FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES SOARES DA ROCHA” – UNIVEM
PROGRAMA DE MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

# UM SISTEMA ÓPTICO DE CAPTURA DE MOVIMENTOS BASEADO EM ESQUELETO HIERÁRQUICO UTILIZANDO A BIBLIOTECA ARTOOLKIT

**Fernando Lopes Giovanini**

MARÍLIA
2007

FERNANDO LOPES GIOVANINI

# Um Sistema Óptico de Captura de Movimentos Baseado em Esqueleto Hierárquico Utilizando a Biblioteca ARToolkit

Dissertação apresentada ao Programa de Mestrado do Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, para obtenção do título de Mestre em Ciência da Computação (Área de Concentração: Realidade Virtual).

Orientador:
Prof. Dr. Antonio Carlos Sementille

MARÍLIA
2007

FERNANDO LOPES GIOVANINI

# UM SISTEMA ÓPTICO DE CAPTURA DE MOVIMENTOS BASEADO EM ESQUELETO HIERÁRQUICO UTILIZANDO A BIBLIOTECA ARTOOLKIT

Banca examinadora da dissertação apresentada ao Programa de Mestrado da UNIVEM/F.E.E.S.R., para a obtenção do Título de Mestre em Ciência da Computação. Área de Concentração: Realidade Virtual.

Resultado: _____

ORIENTADOR: Prof. Dr. Antonio Carlos Sementille

1º EXAMINADOR: _____

2 º EXAMINADOR: _____

Marília, 08 de Novembro de 2007.

*Para meus pais, Miguel e Mirian*
*E para minha irmã, Camila.*

# AGRADECIMENTOS

Aos meus pais, Miguel e Mirian, pela paciência e compreensão dispensadas. Tenho absoluta certeza de que sem eles eu não teria conseguido. Uma página de agradecimentos seria pouco para demonstrar a minha gratidão.

À minha irmã, Camila, por ter me mostrado, nem sempre de forma muito afável, algumas verdades e caminhos óbvios que muitas vezes não percebi. E por ser, junto com meus pais, o maior exemplo de personalidade e responsabilidade que eu poderia ter.

Ao meu orientador, Prof. Dr. Antonio Carlos Sementille, por ser um dos maiores exemplos de professor e orientador que eu tive durante o curso de Mestrado, por seu conhecimento, paciência e humildade.

À Juliana, ou Ju, por corrigir minha Dissertação e fazer o meu Abstract mesmo em cima do prazo de entrega, e por ser durante os últimos anos uma ótima amiga.

Ao meu amigo Cristiano, o Bart, por passar algumas tardes comigo servindo de "cobaia" para os testes realizados. E por algumas das noites mal dormidas que tive.

Aos meus amigos/irmãos Paulo, Ian, Antonio Paulo, Patrícia e Ricardo, o Sebão, pois sem os momentos, as cervejas e as músicas que apreciamos juntos, teria sido muito mais difícil.

Aos meus novos amigos Ana Claudia, Fabrício, José Ivo, Sérgio e Richard, pois sei que posso contar com eles para tudo o que eu precisar e no momento em que houver necessidade. Ou não.

A todos os meus professores que de alguma forma me ajudaram com críticas ou elogios.

E aos amigos, colegas e parentes que eu possa ter esquecido, mas que sabem que deveriam estar aqui.

*"Existe uma teoria que diz que, se um dia alguém descobrir exatamente para que serve o Universo e por que ele está aqui, ele desaparecerá instantaneamente e será substituído por algo ainda mais estranho e inexplicável. Existe uma segunda teoria que diz que isso já aconteceu".*

Douglas Adams

GIOVANINI, Fernando Lopes. **Um Sistema Óptico de Captura de Movimentos Baseado em Esqueleto Hierárquico Utilizando a Biblioteca ARToolkit**. 2007. 127 f. Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2007.


RESUMO

Os sistemas de Captura de Movimentos humanos podem ser aplicados em diversas áreas de conhecimento por ser a forma mais fiel de transformar os movimentos humanos em dados computacionais. Entretanto, mesmo com a diminuição de custos do software e hardware utilizados, estes são acessíveis apenas a grandes estúdios de entretenimento e laboratórios de pesquisas. Este trabalho aborda as principais tecnologias de Captura de Movimentos e suas características, os dispositivos utilizados na Realidade Aumentada, as características da biblioteca ARToolkit e algumas das aplicações desenvolvidas com esta ferramenta. Também é apresentada a estruturação e implementação de um sistema de Captura de Movimentos de baixo custo baseado na biblioteca ARToolkit, utilizando um esqueleto hierarquizado.



**Palavras-chave:** ARToolkit. Captura de Movimentos corporais. Realidade Aumentada. Sistemas ópticos de captura.

GIOVANINI, Fernando Lopes. **Um Sistema Óptico de Captura de Movimentos Baseado em Esqueleto Hierárquico Utilizando a Biblioteca ARToolkit**. 2007. 127 f. Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2007.


ABSTRACT

The Human Motion Capture systems can be applied in a myriad of knowledge areas, since they are the most reliable manner of changing human movement into computational data. However, even presenting decreasing of costs in software and hardware, these systems are accessible only to great entertainment studios and research laboratories. In this manner, the present work is intended for approaching the main Motion Capture technologies as well as its characteristics, pieces of device used in Augmented Reality, ARToolkit library characteristics, and some applications developed by using this tool. It is also presented the structuring and implementation of a low-cost Motion Capture system based on the ARToolkit library using a hierarchized skeleton.



**Keywords:** ARToolkit. Augmented Reality. Body Motion Capture. Capture Optical Systems.

# LISTA DE ILUSTRAÇÕES

# LISTA DE TABELAS

# LISTA DE ABREVIATURAS E SIGLAS

2D: Bidimensional

3D: Tridimensional

AMC: *Acclaim Motion Capture*

ARMouse: *Augmented Reality Mouse*

ASCII: *American Standard Code for Information Interchange*

ASF: *Acclaim Skeleton File*

AV: Ambiente Virtual

AVI: *Audio Video Interleave*

BMP: *Bit Map*

BVA: *BioVision Animation*

BVH: *Biovision Hierarchy*

CAVE: *Cave Automatic Virtual Environment*

CM: Captura de Movimentos

TRC: Tubo de Raios Catódicos

DIP: *Dual in Package*

DSVL: *DirectShow Vídeo Processing Library*

DOF: *Degree of Freedom*

FPS: *Frames* por Segundo

GLUI: *Graphical Library User Interface*

GLUT: OpenGL *Utility Toolkit*

GUI: *Graphical User Interface*

HMD: *Head Mounted Display*

HUD: *Head-up Display*

IHC: Interface Homem-Computador

IRM: Imagens de Ressonância Magnética

JPEG: *Bit Map Joint Photographic Experts Group*

LED: *Light Emitting Diode*

LOA: Level of Articulation

MCP: *Motion Capture File*

OpenGL: *Open Graphics Library*

RA: Realidade Aumentada

RAM: *Random Access Memory*

RM: Realidade Misturada

RV: Realidade Virtual

SGI: Silicon Graphics Inc.

TC: Tomografia Computadorizada

TCL: Tela de Cristal Líquido

UDP: *User Datagram Protocol*

UvA-DRIVE: *Distributed Real-time Interactive Virtual Environment*

VRML: *Virtual Reality Markup Language*

VRD: *Virtual Retinal Displays*

WAV: *Waveform Audio Format*

WoW: *Window-on-the-World*

# SUMÁRIO

# INTRODUÇÃO

Ao longo da última década, as pesquisas na área de Realidade Aumentada cresceram significativamente, bem como o número de eventos e publicações voltadas para esta área. Pode-se citar entre os motivos deste crescimento a diminuição dos custos de equipamentos e a capacidade inerente desta tecnologia em combinar o real com o virtual, tornando a interação entre o ser humano e o computador mais natural e intuitiva.

Uma das áreas de aplicação da Realidade Aumentada é o rastreamento de movimentos humanos, também chamada de Captura de Movimentos. A função das aplicações de Captura de Movimentos é converter os movimentos realizados por um ator – que dependendo da finalidade da aplicação pode ser um paciente, um atleta, um ator, entre outros – para dados computacionais a fim de serem analisados e visualizados por *softwares* específicos.

A utilidade mais conhecida para a Captura de Movimentos é na área do entretenimento para animação de personagens de filmes, jogos e comerciais para a televisão. Entretanto, esta área não é a única em que se pode aplicar a Captura de Movimento, já que esta também é utilizada na Medicina (VALLINO, 2002) para a captura e análise de movimentos de pacientes, em aplicações militares (AZUMA, 1997), Robótica (MILGRAM, 1995), entre outras.

É possível dividir as técnicas de Captura de Movimentos em quatro principais categorias: óptica, mecânica, acústica e magnética (SILVA, 2002). Os sistemas do tipo óptico são compostos por fontes de luz (ou refletores) no corpo do ator representando as articulações deste e sensores ópticos (câmeras de vídeo) posicionado no ambiente onde ocorrerá a captura. Os movimentos realizados pelo ator são então capturados pelas câmeras e transformados em dados computacionais.

Em geral os sistemas de Captura de Movimentos disponíveis comercialmente são caros, o que faz com que sejam acessíveis apenas a grandes laboratórios de pesquisa e grandes empresas de entretenimento. Além do alto custo, os sistemas ópticos de Captura de Movimentos apresentam problemas relacionados à oclusão dos marcadores, necessidade de pós-processamento para a identificação das juntas, e estão sujeitos a erros de detecção dos marcadores.

Uma solução para o problema de custo e de identificação das juntas é a biblioteca ARToolkit (ARTOOLKIT,2006), que oferece uma opção para a construção de sistemas de Realidade Aumentada mais baratos, baseados em rastreamento óptico com marcadores passivos.

## Objetivos

Considerando esse contexto, este trabalho apresenta um sistema óptico de Captura de Movimentos utilizando a biblioteca ARToolkit. Esse sistema utiliza estrutura de esqueleto hierarquizada, permite a visualização dos movimentos em tempo real e a geração de um arquivo de captura em formato padrão, o BVH, aperfeiçoando o trabalho de Lourenço (2004). Para o desenvolvimento deste sistema de captura de movimentos, foi necessário realizar um estudo sobre a Realidade Aumentada, captura de movimentos humanos e alguns sistemas ópticos de rastreamento encontrados na literatura.

## Organização do trabalho

No Capítulo 1 é feito um estudo sobre a Realidade Aumentada, apontando as suas definições e aplicação. São descritos os equipamentos utilizados nessa tecnologia divididos em equipamentos de saída visual e outros. É dada maior ênfase aos equipamentos de saída visual pelo fato de a visão ser considerada o principal sentido do ser humano. Também são

descritos os problemas relacionados à Realidade Aumentada, e é descrita a biblioteca ARToolkit, seu funcionamento, suas limitações e aplicações.

O Capítulo 2 aborda os temas relacionados à Captura de Movimentos humanos. São apresentadas as principais tecnologias utilizadas, descrevendo as suas vantagens e desvantagens. Também é realizada uma comparação entre as técnicas e *softwares* de Captura de Movimentos de acordo com as formas de classificação empregadas, e é apresentada a especificação H-Anim, que tem por objetivo padronizar a forma como é estruturado um esqueleto humanóide, e por fim são descritos três formatos padrões de arquivos para a CM humanos: BVA, BVH e ASF/AMC.

Alguns trabalhos sobre rastreamento óptico encontrados na literatura são apresentados no Capítulo 3. Os trabalhos descritos estão relacionados à diversas áreas, tais como interação humano-computador, entretenimento e medicina. Alguns destes trabalhos foram implementados utilizando a biblioteca ARToolkit, citada anteriormente.

