmentar a imagem em regiões de interesse
Filtro negativo	Inverter a intensidade da imagem
Filtro mediana	Remover ruídos da imagem, dando um efeito de suavização
Filtro gaussiano	Remover ruídos e detalhes da imagem, dando um efeito de suavização
Filtro laplaciano	Detectar bordas na imagem
Filtro laplaciano gaussiano	Remover ruídos e detectar bordas na imagem
Filtro sobel	Detectar bordas na imagem
Bounding-box de objeto	Detectar grupos de *pixel* conectados na imagem
Transformada de hough	Reconhecer formas geométricas na imagem

Quadro 2: Lista de efeitos do protótipo

Depois de configura dos os efeitos, o usuário requisita para o protótipo criar um vídeo com os todos os efeitos aplicados, é neste momento que o sistema segmenta os marcadores em cada quadro do vídeo, para que posteriormente possam ser rastreados pelo módulo de rastreamento de marcadores.

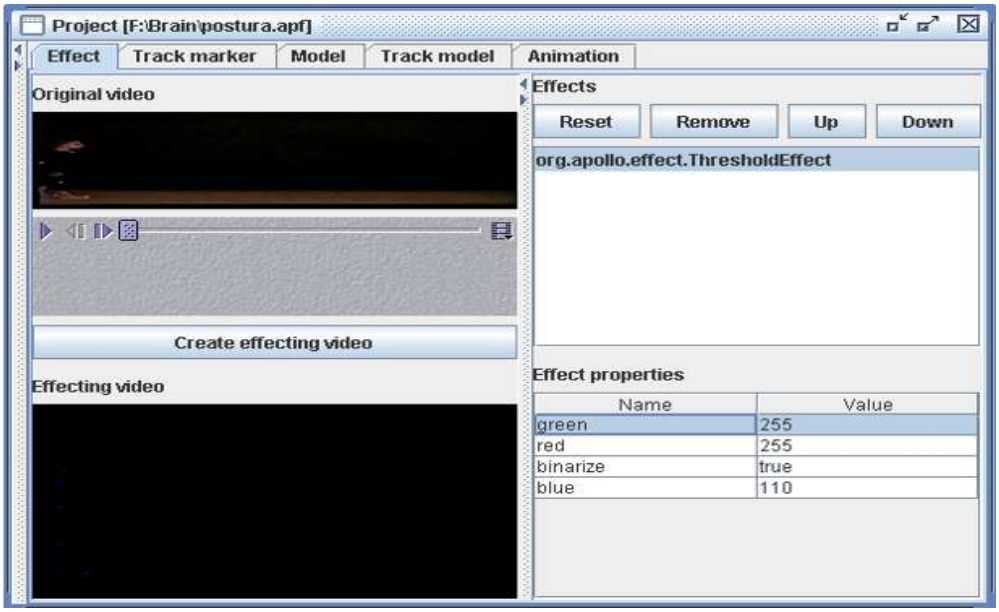

Figura 4: Tela de processamento do vídeo

3.5 Rastreamento dos marcadores

A tela de rastreamento de marcadores, apresenta na fig. 5, permite o usuário intervir nos marcadores segmentados no processamento do vídeo. Esta característica existe pois muitas vezes a qualidade do vídeo processado não é adequada, com isso, muito marcadores não são segmentados, exigindo que o usuário adicione manualmente marcadores adicionais.

Figura 5: Tela de rastreamento dos marcadores

Nesta tela o usuário requisita para o sistema rastrear os marcadores ao longo do vídeo, rastrear um marcador significa identificá-lo e informa sua posição em cada quadro do vídeo. O processo de identificação utilizado pelo sistema é de colorização, isto é, cada marcador identificado recebe uma cor única, marcadores com cor branca são considerados como não identificados.

O algoritmo de rastreamento inicialmente atribui uma cor (identificação) para os marcadores do primeiro quadro do vídeo e depois itera por cada quadro do vídeo identificando os marcadores no quadro subsequüente, isto é, ao analisar o quadro 1 o algoritmo tenta identificar os marcadores no quadro 2 e assim sucessivamente.