No Capítulo 4 é apresentado um sistema óptico de Captura de Movimentos baseado em esqueleto hierárquico utilizando a biblioteca Artoolkit. Este sistema permite armazenar os movimentos capturados em um formato de arquivo padrão. É também descrita a implementação deste sistema.

Já no Capítulo 5 são descritos os testes realizados e os resultados obtidos para avaliar o sistema desenvolvido. Os testes realizados estão relacionados à taxa de quadros por segundo obtida, precisão dos movimentos capturados e taxa de detecção dos marcadores rastreados, que são quadrados impressos em papel comum utilizados pelo ARToolkit para identificar as articulações do ator cujo movimento é capturado.

Logo em seguida são apresentadas as conclusões e sugeridos os trabalhos futuros para o aperfeiçoamento do sistema implementado. Por fim, são listadas as referências bibliográficas utilizadas neste trabalho.

# CAPÍTULO 1 - REALIDADE AUMENTADA E O ARTOOLKIT

Azuma (1997) define Realidade Aumentada (RA) como a integração entre objetos virtuais tridimensionais (3D) com um ambiente 3D real, em tempo real, tratando-se de uma variação da Realidade Virtual (RV), em que o usuário interage somente com o mundo virtual sem ter contato visual com o mundo real. O mesmo autor afirma em (AZUMA et al., 2001) que a RA visa suplementar o mundo real com objetos gerados por computador, fazendo com que estes pareçam coexistir no mesmo espaço no mundo real e em tempo real.

O termo RA é definido por Milgram e Kishino (1994) como um subconjunto de Realidade Misturada (RM) que pode ser localizada dentro de um *continuum* de virtualidade, onde em um dos extremos encontra-se o Ambiente Real e no outro o Ambiente Virtual (AV). Nesse *continuum* classifica-se uma aplicação como RA ou Virtualidade Aumentada de acordo com o nível de interação entre os mundos reais e virtuais, conforme mostra a Figura 1.

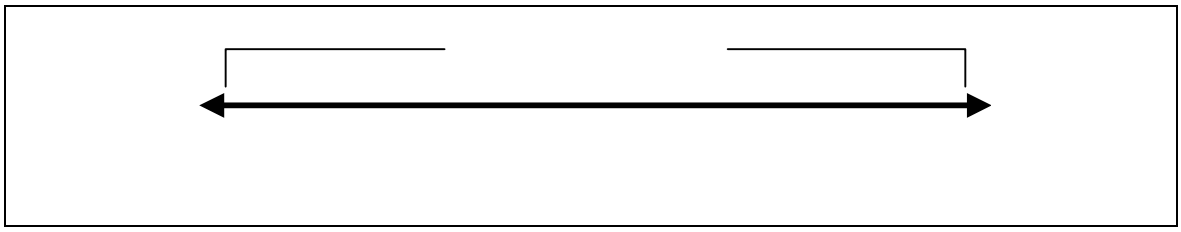

Figura 1 – Continuum de virtualidade (MILGRAM e KISHINO, 1994)

A RA pode ser classificada, de acordo com a Figura 1, como imagens e objetos gerados por computador, inseridos no mundo real que complementam o AV com informações que não são perceptíveis aos sentidos do usuário sem a ajuda de dispositivos especiais de interação.

Classificam-se como Virtualidade Aumentada os sistemas nos quais imagens e objetos reais são inseridos em um mundo predominantemente gerado por computador. Um exemplo de aplicação dessa tecnologia é a técnica de *Chroma-key* (Figura 2) utilizada no cinema e na televisão, em que um ator é posicionado à frente de um fundo verde ou azul e

este fundo é substituído por imagens ou vídeos utilizando programas específicos, fornecendo uma imagem combinada do ator com a imagem ou vídeo escolhidos.

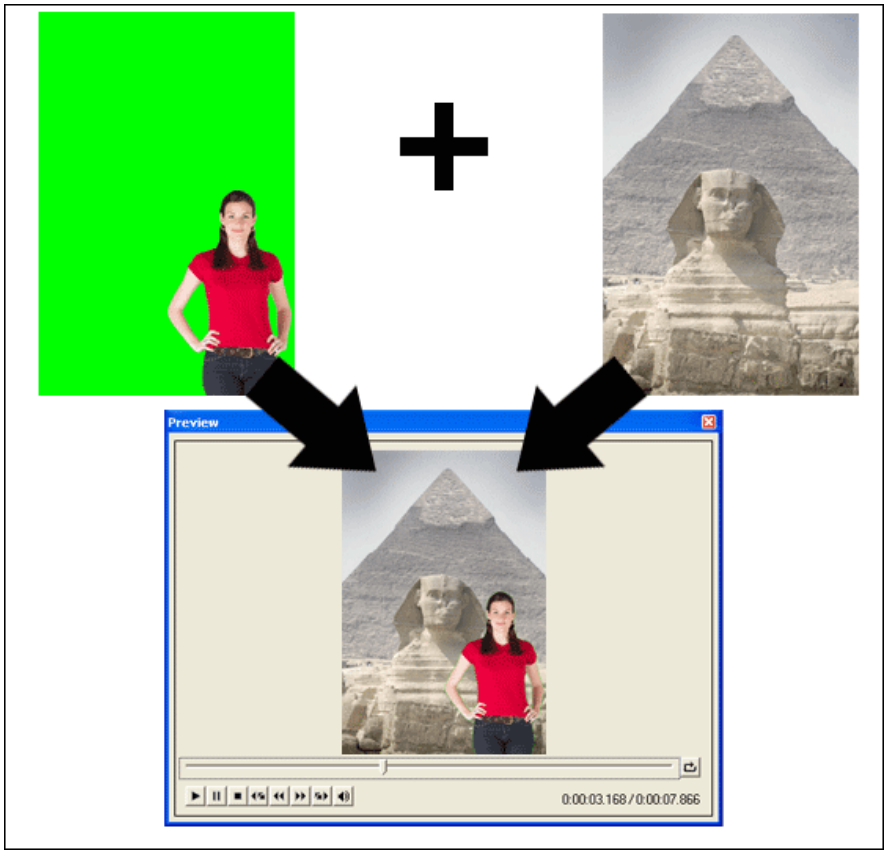


Figura 2 – Exemplo de software de Virtualidade Aumentada (DESKSHARE, 2007)

Nos dois extremos do *continuum* de Milgram e Kishino estão os ambientes virtuais e reais. No primeiro, o usuário tem a sensação de estar totalmente imerso em um ambiente sintetizado por computadores. O segundo é composto somente por objetos reais.

Kirner e Tori (2004) apresentam algumas definições da RA. Basicamente, RA, segundo a combinação dessas definições, seria um subconjunto da RM, com predominância do real sobre o virtual, em que este enriquece o ambiente real com informações, imagens e objetos virtuais, em tempo real, aparentando coexistirem e permitindo a interação. Os autores ainda apontam o grande impacto que essa tecnologia deverá trazer para o relacionamentos pessoais e para a “formalização de idéias, através de novas maneiras de visualizar, comunicar e interagir com pessoas e informações”, principalmente em áreas como ensino, aprendizagem e treinamento.

## 1.1 Definição de um sistema de RA

Há várias áreas nas quais se pode aplicar a RA, variando entre Medicina até treinamento militar. Porém, apesar dos diferentes domínios abrangidos, os sistemas de RA seguem basicamente um mesmo modelo (Figura 3): o ambiente real é capturado pela câmera de vídeo, a posição da câmera é estimada – geralmente por meio de rastreadores ou visão computacional – as coordenadas dos objetos gerados pelo computador são alinhadas com as da câmera e a imagem real incrementada com a imagem virtual é exibida. A utilização de uma *webcam* é apresentada como a maneira mais simples de realizar a captura das imagens reais para integrá-las com a imagem virtual (KIRNER e TORI, 2004).

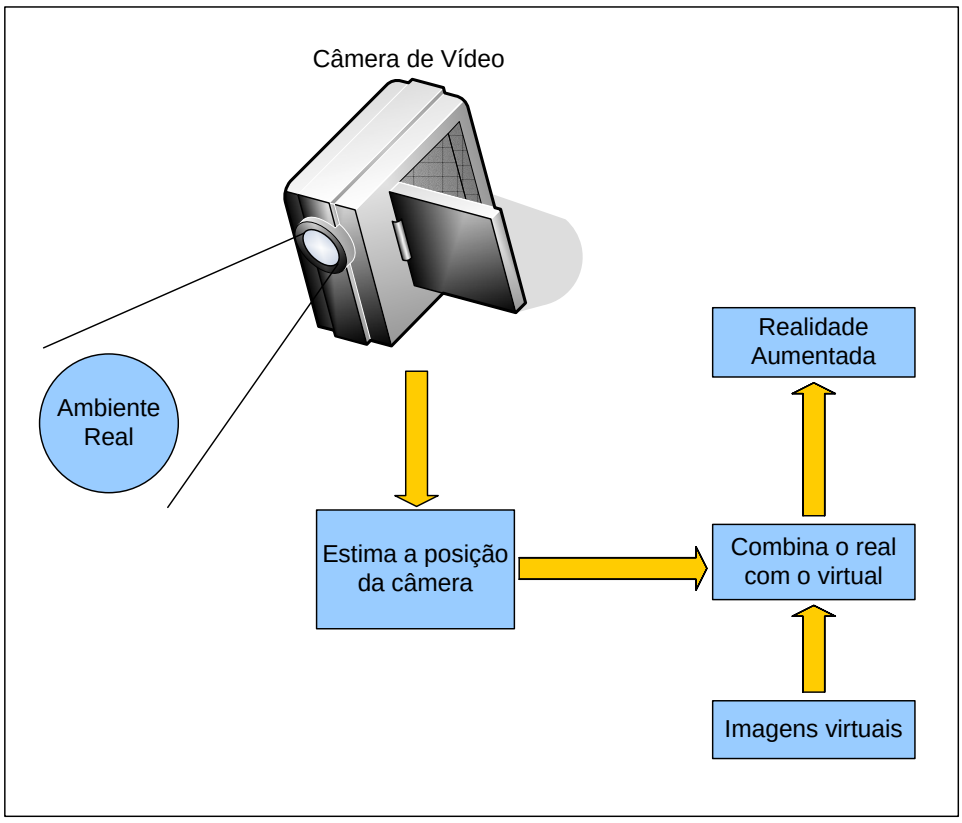


Figura 3 – Sistema de RA

Em Azuma (1997), os sistemas de RA são caracterizados segundo três aspectos básicos: combinação do real e do virtual, interação em tempo real e imagens 3D. Tais

características podem ser obtidas com o uso de câmeras de vídeo e *hardware* e *software* específicos.

A combinação de imagens reais com virtuais é o que faz com que os sistemas de RA sejam os mais imersivos, já que o ambiente em que o usuário se encontra é o real, e é impossível sentir-se mais imerso no mesmo (VALLINO, 2002). Esta sensação pode ser obtida por meio de três formas distintas de *displays*, que serão descritos mais adiante: *Head Mounted Displays* (HMD*)*, *Hand held* e projeção de imagens. Entretanto, esta característica apresenta um problema básico de registro das imagens virtuais no ambiente real, aspecto analisado na subseção 1.3.

Na maioria dos sistemas computacionais, respostas em tempo real são cruciais para garantir o interesse do usuário. Conforme citado no início deste capítulo, a RM visa convencer o usuário, fazendo com que este pense que objetos reais e virtuais coexistem em tempo real. Conforme Kirner e Tori (2004), “a meta de um sistema de RM é criar um ambiente tão realista que faça com que o usuário não perceba a diferença entre os elementos virtuais e reais participantes da cena, tratando-os como uma coisa só”.

Dentre as características que um sistema de RM deve ter para iludir o usuário, podemos citar o baixo tempo de resposta às ações realizadas pelo mesmo e a alta qualidade das imagens 3D. Segundo Watson (1997), o atraso máximo aceitável para respostas em um AV é de 100 ms. Caso ocorra um atraso maior do que 100 ms em relação a uma ação realizada pelo usuário – como, por exemplo, a mudança de posição de um determinado objeto – ele perderá a sensação de coexistência e poderá perder a motivação para continuar a utilizar a aplicação.

Em relação às imagens 3D, estas estão associadas à forma espacial com que a complementação virtual do ambiente é exibida. A tridimensionalidade pode ser obtida por meio de efeitos de perspectiva ou estereoscopia.

A caracterização de um sistema como sendo de RA, entretanto, ocorre quando é possível encontrar as três características abordadas por Azuma (1997): combinação do real e do virtual, interação me tempo real e imagens 3D. Se somente uma ou duas características são encontradas, o sistema não é classificado como RA. Um exemplo são os filmes que utilizam computação gráfica, a forma mais comum de combinação entre real e virtual, mas que não pode ser classificada como RA, por não ocorrem em tempo real.

## 1.2 Dispositivos utilizados em RA

É possível encontrar muitas definições de RV na literatura. Kirner e Tori (2004) a define como “uma interface avançada para aplicações computacionais, onde o usuário pode navegar e interagir, em tempo real, em um ambiente 3D gerado por computador, usando dispositivos multisensoriais”. Na mesma obra, os autores apresentam uma segunda definição de RV, classificada como a integração de imersão, interação e envolvimento. Como imersão entende-se a capacidade do sistema de RV de fazer o usuário se sentir dentro do AV. Interação é a capacidade do computador entender as ações do usuário e alterar o AV em tempo real. A idéia de envolvimento está relacionada com o engajamento do usuário na tarefa que está sendo realizada. Para que um sistema seja assim classificado, é necessária a utilização de equipamentos especiais para a interação.

Para Terashima (2001), a Interface Homem-Computador (IHC) tem como um de seus objetivos criar uma sensação de interação entre o homem e o computador similar à interação entre homem e homem por meio de dispositivos visuais, táteis, acústicos, entre outros. Azuma (2001) enfatiza que, apesar da visão ser o sentido mais importante para a percepção de um ambiente, a RA não faz uso somente de dispositivos visuais para fornecer a sensação de imersão e envolvimento ao usuário, mas também pode ser aplicada a todos os sentidos.

### 1.2.1 Dispositivos de saída visual

Há várias formas de visualização para RA. Azuma (2001) e Milgram et al. (1994) abordam a taxonomia dos *displays* de formas diferentes. Azuma (2001) os classifica como *HMD, handheld* e projeção de imagens. Milgram et al. (1994) classificam os dispositivos para RA em *See-through displays* e baseados em monitor. Neste trabalho, a apresentação dos dispositivos de saída visual será dividida em o*ptical see-through* HMD, v*ideo see-through* HMD, *handheld displays*, projeção, monitores e, em acréscimo, *virtual retinal displays* (VRD).

#### 1.2.1.1 *Optical see-through HMD*

Esta classe de dispositivos pode também ser chamada de “dispositivos ópticos”. Ao contrário dos HMD utilizados em RV, os HMD para RA devem permitir que o usuário tenha percepção do mundo real. No caso da tecnologia óptica, esta característica é obtida por meio de espelhos semitransparentes, também chamados de combinadores (AZUMA, 1997). Nestes combinadores, imagens virtuais são projetadas por telas de cristal líquido (TCL) ou telas de tubos de raios catódicos (TRC) e, por serem semitransparentes, permitem que o mundo real seja visto de forma enriquecida pelas informações adicionais. Essa capacidade permite que o grau de presença alcançado pela aplicação seja máximo (MILGRAM et al., 1994).

Entretanto, essa tecnologia apresenta alguns problemas. Por serem semitransparentes, os HMD ópticos impedem a passagem de determinada quantidade de luz do mundo real. Isto se torna um problema em ambientes escuros, pois estes dispositivos semitransparentes podem bloquear o pouco de luz existente no ambiente real, fazendo com que ele não apareça para o usuário.

Azuma (1997) apresenta um diagrama conceitual do *Optical See-through* em que um rastreador de cabeça (dispositivo que indica a posição do usuário) é utilizado (Figura 4).

Porém, os sistemas de RA que adotam essa tecnologia podem optar por outras tecnologias para melhorar a localização do usuário e exibir a imagem virtual no local desejado, como o ARToolkit (KATO e BILLINGHURST, 1999), com câmeras de vídeo.

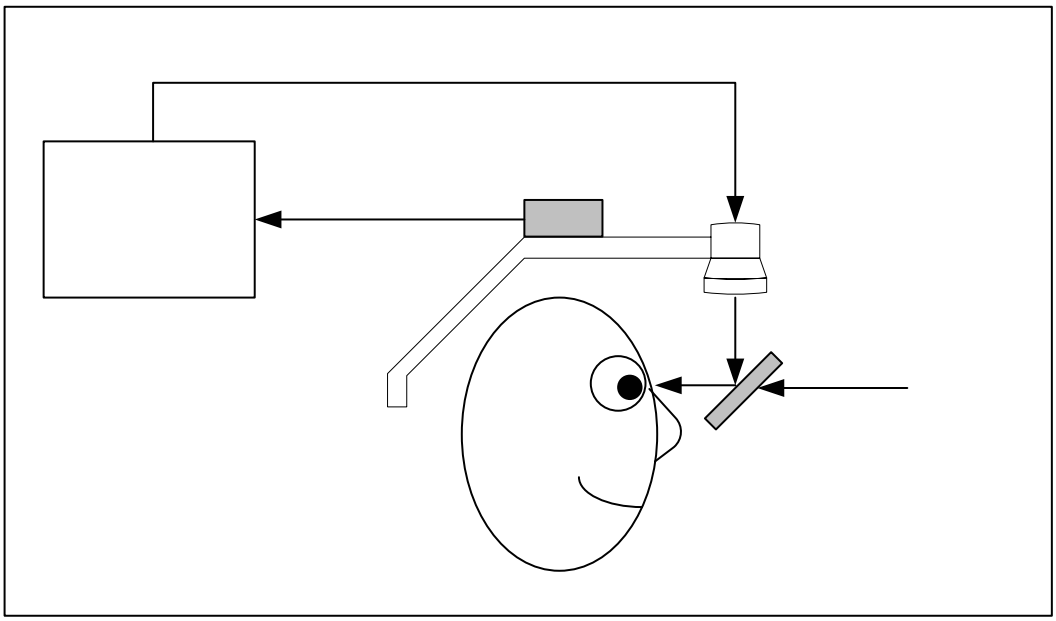

Figura 4 – Diagrama do Optical See-through (AZUMA, 1997)

A empresa Liteye (LITEYE, 2007) desenvolve capacetes ópticos para RA. Entre os produtos comercializados pela empresa está o LITEYE-500 (Figura 4). Apresentado em 2005, este *Optical See-through* HMD tem resolução de 800x600 *pixels* e 24 bits de cor.

Figura 5 – LITEYE-500 (LITEEYE, 2007)

Gerador de cena

#### 1.2.1.2 *Video see-through* HMD

Da mesma forma que o *Optical See-through*, o *Video See-through* deve permitir que o usuário veja o mundo real. Nesse tipo de dispositivo, as imagens do mundo real são capturadas por uma ou duas câmeras de vídeo, enviadas a um computador que combina as imagens virtuais e reais depois apresentadas ao usuário por meio de duas pequenas telas, que podem tanto ser TCL como TRC, fixadas em capacetes ou óculos e posicionadas diante dos olhos (AZUMA, 1997). Os capacetes e óculos utilizados são do tipo *Closed-view*, ou capacetes fechados, que permitem ao usuário ter visão do mundo real somente através das câmeras de vídeo.

Os óculos ARvision-3D (Figura 5), produzidos pela Trivisio (TRIOVISIO, 2007), possuem duas câmeras de foco manual e dois microdisplays de 1,44 megapixels de resolução.

Figura 6 – ARVision-3D (TRIOVISIO, 2007)

Os *Video See-Through* podem apresentar problemas de paralaxe, já que a imagem transmitida para o usuário é aquela do ponto de vista das câmeras de vídeo, que em muitas ocasiões são acima da cabeça do usuário. Esse problema, porém, pode ser resolvido posicionando a câmera na posição dos olhos do usuário ou com um conjunto de espelhos para criar um novo conjunto óptico que simule o conjunto óptico direto para os olhos do usuário.

### 1.2.1.3 *Handheld displays*

De acordo com a definição de Migram et al. (1994), os *displays* do tipo *handheld* podem ser considerados como baseados em monitores, já que utilizam pequenas telas de LCD que funcionam como uma “janela para o mundo”, ou *window-on-the-world* (WoW). Também é necessária a utilização de uma câmera de vídeo para captar as imagens do mundo real, em que as imagens virtuais são a elas sobrepostas.

A maior vantagem dessa tecnologia é a mobilidade que ela oferece devido à ausência de fios ou cabos. Em alguns sistemas de RA (MOGILEV, 2002) um *desktop* ou *laptop* é utilizado como servidor para a geração das imagens virtuais. No caso das aplicações que necessitam de mobilidade, isso se torna um empecilho, principalmente no caso dos *desktops*, já que um *laptop* pode ser preso ao corpo do usuário.

Mogilev (2002) apresenta uma interface para colaboração face-a-face, o ARPad (Figura 7), que constitui de uma tela LCD portátil, com um *mouse* do tipo *spaceball*, uma câmera de vídeo e um servidor para a geração das imagens virtuais.

Figura 7 – Visão frontal do ARPad (MOGILEV, 2002)

### 1.2.1.4 *Displays* de projeção

Segundo a classificação feita por Mackay (1998), a tecnologia de *displays* de projeção é considerada uma melhoria do ambiente real, onde o usuário e objeto não são alterados pela aplicação. Há várias formas de aplicar a tecnologia de projeção de imagens. Kirner e Tori (2004) citam três: tela panorâmica; mesa virtual e *Cave Automatic Virtual Environment* (CAVE). Para RA, os *displays* de projeção utilizam projetores ligados a supercomputadores, onde são geradas as imagens virtuais que podem ser projetadas sobre superfícies planas, como telas ou mesas, ou sobre objetos. Assim como os *displays* do tipo *handheld*, os *displays* de projeção, com exceção das CAVE, também podem ser classificados por Milgram (1994) como WoW.

Esse tipo de tecnologia pode dispensar a utilização de dispositivos especiais de visualização e proporciona ao usuário uma interação mais natural com o ambiente, já que o usuário pode se relacionar com os objetos virtuais como se fossem reais. Um exemplo de interação natural com o Ambiente Virtual é o *Interactive Paper* (Figura 8) em Mackay (1998), que utiliza a vantagem da flexibilidade do papel unida à manipulação da informação e à capacidade de comunicação do computador.


Figura 8 – Interactive Paper (MACKAY, 1998)

Azuma (2001) relata duas utilizações da projeção em RA. Em ambas, os objetos são recobertos com um material retroreflexivo que reflete a luz incidente exatamente na direção

de onde ela foi projetada. No primeiro exemplo citado pelo autor, o usuário utiliza um projetor acoplado em um capacete e, devido ao material retroreflexivo, as imagens originadas do seu projetor são refletidas somente para ele. Isso permite que este se movimente pelo ambiente e que o projetor acompanhe o seu ponto de vista, que não sofrerá intervenções dos projetores de outros usuários. Na segunda, as imagens projetadas nos objetos fazem com que eles pareçam semitransparentes e camuflados no ambiente real.