Para cada marcador é aplicada uma seqüencia de técnicas, e a primeira técnica é a predição de 1 quadro. Esta técnica consiste em analisar a trajetória do marcador do quadro K-1 para o quadro K, e com base nesta trajetória predizer a posição do marcador no quadro K+1. Então tenta-se achar um marcador de cor branca no quadro K+1 que esteja próximo a posição predita, caso seja encontrado, este marcador no quadro K+1 passa a ter a mesma cor do marcador do quadro N.

A segunda técnica aplicada é a do marcador mais próximo, então tenta-se achar um marcador de cor branca no quadro K+1 que esteja próximo ao marcador do quadro K, se encontrou, este marcador recebe a mesma cor.

A terceira técnica aplicada é a predição de N quadros, que para este protótipo está fixada em no máximo três quadros. O funcionamento desta técnica é idêntico ao da predição de 1 quadro, porém caso não encontre um marcador próximo a posição predita, esta técnica continua analisando até N quadros a frente. Esta técnica serve para situações em que um marcador “desapareça” por alguns quadro e então “reapareça”. Para os quadros em que o marcador esteve “desaparecido” são criados marcadores intermediários através de interpolação linear.

Os marcadores que não são identificados por nenhuma das técnicas são considerados perdidos e deixa de ser identificado no restante do vídeo.

3.6 Encaixar modelo 3D

A tela de rastreamento do modelo, apresentada na fig. 6, possibilita o usuário encaixar o modelo do personagem virtual ao ator no primeiro quadro do vídeo. Este encaixe deve ser efetuado pois as dimensões do modelo não são iguais as dimensões do ator.

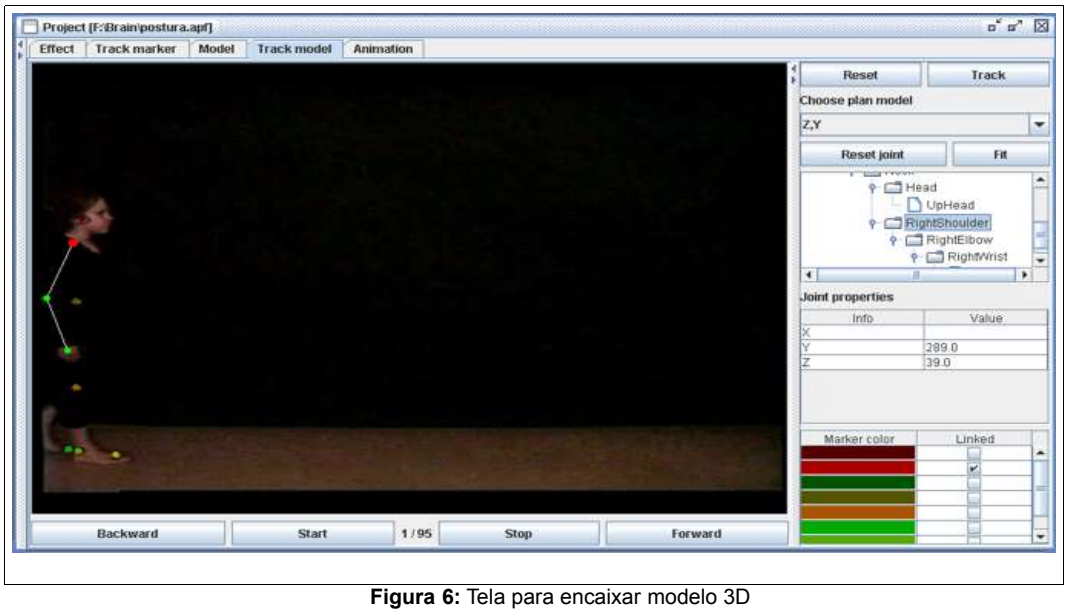

Figura 6: Tela para encaixar modelo 3D

Para encaixar o modelo o usuário informa a localização, atrabés de um clique na tela, das articulações do personagem. O usuário não precisa informar a localização de todas as articulações, pois o algoritmo efetua o encaixe automático de articulações faltantes, porém quanto mais articulações informadas, mais precisa a fica animação do personagem virtual.