#### 1.2.1.5 Monitores

Classificado como WoW de acordo com Milgram (1994), os *displays* baseados em monitores utilizam uma câmera de vídeo para capturar a imagem do mundo real, que é então sobreposta pela camada de objetos virtuais gerada pelo computador e exibida ao usuário por um monitor. O usuário só poderá interagir com os elementos visíveis no monitor, já que ele não tem a liberdade oferecida pelos HMD.

Caso haja a necessidade de visualização estereoscópica, o usuário deve utilizar óculos estereoscópicos, também conhecidos como *shutter glasses*, compostos por lentes polarizadas ou de cristal líquido. Neste caso, o monitor exibe alternadamente as imagens para o olho direto e esquerdo, e os óculos estereoscópicos deverão estar sincronizados com o monitor para que no momento em que o monitor estiver exibindo a imagem para o olho direito os óculos bloqueiem a visão do olho esquerdo, e quando a imagem para o olho esquerdo estiver sendo exibida, bloqueiem a do olho direito.

O sistema UvA-DRIVE (Distributed *Real-time Interactive Virtual Environment*) (Figura 9), da Universidade de Amsterdã, é um exemplo de RV baseada em monitores. Nele, uma grande tela e óculos estereoscópicos são utilizados para exibir imagens 3D para um ou mais usuários simultaneamente (UVADRIVE, 2007).

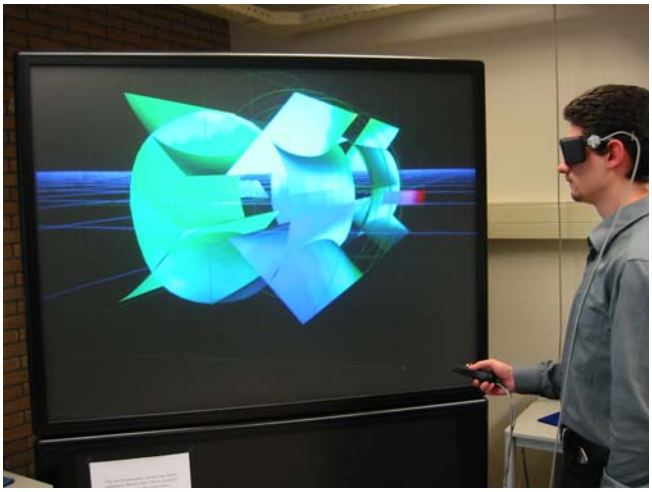

Figura 9 – Sistema UvA-DRIVE (UVADRIVE, 2007)

### 1.2.1.6 *Virtual retinal displays*

O *display* de retina começou a ser desenvolvido pelo Laboratório de Tecnologia em Interface Humana da Universidade de Washington, Estados Unidos, no início da década de 1990. Em Viirre (1998) ele é descrito como um dispositivo visual que realiza a rasterização com um feixe de laser direcional de baixa potência diretamente na retina do usuário. O *Virtual Retinal Display* (VRD) produz imagens com alto contraste, alta qualidade de brilho e alta resolução.

Empresas como a Microvision (MICROVISION, 2007) produzem comercialmente o VRD. Um dos displays disponíveis é o Nomad (Figura 10), que produz imagens monocromáticas na cor vermelha, com resolução de 800x600 *pixels* e tem suporte a rede sem fio.

Figura 10 – VRD Nomad (MICROVISION, 2007)

### 1.2.2 Outros dispositivos

Além dos dispositivos de saída visual, é possível citar os dispositivos de interação, os dispositivos físicos, os dispositivos auditivos e os dispositivos de rastreamento.

São classificados como dispositivos de interação qualquer dispositivo que permita a seleção e interação com objetos em cena e navegação pelo ambiente, entre eles, as luvas de dados (ou *data gloves*) , *mouse*, *trackball, spaceball,* bastões, entre outros. Já alguns sistemas de RA (MACKAY, 1998) utilizam os dedos do usuário para selecionar e manipular os dados.

Os dispositivos físicos “procuram estimular as sensações relacionadas ao tato, tensão muscular e temperatura” (CARDOSO e MACHADO, 2004). Infelizmente estes dispositivos não atingem o mesmo nível de realismo dos dispositivos visuais e auditivos. A divisão se faz em hápticos e térmicos. Os dispositivos hápticos estão relacionados ao tato e à força, possibilitando que o usuário obtenha respostas em relação a texturas e resistência de materiais. Já os dispositivos térmicos oferecem ao usuário características térmicas dos objetos.

Dispositivos auditivos têm por objetivo simular sons a fim de convencer a audição humana de que estes pareçam vir de determinada direção, em tempo real. Para isso é indispensável a utilização de placas específicas.

Os dispositivos de rastreamento permitem que se rastreie a trajetória exata dos objetos e pessoas contidos no AV e que se saiba sua posição. Estes dispositivos são divididos em quatro tipos: ópticos, magnéticos, acústicos e mecânicos. Essas quatro tecnologias serão descritas no Capítulo 2.

## 1.3 Registro

O Registro é apontado por Azuma (1997) como um dos mais básicos problemas na RA, e é descrito como o alinhamento correto entre o mundo virtual e real ou a ilusão de coexistência entre ambos. Como a informação visual se sobressai em relação às informações

providas pelos outros sentidos, erros de poucos *pixels* no registro podem fazer com que se perca a ilusão de coexistência.

Erros de registro são difíceis de controlar adequadamente devido à necessidade de grande exatidão e das inúmeras fontes de erros. Azuma (1997) classifica esses erros em estáticos e dinâmicos. Erros estáticos “causam erros mesmo quando o ponto de vista do usuário e o objeto em cena estão completamente parados”. Os erros dinâmicos só ocorrem quando o ponto de vista do usuário, ou o objeto, se move.

### 1.3.1 Erros estáticos

Azuma (1997) divide as fontes de erros estáticos em quatro tipos: distorção óptica, erros no sistema de rastreamento, alinhamento mecânico incorreto e parâmetros de visualização incorretos.

A distorção óptica ocorre na maioria das câmeras de vídeo e em conjuntos de lentes – tanto em câmeras que capturam as imagens do mundo real como nos conjuntos de lentes dos *displays* – principalmente quando o campo de visão utilizado é amplo. Por se tratarem de erros sistemáticos, as distorções ópticas podem ser compensadas e corrigidas por meio de lentes adicionais ou digitalmente por algoritmos. A utilização de lentes adicionais, entretanto, aumenta o peso dos equipamentos e os algoritmos aumentam o custo computacional.

Erros no sistema de rastreamento são considerados por Azuma (1997) como o tipo mais sério de erros estáticos. Por serem intrínsecos aos sistemas de rastreamento, a única forma de eliminá-los é adicionando outro sistema de rastreamento em conjunto, aumentando a exatidão dos dados obtidos.

O alinhamento mecânico incorreto diz respeito às diferenças entre o modelo ou especificação do *hardware* e as propriedades físicas do sistema real. Isso ocorre, por exemplo, quando combinadores, lentes e *displays* em um *optical see-through* HMD não estão nas distâncias ou orientações corretas entre si, ocasionando mudanças sutis na posição e

orientação das imagens projetadas. Alguns desses erros de alinhamento podem ser calibrados, outros só poderão ser anulados caso o dispositivo seja corretamente construído.

Parâmetros de visualização incorretos também podem ser vistos como um caso especial de erros de alinhamento em que se pode aplicar calibração. Parâmetros de visualização especificam como converter a localização informada da cabeça do usuário ou da câmera em matrizes de visualização requerida pelo gerador de cena para montar as imagens virtuais.

### 1.3.2 Erros dinâmicos

Os erros dinâmicos ocorrem devido ao atraso natural dos sistemas e dos dispositivos causados pelo tempo necessário para o processamento dos dados. Azuma (1997) define os atrasos ponto-a-ponto como “a diferença de tempo entre o momento que o sistema de rastreamento mede a posição e orientação do ponto de vista do usuário até o momento em que a imagem correspondente a esta posição e orientação aparece no *display*”. Normalmente, quando há movimento, o atraso é de 100 ms. Entretanto, atrasos maiores que 250 ms podem ocorrer em sistemas mais lentos, prejudicando consideravelmente a interatividade da aplicação.

Os métodos utilizados para reduzir o atraso são divididos em quatro categorias: reduzir o atraso do sistema, reduzir o atraso aparente, igualar o fluxo temporal e prever as localizações futuras.

### 1.3.3 Visão computacional

Ao contrário das técnicas de registro que utilizam somente os dados obtidos dos rastreadores de cabeça, as técnicas baseadas em visão empregam técnicas de visão computacional para auxiliar no registro.

Em alguns dos sistemas de RA que aplicam estas técnicas, são utilizados marcadores fixos – que podem ser marcadores especiais, *Light Emitting Diode* (LED), pontos coloridos, esferas brancas, entre outros. Estes marcadores têm por finalidade indicar pontos específicos do mundo real que devem ser rastreados pelos sistemas de RA. A localização destes marcadores é detectada por processamento de imagem e então utilizada para correções que reforçam o registro correto.

Outra técnica referida por Azuma (1997) é a utilização de comparação de padrões para melhorar o registro, em que imagens dos objetos reais são capturadas em diversos pontos de vista e, então, utilizadas para buscar a imagem virtual correta para o objeto real.

## 1.4 Aplicações da Realidade Aumentada

A RA pode ser útil em diversas áreas, do treinamento militar à Medicina. Pode-se citar como uma das causas dessa versatilidade a capacidade da RA em agregar informações ao mundo real que não seria percebida simplesmente por nossos sentidos. Outro fator importante é que, ao contrário da RV, a RA permite adicionar as informações virtuais sem desvinculá-los visualmente do ambiente principal, no caso o real, tornando a tarefa mais amistosa ao ser humano. Algumas das áreas de aplicação de RA serão listadas a seguir.

### 1.4.1 Medicina

A medicina é considerada uma das mais importantes para a RA, já que a utilização de imagens é muito difundida nessa área (VALLINO, 2002), não necessitando grandes adaptações por parte de seus profissionais. As áreas que recebem mais atenção na literatura são as cirurgias guiadas por imagens, treinamento e exames como ultra-sonografia.

Atualmente, nas cirurgias realizadas, os estudos pré-operatórios são realizados sobre imagens de tomografia computadorizada (TC) ou imagens de ressonância magnética (IRM). Com a RA essas imagens podem ser sobrepostas ao paciente na mesa de cirurgia enquanto o

procedimento cirúrgico é realizado, melhorando a performance da equipe cirúrgica e, no caso de neurocirurgia, eliminando a necessidade de dolorosas molduras estereotácticas. A Figura 11 apresenta um exemplo de moldura estereotáctica para neurocirurgia (MBCHB, 2007) e um exemplo de neurocirurgia com RA (WATERWORTH, 1998). A RA permite a realização de cirurgias minimamente invasivas, já que possibilita uma visão interna do paciente sem a necessidade de grandes cortes.

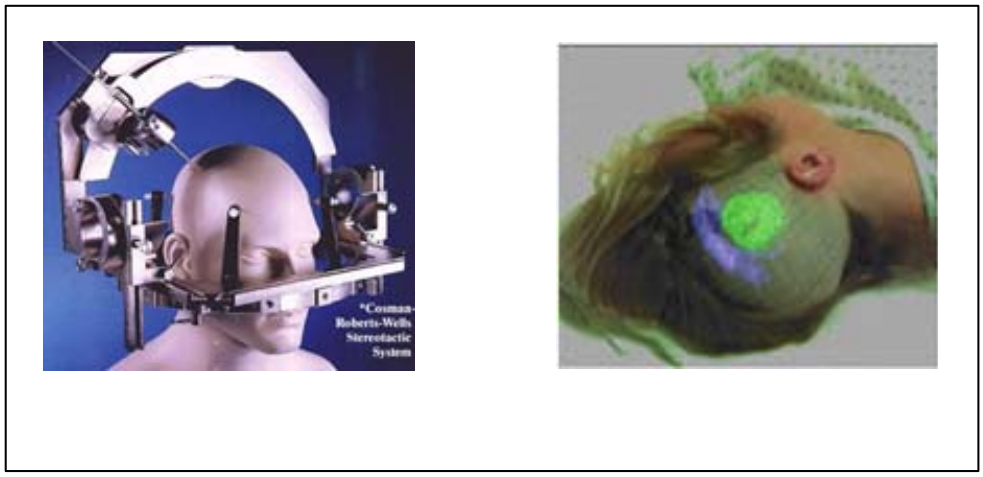


Figura 11 – Neurocirurgia sem a aplicação da RA e com a aplicação da RA

Uma das aplicações mais conhecidas da RA na Medicina é a obstetrícia, com imagens de ultra-sonografia sobrepostas ao corpo da paciente. Com a utilização de um *display* do tipo *optical see-through* o profissional pode visualizar a imagem do feto 3D sobreposta ao abdome da mãe.

A RA também é útil para o treinamento médico. Instruções virtuais podem ser mostradas aos cirurgiões novatos durante determinadas etapas da cirurgia, sem a necessidade de desviar a atenção no paciente para consultar livros.

Outra aplicação de RA na medicina, mais especificamente na ortopedia e fisioterapia, é na captura e análise de movimentos de pacientes (GREENLEAF, 2001). Nesta situação, os movimentos de um paciente são capturados por uma câmera de vídeo e, através de técnicas de visão computacional, o movimento é transformado em dados computacionais para que o

médico o analise. Esses dados podem ser armazenados para que futuramente o ortopedista possa acompanhar a evolução do tratamento.

### 1.4.2 Manufatura e reparo

Manutenção e reparo são duas outras áreas em que a RA é aplicada. Um técnico, ao realizar a manutenção em um equipamento desconhecido, pode utilizar um *display* de RA para aumentar a imagem do equipamento com anotações e informações necessárias para se realizar as etapas da manutenção. Pode-se destacar, por exemplo, as partes do hardware que deverão ser removidas para a realização do reparo, e então a visão interna da máquina pode destacar as placas que deverão ser trocadas (VALLINO, 2002), pois segundo Azuma (1997), instruções são mais fáceis de entender se estiverem disponíveis em forma de textos 3D diante do equipamento que estiver sendo manufaturado ou reparado, ao invés de apresentadas em manuais com textos e figuras.

Um exemplo da utilização de RA na manutenção de equipamentos é o projeto KARMA (*Knowledge-based Augmented Reality for Maintenance Assistance*) desenvolvido pelo Laboratório de Computação Gráfica e Interfaces com o Usuário da Universidade de Columbia, em Nova Iorque, Estados Unidos. KARMA (2006) é um protótipo de um sistema que utiliza *see-through* HMD para explicar simples tarefas de manutenção de impressoras a *laser* (Figura 12). São utilizados pequenos marcadores triangulares como rastreadores, permitindo que o sistema realize a localização da posição e orientação da impressora. Depois de calculados esses dados, o sistema exibe interativamente dicas textuais simples.

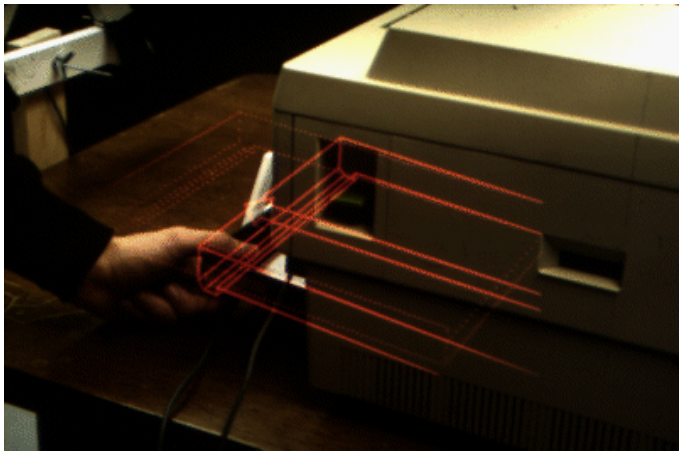

Figura 12 – Projeto KARMA (KARMA, 2006)

### 1.4.3 Anotação e visualização

Outra aplicação da RA é na anotação e visualização de objetos e ambientes públicos ou privados. O usuário portaria um display do tipo *handheld* ou um *optical see-through* HMD para visualizar dicas como, por exemplo, o ARLib (UMLAUF et al., 2002), que permite localizar determinado livro em uma biblioteca com o auxílio de um *notebook*, um HMD com uma câmera de vídeo e o software Studierstube (STUDIERSTUBE, 2006). Esta aplicação tem o objetivo de enriquecer o ambiente real com informações sobrepostas às estantes da biblioteca. O livro desejado é destacado dos demais quando ele aparece no campo de visão do usuário.

Outra aplicação de anotação que utiliza RA, descrita por Mackay (1998), é o *Caméléon* (Figura 13), que utiliza a RA para o controle de vôo, em que os controladores anotam os dados necessários para rastrear os aviões em pedaços de papel. Ao invés de eliminar as tiras de papel, a autora propõe enriquecê-las com RA, já que substituí-las completamente por informações digitais traria mais malefícios do que benefícios aos trabalhadores dos aeroportos, como o aumento da poluição visual, entre outros.

Figura 13 – Projeto Caméléon (MACKAY, 1998)

### 1.4.4 Robótica

Na robótica, a RA pode ser aplicada no auxílio da movimentação de robôs à distância. Caso seja utilizada uma única câmera de vídeo para capturar a imagem do robô real, o sentimento de profundidade é perdido, podendo ocasionar acidentes ou erros. Com o auxílio da RA, porém, o usuário pode planejar e executar os passos em um robô virtual sobreposto à imagem do robô real e caso a representação virtual consiga executar a tarefa, o comando pode ser dado ao robô real, evitando ocorrência de erros, além de eliminar possíveis oscilações devido ao atraso da conexão.

A Figura 14 mostra um exemplo de aplicação da RA na área da robótica citada por Azuma (1997). Nela, o braço robótico, representado com as linhas vermelhas, é reconstruído por meio de computação gráfica e sobreposto ao braço real.

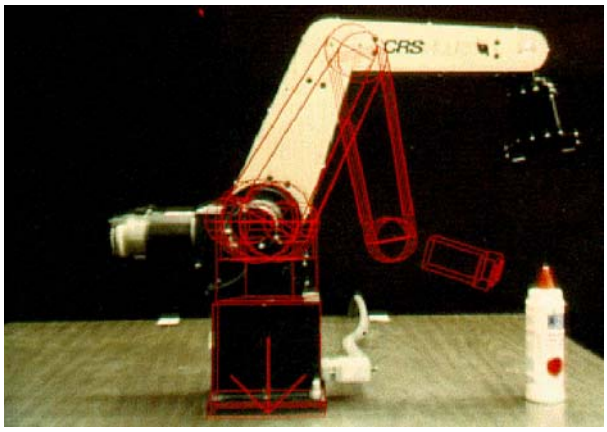

Figura 14 – Aplicação de RA na robótica (AZUMA, 1997)

### 1.4.5 Entretenimento

O exemplo mais comum de RA na área do entretenimento é o *Chroma-key*, descrito no início deste Capítulo. A utilização de cenários virtuais 3D diminui os custos de produção, pois são mais baratos que cenários reais, além de exigirem menos espaço para armazenamento.

A utilização da RA também se aplica a jogos, como o ARQuake (PIEKARSKI e THOMAS, 2002), uma versão aumentada do jogo Quake. Entretanto, o jogador pode se mover livremente pelo ambiente real, permitindo que o usuário interaja de forma natural.

Mackay (1998) apresenta o *Video Mosaic* (Figura 15), que permite a criação de *storyboards* e a edição de vídeos sobre uma mesa contendo câmeras – para capturar as anotações escritas pelo usuário – e um projetor – que mostra os vídeos e os *storyboards*.

Figura 15 – Video Mosaic (MACKAY, 1998)

### 1.4.6 Militar

Na área militar, a RA pode ser aplicada em treinamentos e combates. Em um avião ou simulador de vôo, é possível projetar dados sobre inimigos ou sobre o próprio avião em um *Head-up Display* (HUD), que é uma tela transparente posicionada em frente ao piloto, oferecendo uma visão aumentada do mundo real. A Figura 16 mostra um exemplo de HUD desenvolvido pela *Elbit Systems* (ELOP, 2007).

Figura 16 – Head-up Display (ELOP, 2007)

A RA pode também ser aplicada no treinamento de soldados, utilizam um HMD semitransparente onde são projetados dados sobre seus inimigos. Pode-se também adicionar veículos militares durante a simulação, deixando o campo de batalha mais real.

## 1.5 O ARToolkit

O ARToolkit é um conjunto de bibliotecas desenvolvidas na linguagem C pelo Laboratório de Interface Humana da Universidade de Washington que tem como objetivo auxiliar o programador na construção rápida de aplicações na área de RA. Disponível gratuitamente para fins não comerciais, o ARToolkit está atualmente na versão 2.72.1 e é disponível para Windows (95, 2000, XP), Linux, IRIX (da *Silicon Graphics Inc.*, ou SGI) e MacOS. O pacote disponibilizado pela Universidade de Washington contém as bibliotecas para rastreamento e os códigos-fonte destas bibliotecas para permitir que programadores adaptem o ARToolkit para as suas necessidades (ARTOOLKIT, 2006).

Conforme citado anteriormente, registro é um dos grandes problemas no desenvolvimento de aplicação de RA (AZUMA, 1997) (ARTOOLKIT, 2006). O ARToolkit utiliza técnicas de visão computacional para calcular o ponto de vista da câmera em relação a marcadores quadrados que contêm um padrão na parte interna (Figura 17), possibilitando o registro correto das imagens.


Figura 17 – Exemplo de marcador que acompanha o ARToolkit

Os marcadores do ARToolkit são compostos por quadrados que podem ser impressos em impressoras comuns. No centro do quadrado deve haver uma imagem não simétrica e de alto contraste.

Nas subseções seguir serão descritos o funcionamento do ARToolkit e algumas aplicações que utilizam a biblioteca.

### 1.5.1 O funcionamento do ARToolkit

O funcionamento do ARToolkit pode ser descrito em seis etapas, conforme descrito em (ARTOOLKIT, 2006), realizadas durante toda a execução do programa e em todos os *frames*, como mostra a Figura 18:

1. a câmera de vídeo captura as imagens do mundo real e as envia ao computador;
2. busca por marcadores;
3. encontrar a posição e orientação 3D do marcador;
4. identifica os marcadores;
5. posiciona o objeto virtual na posição e orientação desejadas e,
6. sobrepõe a imagem real com o objeto virtual

Na etapa 2, a imagem capturada pela câmera de vídeo é binarizada, ou seja, os *pixels* são configurados como preto caso o seu valor esteja abaixo de um determinado limiar pré-estabelecido, ou em branco caso o valor esteja acima deste limiar. Em seguida, o ARToolkit procura e extrai regiões que podem ser ajustadas dentro de quatro linhas, como quadrados.