Depois de informadas as articulações, o usuário requisita para o sistema encaixar o restante do modelo. Na primeira etapa do algoritmo é calculado um fator de ajuste utilizando as articulações informadas pelo usuário e o modelo original, isto é, calcular quanto o modelo original diminui ou aumentou.

O segundo passo consiste em navegar pela estrutura hierárquica do personagem e criar o novo modelo encaixado ao ator. As articulações que não são informadas pelo usuário, têm sua posição simulada através do fator de ajuste da etapa anterior.

A terceira etapa é responsável por simular as articulações que não são visualizadas pelo plano sagital. A posição destas articulações é baseada na posição da articulação do outro lado do modelo, porém a coordenada que o plano não visualiza tem seu sinal invertido. Por exemplo, o pé direito está nas coordenadas (-10, 20, 30), o pé esquerdo estará nas coordenadas (10, 20, 30), a coordenada X tem sinal invertido.

A fig. 6 apresenta a tela para encaixar o modelo, do lado direito da tela estão as articulações da estrutura hierárquica e do lado esquerdo é onde o usuário informa a posição das articulações.

3.7 Rastreamento do modelo

A tela de rastreamento do modelo, apresentada na fig. 7, permite o usuário requisitar e visualizar o rastreamento do modelo em todos os quadros do vídeo. Para o rastreamento do modelo é proposto um algoritmo flexível e escalável, onde técnicas podem ser inseridas sem

comprometer o funcionamento das técnicas já existentes. O algoritmo consiste em iterar e analisar cada quadro do vídeo, e a cada iteração é aplicado um conjunto de técnicas em ordem de confiabilidade, isto é, as técnicas que efetuam um posicionamento mais eficaz das articulações devem ser executadas antes. O presente protótipo implementa quatro técnicas para o posicionamento das articulações do modelo e estas técnicas são apresentadas na ordem de confiabilidade.

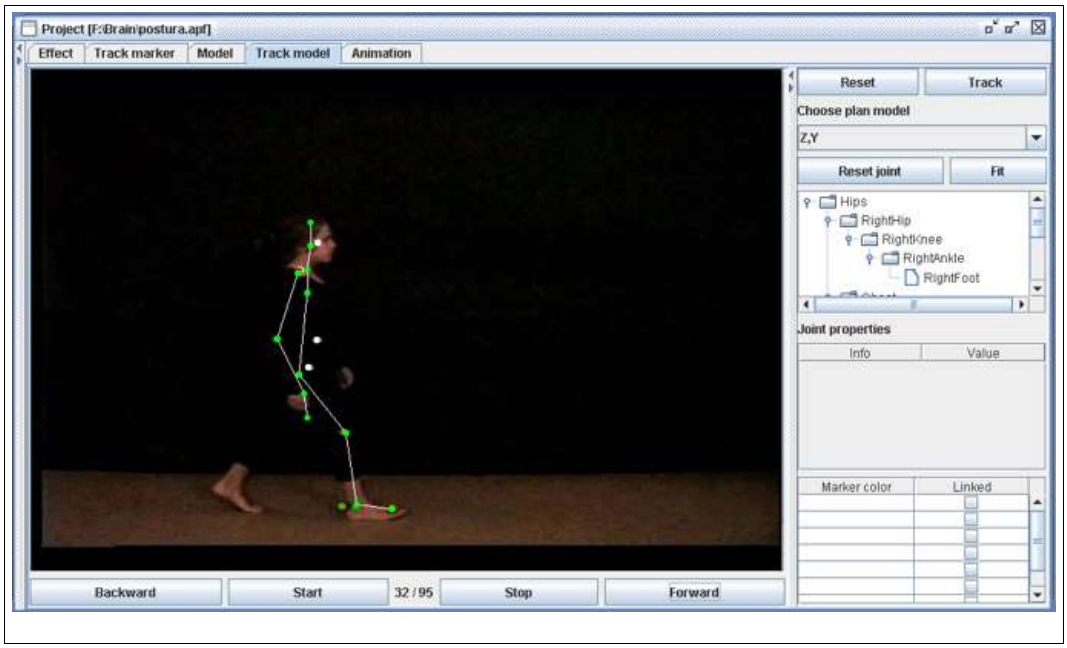

Figura 7: Tela de rastreamento do modelo

A primeira técnica aplicada para o posicionamento das articulações é a que utiliza os marcadores identificados no quadro analisado. Basicamente as articulações que tenham ligação com algum marcador e este marcador esteja identificado no quadro analisado, são posicionadas por esta técnica, em outras palavras, onde o marcador está a articulação estará. Está e a técnica que oferece o posicionamento mais preciso dentre as implementadas.