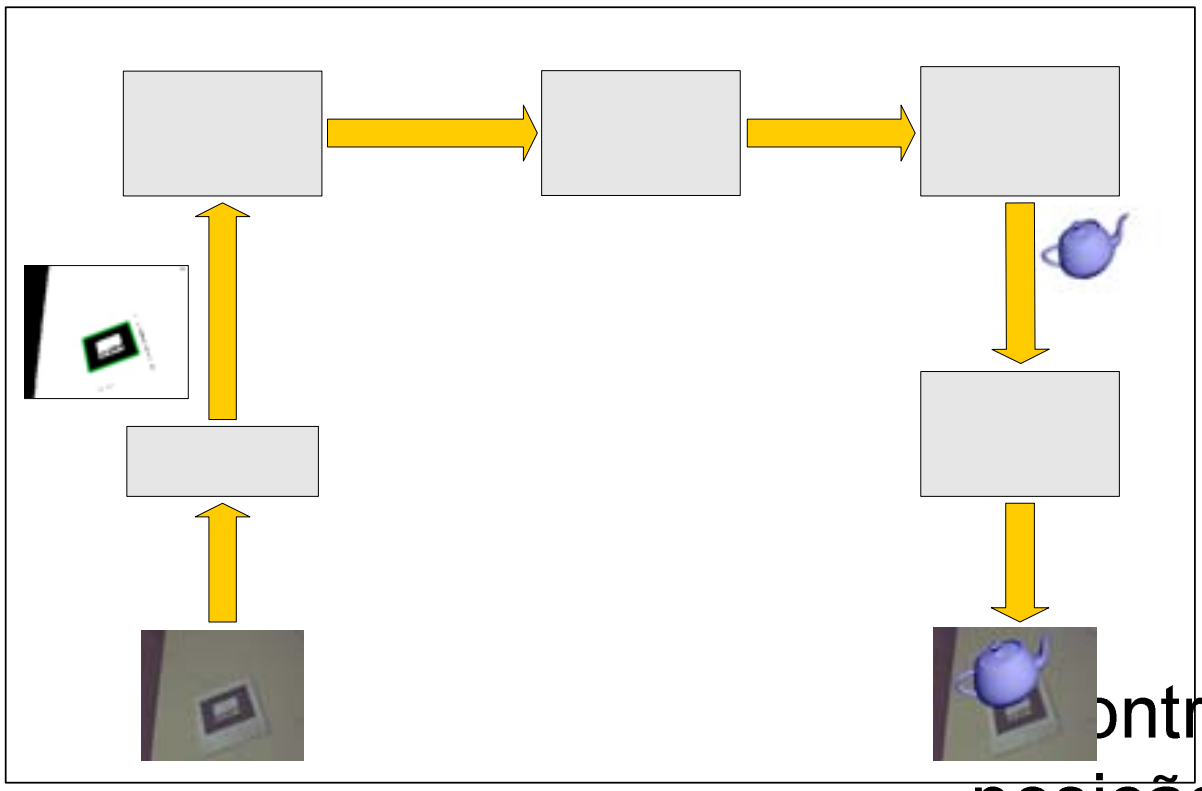


Figura 18 – As etapas de funcionamento do ARToolkit

Na terceira fase são calculadas a posição e a orientação dos marcadores encontrados na segunda etapa, em relação à câmera de vídeo. É obtida uma matriz que conterá a posição da câmera em relação ao marcador detectado. A Figura 19 representa esta matriz, sendo que a matriz $R_{3x3}$ contém os dados da orientação e a coluna $T_{1x3}$ os valores do posicionamento nos eixos X, Y e Z.

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \end{bmatrix}$$

Figura 19 – Matriz de transformação fornecida pelo ARToolkit

Na etapa seguinte, as regiões localizadas são normalizadas e as imagens contidas no centro dos marcadores encontrados são comparadas a padrões armazenados previamente no sistema.

Na fase 5, a matriz de transformação obtida na etapa 3 é utilizada para alinhar o objeto real com o seu respectivo marcador.

Por fim, a imagem real é sobreposta pela virtual e exibida no dispositivo de saída.

### 1.5.2 Limitações do ARToolkit

A exemplo da maioria dos sistemas óptico de rastreamento, o ARToolkit apresenta falhas relacionadas à oclusão, ou seja, caso o marcador não seja completamente visível para a câmera de vídeo, não é possível localizá-lo e exibir a imagem virtual relacionada a ele. Ledermann et al. (2002) listam três situações que explicam a maioria das falhas no rastreamento: más condições de iluminação, marcador não completamente visível e marcador muito pequeno ou muito distante da câmera.

Más condições de iluminação podem ocorrer tanto pelo excesso de luz quanto pela falta. Para se obter resultados ótimos, Ledermann et al. (2002) recomendam a utilização de luz branca difusa e de intensidade constante por toda a área que será rastreada. Caso não haja luz suficiente, os marcadores não serão reconhecidos pelo ARToolkit. Se a luz for muito intensa e direta sobre o marcador, as partes pretas refletirão a luz e este não será reconhecido.

Outra limitação é quando o marcador não está completamente visível. Uma única parte do marcador obstruída ou fora do ângulo de visão da câmera já é suficiente para ocorrer falha de rastreamento. Este problema pode ser solucionado adicionando-se vários marcadores em objetos fixos e informando ao sistema a distância entre eles. Assim, quando um marcador não estiver visível, o ARToolkit utilizará a posição de outro marcador para exibir o objeto virtual.

A última situação que ocorre a oclusão é quando o marcador é muito pequeno ou está muito distante da câmera. Isto acontece quando a imagem do marcador não preenche o número mínimo de *pixels* para que possa ser reconhecido pelas funções do ARToolkit. A solução é utilizar marcadores de tamanhos apropriados para cada situação, além de câmeras com sensores de alta resolução.

### 1.5.3 Aplicações

Atualmente, encontram-se diversas aplicações de RA que utilizam a biblioteca ARToolkit. Kato e Billinghurst (1999) apresentam um sistema de conferência que exibe as imagens captadas pelas câmeras dos outros participantes em monitores virtuais sobrepostos aos marcadores do ARToolkit sendo possível, assim, posicioná-los onde melhor convier para o usuário. O usuário pode ver e interagir de forma colaborativa com objetos virtuais utilizando um quadro branco virtual. A Figura 20 mostra os monitores virtuais e o quadro branco.

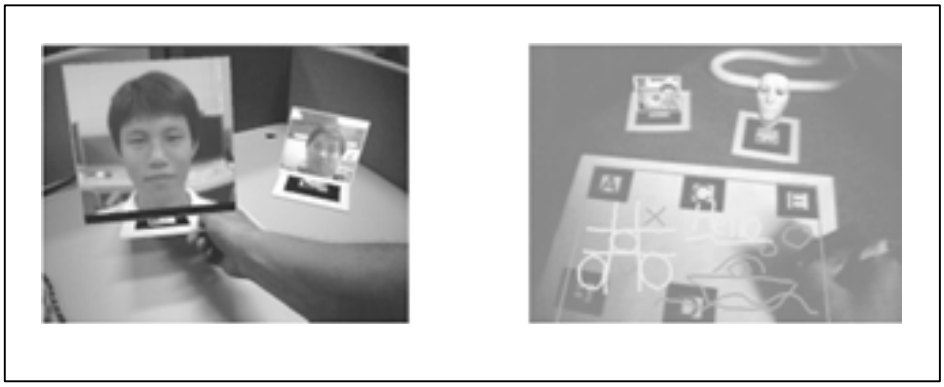

Figura 20 – Sistema de conferência com a ARToolkit (KATO e BILLINGHURST, 1999)

Liarokapisa e Whiteb (2005) apresentam um sistema de RA para museus e exposições, em que objetos danificados ou incompletos têm suas partes recompostas pela sobreposição de imagens reais por imagens virtuais, além de possibilitar a complementação do ambiente real com informações sobre os objetos expostos. A Figura 21 mostra a imagem de uma caneca com a asa quebrada recomposta pela RA e com informações sobre o objeto adicionadas.


Figura 21 – Sistema de auxílio a museus (LIAROKAPISA e WHITEB, 2005)

Sementille et al. (2004) apresentam um sistema de rastreamento baseado na detecção de marcadores do ARToolkit e em uma única câmera de vídeo para determinar a posição e orientação das juntas do corpo humano. Os dados obtidos podem ser utilizados para diversos fins, tais como entretenimento – na forma de jogos ou filmes – e medicina para avaliar resultados de tratamentos fisioterápicos. A Figura 22 mostra o posicionamento dos marcadores no corpo humano.

Figura 22 – Posicionamento dos marcadores no corpo humano (SEMENTILLE et al., 2004)

## 1.6 Considerações finais

Devido ao fato de a visão ser o sentido mais importante para a percepção de um ambiente, a maioria dos dispositivos existentes hoje para a RM é visual. Entretanto, outras formas de interação com o usuário devem ser utilizadas para permitir ao usuário a sensação de imersão e envolvimento.

Mesmo apresentando alguns problemas de registro, a RA tem demonstrado grande versatilidade, visto que pode ser aplicada em áreas completamente diferentes. Na maioria das vezes, porém, sempre com o objetivo de adicionar informações ao mundo real.

A biblioteca ARToolkit, que utiliza reconhecimento de padrões e visão computacional para complementar o mundo real com dados virtuais, também tem demonstrado ter vasta aplicabilidade em áreas como Medicina, Entretenimento e Anotação e Visualização.

# CAPÍTULO 2 - CAPTURA DE MOVIMENTOS HUMANOS

A captura de movimentos (CM) pode ser definida como a utilização de *software* e *hardware* que recebem como entrada movimentos realizados por um ator e os transformam em dados computacionais para serem utilizados em outras aplicações. Silva (2002) define sistemas de CM como sistemas "que geram para o computador informações que representam as medidas físicas do movimento capturado".

Há diversas aplicações para os dados capturados, tanto na RV quanto na RA. Silva (2002) e Welch e Foxlin (2002) citam alguns exemplos de aplicação desta tecnologia, tais como cinema e televisão – para produzir animações mais realísticas – e a captura dos movimentos das mãos – para sistemas de RV interativos.

De uma forma geral, para realizar o processo de CM humanos, o ator deve vestir uma roupa especial, composta por marcadores ou transmissores (dependendo da tecnologia utilizada) posicionados no corpo humano de forma a representar as principais articulações. Após vestir a roupa, o ator realiza os movimentos necessários que são captados por sensores e enviados para um computador onde serão processados e otimizados para gerarem os dados relativos aos movimentos executados.

## 2.1 Tipos de sistema de Captura de Movimentos

Os quatro tipos sistemas de CM mais difundidos, e que serão descritos neste trabalho são: acústicos, magnéticos, mecânicos e ópticos, sendo os dois últimos os mais empregados. Além destes, porém, há os sistemas de CM inercial, por ondas de rádio, entre outros.

### 2.1.1 Sistemas ópticos

Os sistemas ópticos podem ser divididos em dois componentes: fontes de luz e sensores ópticos. As fontes de luz podem ser emissores de luz ou refletores, e os sensores ópticos são câmeras de vídeo, que podem ser analógicas ou digitais.

O ator veste a roupa especial já citada no início deste Capítulo (Figura 23) e nela são posicionados os marcadores. No caso de sistemas de CM com marcadores passivos, podem ser utilizados refletores, esferas coloridas, marcadores especiais (como os da biblioteca ARToolkit, citados no Capítulo 1) ou esferas prateadas. No caso de sistemas com marcadores ativos, são utilizados LEDs, *lasers* ou lâmpadas comuns para proverem as informações. A área a ser rastreada é iluminada de forma difusa para que diminuam as sombras duras e não ocorram reflexos indesejados. As câmeras de vídeo são posicionadas em volta da área a ser rastreada e as posições dos marcadores são capturadas e enviadas ao computador para serem processadas.

Figura 23 – Roupa com marcadores passivos (SILVA, 2002)

As vantagens da utilização dos sistemas ópticos em relação aos outros tipos são: a alta taxa de amostragem, que permite a captura de movimentos rápidos, marcadores pequenos, liberdade durante os movimentos, pois não há fios limitando o ator, permite a utilização de

amplas áreas de trabalho, já que o único limitador é o campo de visão das câmeras e permite a utilização de um número ilimitado de marcadores, aumentando a precisão dos dados.

As desvantagens dos sistemas ópticos são: oclusão, visto que, se o marcador não estiver visível à câmera, a sua posição não poderá ser rastreada a necessidade de processamento dos dados obtidos, o que não permite a interatividade durante a captura e o alto preço, que pode chegar a até US$ 250.000,00 (SILVA, 2002).

### 2.1.2 Sistemas mecânicos

Ao invés de uma roupa especial, os sistemas de CM mecânicos utilizam um exoesqueleto (Figura 24) composto por potenciômetros entre duas partes rígidas. Conforme o ator se movimenta, os potenciômetros são acionados e as informações sobre as rotações realizadas são transformadas em dados computacionais.

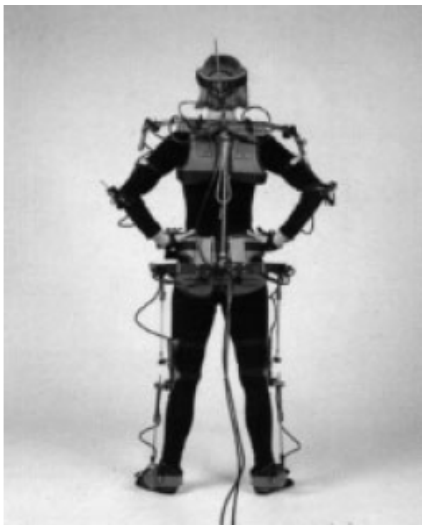

Figura 24 – Exoesqueleto do software de captura Animatton (SILVA, 2002)

Os sistemas mecânicos são considerados os sistemas de CM mais simples e os mais precisos. Outras vantagens apresentadas por este tipo de captura são: alta taxa de amostragem, preço baixo e não são afetados por interferências.

Entre as desvantagens dos sistemas de CM mecânicos, é possível listar: são extremamente obstrutivos devido ao exoesqueleto, pequeno espaço de trabalho devido aos fios e gera muito desconforto, caso o exoesqueleto seja grande e pesado.

### 2.1.3 Sistemas magnéticos

Nos sistemas magnéticos, um conjunto de receptores é posicionado nas articulações do ator e transmissores são posicionados em volta do espaço que será rastreado. Estes transmissores emitem pulsos eletromagnéticos que são captados pelos receptores. Com as informações da distância do receptor aos transmissores, é possível calcular a posição das articulações. A Figura 25 mostra o conjunto de receptores do sistema Motion Star da Ascension Technology (ASCENSION, 2007).


Figura 25 – Receptores magnéticos do Motion Star

Algumas vantagens dos sistemas de CM magnéticos são: baixo custo computacional, não ocorrência de oclusão e menor preço.

A maior desvantagem dos sistemas magnéticos é a grande quantidade de cabos que restringem o movimento do ator. Outra grande desvantagem é a suscetibilidade às interferências causadas por objetos metálicos.

### 2.1.4 Sistemas acústicos

Os sistemas acústicos utilizam a transmissão e recepção de ondas sonoras, calculando o tempo decorrido da emissão até a recepção do som. Os emissores são colocados nas principais articulações do ator e três receptores são posicionados no espaço monitorado.

Um som característico é emitido seqüencialmente por cada emissor e, quando captado pelos receptores, possibilitam o cálculo da posição das juntas por triangulação das distâncias.

Silva (2002) aponta a “dificuldade de se obter uma descrição correta dos dados num instante desejado devido ao caráter seqüencial dos disparos” como um dos problemas deste tipo de sistema. Outras desvantagens são: a grande quantidade de cabos, a limitação no número de transmissores que podem ser utilizados simultaneamente, a reflexão do som emitido e a interferência dos sons externos.

As únicas vantagens deste tipo de sistema apresentadas por Silva (2002), é o fato de não existir problemas de oclusão ou interferência por objetos metálicos. A análise das vantagens e desvantagens dos sistemas acústicos faz com que estes sejam a última opção dos sistemas de CM.

## 2.2 Classificação dos sistemas de CM

Além de classificar os sistemas de CM pela tecnologia aplicada, Silva (2002) apresenta outras formas de se caracterizar um sistema de CM. Para isso, o autor apresenta o conjunto de medidas proposto por Kalawsky (1993):

**Medidas Descritivas:**

- Processo de captura utilizado (acústico, magnético, mecânico, óptico, entre outros)
- Localização dos sensores e marcadores (sensores e marcadores localizados no corpo do ator ou fora dele)
- Parte do corpo processada (corpo inteiro, mãos, tronco e membros superiores, etc.)

**Medidas Estáticas:**

- Resolução espacial
- Precisão estática
- Linearidade e calibragem

**Medidas Dinâmicas:**

- Resolução temporal e alcance da freqüência
- Precisão dinâmica
- Taxa de amostragem, exatidão temporal e latência

**Medidas de Precisão:**

- Estabilidade
- Ruído, filtragem, suavidade e interferência externa (materiais metálicos, oclusão, etc.)

**Medidas de Interface:**

- Interface com o sensor (câmera, transmissores, etc.)
- Suprimento de energia
- Interface com o usuário e modos de operação
- Interface de comunicação com o host
- Conexão com o ator (com ou sem cabos)

**Medidas Computacionais:**

- Dados de referência (coordenadas locais ou globais)
- Formato dos dados (tipo de arquivo utilizado)
- Representação dos dados (tipo de dados)

**Medidas Operacionais:**

- Fator de forma (tamanho e peso dos marcadores, sensores e cabos)
- Espaço de trabalho
- Conforto e obstrução
- Tempo de preparação para a captura (calibragem)
- Compatibilidade com outros sistemas
- Ambiente de operação (umidade, temperatura, etc.)

Os sistemas de CM poderiam ser classificados de acordo com qualquer medida descrita por Kalawsky, entretanto, será abordada somente a classificação pela localização dos sensores e marcadores proposta por Mulder e descrita por Silva (2002). Outra característica que também será descrita é a forma de aquisição dos dados, que é abordada em Silva (2002), mas não é citada na lista anterior.

### 2.2.1 Localização dos sensores e marcadores

Há basicamente três categorias para esta classificação: *inside-in*, *inside-out* e *outside-in*. Todavia, estas classificações não são fixas e podem ser mescladas de acordo com a necessidade do usuário (WELCH e FOXLIN, 2002).

#### 2.2.1.1 Sistemas *inside-in*

Os sistemas de CM mecânicos são exemplos de sistemas *inside-in*, que se caracterizam por terem os sensores e as fontes transmissoras localizadas no corpo do ator, ou seja, não há nenhum sensor ou fonte no local onde é feita a captura. Os sistemas *inside-in* são considerados extremamente obstrusivos, pois o ator usa um exoesqueleto que limita os seus movimentos. São recomendados para a CM de parte pequenas do corpo humano, como dedos e olhos, pois para grandes partes pode ocorrer perda na precisão dos dados obtidos.

#### 2.2.1.2 Sistemas *inside-out*

Diferentemente dos sistemas *inside-in*, os sistemas *inside-out* possuem as fontes emissoras dos sinais fora do corpo do ator, porém, os sensores continuam nele. Os sistemas magnéticos são classificados como *inside-out*. Eles se caracterizam pelo espaço de trabalho e precisão limitados e são utilizados para rastrear partes maiores do corpo, tais como braço, parte superior e corpo inteiro.

### 2.2.1.3 Sistemas *outside-in*

Nos sistemas *outside-in* as fontes emissoras estão localizadas no corpo do ator, enquanto os sensores estão no local onde é realizada a captura. Os sistemas ópticos e acústicos são classificados nesta categoria, e as fontes emissoras são os marcadores - tanto ativos quanto passivos. Apesar de altamente oclusivos e de possuírem um espaço de trabalho limitado, estes sistemas têm o menor grau de obstrução e são utilizados para capturar movimentos de partes grandes e pequenas do corpo humano.

### 2.2.1.4 Comparação entre as três categorias

A Tabela 1 (MULDER, 1994) mostra a comparação entre as três categorias.

Tabela 1 – Comparação entre sistemas *inside-in*, *inside-out* e *outside-in*

| | ***Inside-in*** | ***Inside-out*** | ***Outside-in*** |
|---|---|---|---|
| Resolução espacial | 0.5 - 1 grau | ~ 0.005 - 8 mm;<br>~0.025 - 0.1 grau | ~ 0.0015 - 0.2 %<br>do campo de visão |
| Precisão espacial | <= 5 graus | ~ 0.8 - 5 mm;<br>~ 0.1 to 3 graus | ~ 0.004 - 0.5 %<br>do campo de visão |
| Latência | ~ 1 ms | ~ 1 - 40 ms | ~ 1 ms |
| Precisão | De média a alta | Alta | Alta |
| Referência das coordenadas | Baseado nas articulações | Baseada no mundo | Baseada no mundo |
| Espaço de trabalho | Ilimitado | 1 – 2 m de raio | 1 – 4 m de raio |

Fonte: MULDER, 1994

Os sistemas do tipo *inside-in* apresentam uma pequena desvantagem na precisão dos movimentos capturados em relação aos sistemas *inside-out* e *ouside-in*. Todavia, o espaço de trabalho pode ser ilimitado, já que os sensores e os transmissores estão no ator, ou seja, não dependem do ambiente. Com relação às latências, as dos três sistemas são parecidas, porém, nos sistemas *inside-in* a latência pode ser até quarenta vezes maior do que nos outros dois.

Nos quesitos resolução espacial e precisão espacial, a dificuldade na comparação dos três sistemas é devido à diferença entre as unidades de medida utilizadas. Nos sistemas *inside-in* e *inside-out* estas características são descritas em números absolutos e nos sistemas *outside-*

*in* elas são descritas em proporcionais ao campo de visão. Porém, é possível comparar os sistemas sistemas *inside-in* e *inside-out* e constatar que tanto a resolução espacial quanto a precisão espacial são menores nos sistemas *inside-out*.

### 2.2.2 Forma de aquisição dos dados

Os dados capturados podem ser obtidos de forma direta ou indireta, se diferenciando pela forma com que estes são processados.

Nos sistemas de aquisição direta, os dados são obtidos diretamente dos sensores, não havendo a necessidade de processamento posterior. Sistemas deste tipo limitam o movimento do ator devido aos cabos ligados às fontes emissoras, além de possuírem baixa taxa de amostragem. Os sistemas acústicos, mecânicos e magnéticos se incluem nesta categoria.

Os sistemas de aquisição indireta necessitam de processamento via *software* para obter as coordenadas 3D dos marcadores. Esse pós-processamento impede que estes sistemas sejam utilizados em aplicações que necessitem de interatividade. Estes sistemas, apesar de caros, permitem uma maior liberdade do ator e possuem uma alta taxa de amostragem. Os sistemas ópticos são classificados como sistema de aquisição indireta.