A segunda técnica aplicada para posicionar uma articulação é a que utiliza as suas articulações *pai* e *filha*. Para que esta técnica seja utilizada é necessário que as articulações *pai* e *filha* já estejam posicionadas no quadro analisado. Com base na posição das articulações *pai* e *filha*, mais a relação existente no modelo entre as três articulações, é possível calcular a posição da articulação faltante no quadro analisado.

A terceira técnica aplicada segue a mesma regra da técnica acima, só que utiliza as articulações *avô* e *pai* para efetuar o posicionamento. Conforme a regra, para que esta técnica seja utilizada, é necessário que as articulações *avô* e *pai* estejam posicionadas no quadro analisado.

A quarta técnica aplicada é a que utiliza o modelo para posicionar uma articulação, esta técnica é a mais imprecisa de todas, pois simplesmente copia a posição relativa da articulação que está no modelo, com isto, a posição relativa da articulação é sempre a mesma ao longo do vídeo.

Novas técnicas podem ser implementadas e inseridas neste algoritmo, basta que estas técnicas atendam a ordem de confiabilidade do algoritmo, por exemplo, uma técnica que utiliza as articulações *neta* e *bisneta* para o posicionamento.

3.8 Geração da animação

Na tela de animação, apresentada na fig. 8, o usuário tem a possibilidade de escolher o formato de dados que o arquivo BVH com a animação é gerado. O protótipo disponibiliza a opção de formato translacional, que armazena a posição relativa de cada articulação em relação a sua articulação *pai*. Um limitação deste formato é que a distância entre as articulações pode variar ao longo da animação.

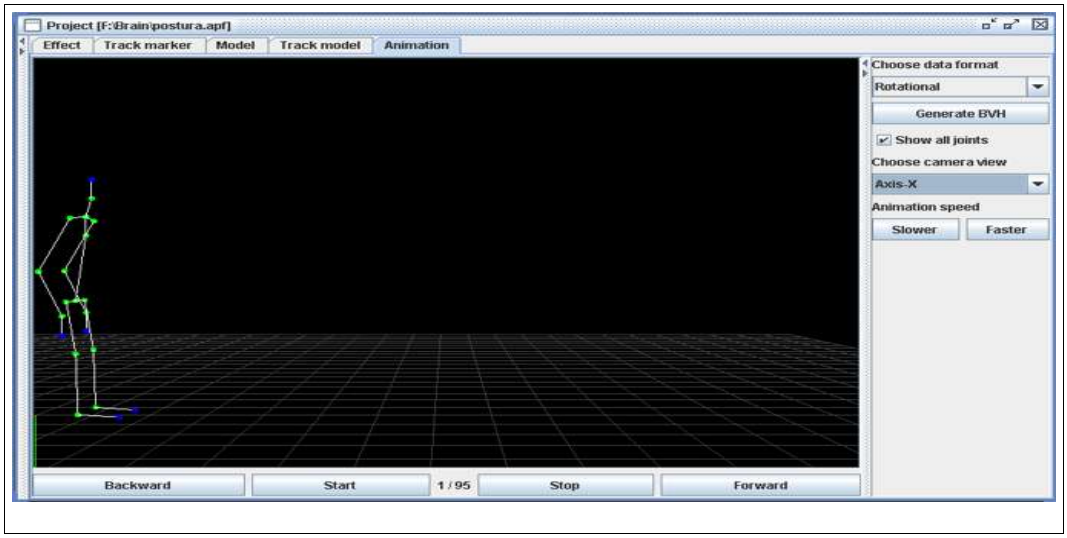

Figura 8: Tela para geração da animação

O segundo formato que o protótipo disponibiliza é o rotacional, que é o formato padrão para arquivos BVH. Neste formato é armazendo o ângulo relativo de cada articulação em relação a sua articulação *filha* para cada um dos eixos. No presente trabalho o formato rotacional está limitado apenas ao eixo X, já que os vídeos disponibilizados contêm apenas gravações do plano sagital.

Nesta tela o usuário requisita para gerar a animação, e o primeiro passo do algoritmo é gerar os dados da animação no formato definido pelo usuário. Depois são simulados os dados que não são visualizados pelo plano sagital, pois as gravações contêm apenas um lado do corpo humano, logo o outro lado deve ser simulado.

Para a simulação é utilizado a técnica de deslocamento de N quadros, que para este trabalho está fixado em 20 quadros. Para simular o dado de uma articulação não visualizada pelo plano é utilizada como base a respectiva articulação do outro lado do modelo, por exemplo, o pé esquerdo no quadro 1 tem os mesmos dados relativos ao pé direito no quadro 20.

Depois de gerados os dados do usuário e os dados simulados, o sistema gera o arquivo BVH com a animação do personagem virtual.

3.9 Execução da animação

Na tela de animação o usuário tem a possibilidade de executar a animação contida no arquivo BVH gerado. Ao lado direito da tela apresentada na fig. 8, o usuário pode selecionar uma das quatro câmeras disponibilizadas para visualização.

Na fig. 9 é apresentada uma seqüencia de quadros de uma animação, que teve como entrada um dos vídeos disponibilizados para o projeto. O formato utilizado para esta animação é o translacional.

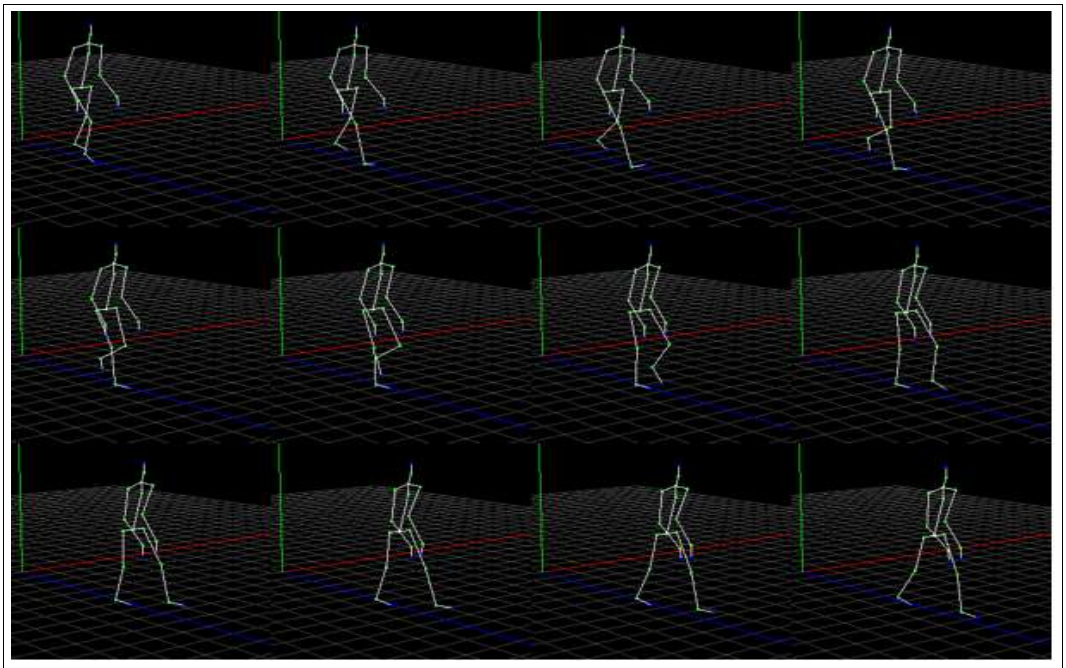

Figura 9: Sequência de quadros da animação

4 Conclusões

O presente trabalho apresentou a especificação e implementação de um protótipo de *software* para animação de um personagem virtual utilizando a captura óptica de movimento. Para isto, foi desenvolvido um módulo para processamento dos vídeos, permitindo ao usuário configurar os efeitos a serem aplicados no vídeo. O módulo de rastreamento dos marcadores foi implementado para identificar e fornecer as coordenadas 2D de cada marcador para cada quadro do vídeo. Outro módulo é o para rastreamento do modelo, permite ao usuário efetuar o encaixe do modelo ao ator do vídeo e posteriormente rastrear o modelo ao longo do vídeo. O último módulo desenvolvido é de animação, que tem por objetivo gerar e executar os arquivos BVH com animação. O desenvolvimento destes módulos soma 15.000 linhas de código, divididas em 110 classes implementadas no protótipo.

No desenvolvimento do protótipo foi utilizado um microcomputador AMD Sempron[TM] 2400+ com 1.66Ghz de freqüência e 1Gb de memória RAM. Quanto aos aspectos de desempenho do protótipo, o módulo de processamento do vídeo não executa sua tarefa em tempo real, devido ao tempo de processamento das imagens de cada quadro do vídeo. Todos os outros módulos, rastreamento dos marcadores, rastreamento do modelo e animação, obtiveram desempenho em tempo real, isto é, sem um tempo de espera para o usuário nas operações realizadas.

Como proposta de continuação do presente trabalho é sugerido a utilização de múltiplas câmeras para o rastreamento dos marcadores, utilizando processos de triangulação. Com isso, situações mais complexas podem ser capturadas e animadas pelo protótipo.

Outra sugestão é desenvolver um módulo para análise dos movimentos do ator, gerando relatórios e gráficos, atendendo assim, as necessidades do grupo de pesquisa de ergonomia da FURB.

5 Referências

AKELEY, Kurt; SEGAL, Mark. **The OpenGL graphics system**: a specification. Mountain View, 2004. Disponível em <http://www.opengl.org/documentation/specs/version2.0/glspec20.pdf>. Acesso em: 14 jun. 2005.

APOLLO. **Motion capture and animation system**. Version 0.1. [S.l]: 2005. Disponível em <http://apollo.dev.java.net>. Acesso em: 05 out. 2005.

BAERLE, Susan Van; THINGVOLD, Jeffrey. **Motion capture file format review.** Minneapolis, [2005?]. Disponível em <http://www.gdconf.com/archives/2000/vanbaerle.doc>. Acesso em: 10 jun. 2005.

ECLIPSE. **Eclipse platform**. Version 3.1. [S.l]: 2004. Disponível em: <http://www.eclipse.org>. Acesso em: 18 maio 2005.

JAVA. **Java[tm] 2 platform standard edition**. Version 5.0. [S.l]: 2005. Disponível em <http://java.sun.com/j2se/1.5.0/index.jsp>. Acesso em: 10 jun. 2005.

JAVA3D. **Java3D API**. Version 1.3.1. [S.l]: 2003. Disponível em <http://java.sun.com/products/java-media/3D/index.jsp>. Acesso em: 10 jun. 2005.

JMF. **Java media framework API**. Version 2.1.1e. [S.l]: 2003. Disponível em <http://java.sun.com/products/java-media/jmf/index.jsp>. Acesso em: 10 jun. 2005.

JOGL. **Java bindings for OpenGL**. Version 1.1b12. [S.l]: 2005. Disponível em <http://jogl.dev.java.net>. Acesso em: 10 jun. 2005.

JUDE. **Java UML object-oriented design tool**. Version 1.5.2 community. [S.l]: 2005. Disponível em <http://jude.esm.jp/>. Acesso em: 10 jun. 2005.

PARENT, Rick. **Computer animation**: algorithms and techniques. San Francisco: Morgan Kaufmann, 2002.

SILVA, Fernando Wagner Serpa Vieira da. **Um sistema de animação baseado em movimento capturado**. 1998. 101 f. Dissertação (Mestrado em Computação Gráfica) – Laboratório de Computação Gráfica COPPE/Sistemas, Universidade Federal do Rio de Janeiro, Rio de Janeiro.