## 2.3 Comparação das técnicas e *software* e captura

Cada uma das tecnologias de CM apresentadas tem a sua área de aplicação devido às suas características. Welch e Foxlin (2002) afirmam que, para uma tecnologia de CM ser considerada perfeita (i.e., ser ótima a todos os domínios de aplicação) esta deveria ser ótima nas dez características apresentadas pelos autores. A Tabela 2 compara dois sistemas de CM desenvolvidos nas tecnologias mais difundidas – óptica e mecânica - de acordo com as características apresentadas pelos autores.

Tabela 2 – Comparação de dois sistemas de CM de acordo com Welch e Foxlin (2002)

| **Características do sistema ideal** | **Sistema ideal** | **Tecnologia - Fabricante** | |
|---|---|---|---|
| | | Mecânico – X-IST | Óptico - PhaseSpace |
| Pequeno | Cada sensor/transmissor do tamanho de um DIP de 8 pinos, ou até mesmo um transistor. | Não. Utiliza exoesqueleto em todo o corpo do ator | Sim |
| Autocontido | Sem nenhuma outra parte para ser montada no ambiente ou no ator | Não, pois além dos potenciômetros para capturar a posição das juntas, necessita de um exoesqueleto para o rastreamento. | Não. Necessita de câmeras de vídeo, sincronizadores de LED, entre outros equipamentos. |
| Completo | Permitir o rastreamento em todos os DOF (*degree of freedom* ou graus de liberdade), tanto para orientação quanto para rotação | Não. Permite uma rotação de 340 graus por eixo | Sim |
| Preciso | Maior que 1mm para posição e 0.1 grau para orientação | Informação não disponível | Informação não disponível |
| Rápido | Taxa de captura de 1kHz e menos de 1ms de latência | Informação não disponível | Não. A taxa de captura é de 480hz |
| Não-oclusivo | Sim | Sim | Não |
| Robusto | Não ocorre degradação do desempenho devido a interferências | Sim | Sim |
| Tenaz | Não há distância máxima para o rastreamento | Não. O ator tem um diâmetro de 10 metros para se mover | Não. A distância máxima entre a câmera e o LED é de 8 metros |
| Sem fios | Sim | Sim (opcional) | Sim |
| Barato | Menos de US$1,00 por unidade | Entre US$ 21.717,90 e US$ 42.355,88 | A partir de US$ 49.000,00 |

Fonte: X-IST, 2006; PHASESPACE, 2006

Por meio da Tabela 2 é possível observar que os dois sistemas comparados têm no mínimo cinco características que não são compatíveis com o sistema ideal proposto por

Welch e Foxlin (2002). Portanto, caso houvesse a necessidade de escolha entre os dois sistemas comparados, deveria ser levado em conta a característica do movimento que seria capturado e o valor que poderia ser gasto.

Outra forma de se comparar os sistemas de CM é de acordo com as medidas descritivas propostas por Kalawsky (1993, apud SILVA, 2002). A Tabela 3 mostra a comparação de sistemas ópticos e mecânicos.

Tabela 3 – Comparação de sistemas ópticos e mecânicos de acordo com Kalawsky (1993)

| | **Sistemas Ópticos** | **Sistemas Mecânicos** |
|---|---|---|
| Resolução especial | ~0.0015 a 0.2% do campo de visão | ~ 0.5 – 1 grau |
| Precisão especial | ~0.004 a 0.5 % do campo de visão | <= 5 graus |
| Latência | ~ 1 ms | ~ 1 ms |
| Precisão | Alta (de ~ 0.0055 a 0.02 % do campo de visão) | Média a elevada |
| Taxa de amostragem | > 200 fps (*frames* por segundo) | > 120 fps |
| Referência das coordenadas | Baseado no mundo | Baseado nas articulações |
| Espaço de trabalho | Raio de 1 a 4m | Ilimitado |

Fonte: SILVA, 2002

As Tabelas 2 e 3 comparam características semelhantes. Entretanto, uma terceira forma de comparação unificando as duas tabelas poderia ser criada, já que algumas informações abordadas na Tabela 3 não são abordadas na Tabela 2, como a resolução espacial.

## 2.4 Representação do corpo humano

Para capturar corretamente os movimentos humanos é necessário haver uma representação da sua estrutura. Entretanto, não havia uma padronização desta representação, o que forçava os estúdios de animação a desenvolverem as suas próprias soluções. Para solucionar este problema, foi desenvolvido a H-Anim (H-ANIM, 2006), uma especificação internacional de representação abstrata 3D da figura humana, permitindo que dados obtidos de

sistemas de CM de uma determinada empresa sejam manipulados em *software* de animação desenvolvido por outra empresa. A descrição do humanóide feita pela H-Anim é em relação às articulações e às interfaces utilizadas para implementação, e não à forma do corpo. Este humanóide é descrito por objetos chamados de juntas, segmentos, locais e deslocadores.

As juntas são utilizadas para descrever as articulações do humanóide. Cada junta representa uma articulação. São organizadas de forma hierárquica, descrevendo o relacionamento entre as juntas pai e filho. A quantidade de juntas em um humanóide é variável e depende da necessidade do desenvolvedor. Esta quantidade é conhecida como *level of articulation* (LOA) ou nível de articulação. A única junta obrigatória em toda figura humanóide descrita de acordo com o H-Anim é a raiz da hierarquia, que na maioria dos casos é o quadril, denominado *HumanoidRoot*.

Os segmentos são as partes do corpo do humanóide – braço, perna, quadril, etc. - e são organizados na hierarquia do esqueleto como objetos filhos das juntas. Segmentos são unidos entre si por juntas.

A Figura 26 mostra a representação de uma figura humanóide simplificada com apenas quinze juntas – as esferas vermelhas – e 18 segmentos - as linhas pretas. Porém, a especificação H-Anim permite definir um nó para cada articulação do corpo humano. Essa liberdade de escolha no número de juntas possibilita que o esqueleto seja definido de acordo com a necessidade da animação.

Os locais são utilizados para três propósitos. O primeiro é definir um *end-effector*, que é uma folha da árvore hierárquica utilizada em sistemas de cinemática inversa (i.e., mãos e pés). O segundo é definir um ponto para anexar acessórios ao humanóide, como jóias e roupas, e o terceiro é definir um local para uma câmera, como os olhos do humanóide. São anexados na árvore de hierarquia do humanóide como filhos dos objetos segmentos.

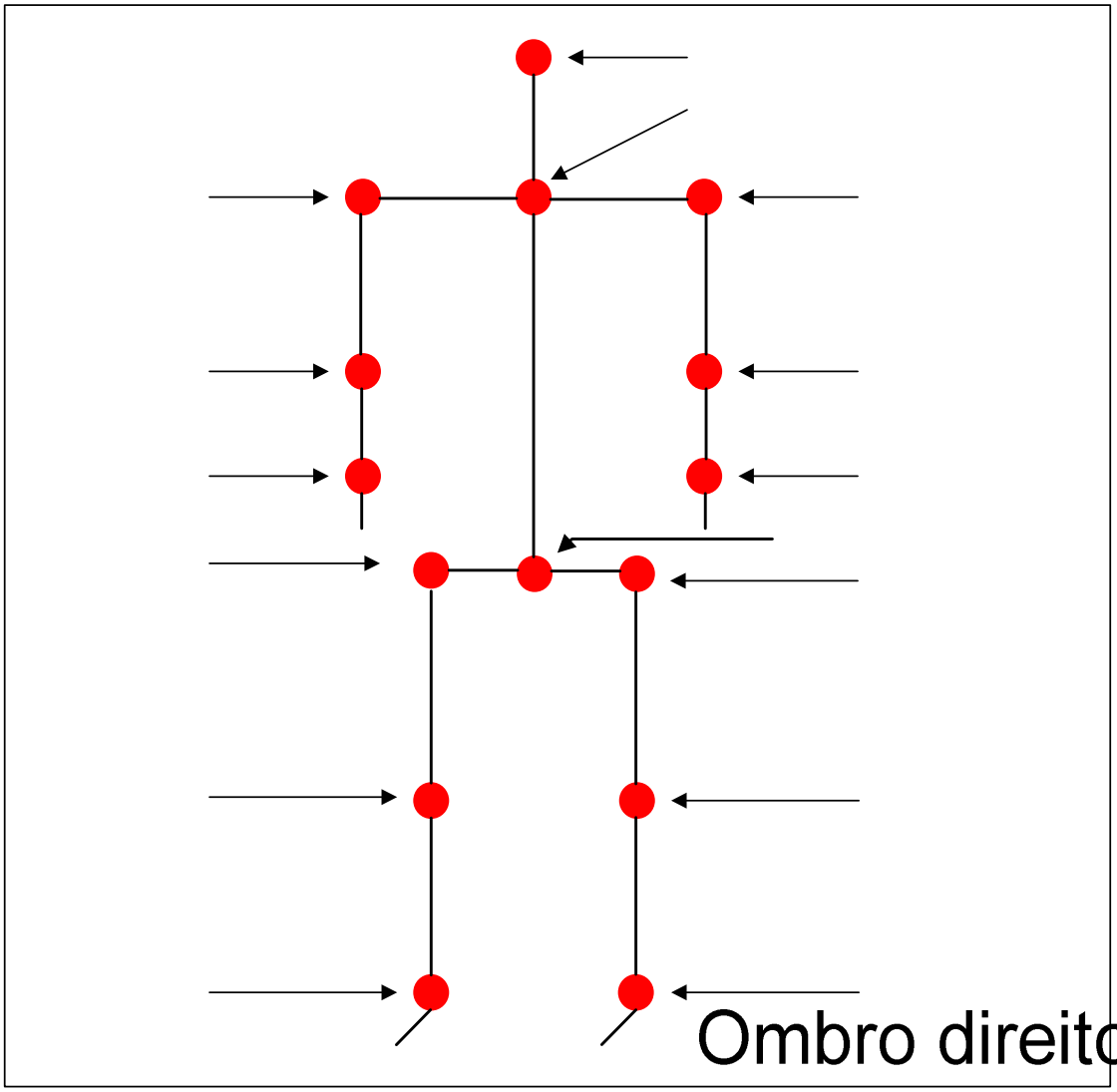


Figura 26 – Esqueleto humanóide com 15 juntas

Basicamente, os deslocadores são utilizados para guardar informações sobre os vértices responsáveis pelas formas do humanóide durante uma deformação.

O padrão H-Anim possibilita que a estrutura do corpo do ator e seus movimentos sejam exportados para a linguagem *Virtual Reality Markup Language* (VRML) e para o padrão X3D, permitindo que a estrutura e os movimentos capturados sejam carregados em programas de animação disponíveis.

## 2.5 Formatos de arquivo

Conforme mencionado na subseção anterior, ainda não há um padrão de arquivos pra CM, entretanto, muitos formatos utilizam o Código Padrão Americano para Troca de

Informações, ou *American Standard Code for Information Interchange* (ASCII), a fim de facilitar a leitura e o entendimento do mesmo pelos seres humanos (MEREDITH e MADDOCK, 2001).

Nas próximas seções serão descritos três tipos de arquivos para CM*: BioVision Animation* (BVA), *BioVision Hierarchy* (BVH) e *Acclaim Skeleton File/Acclaim Motion Capture* (ASF/AMC). Também será descrita a terminologia aplicada na descrição do assunto.

### 2.5.1 Terminologia

Antes de descrever os arquivos utilizados na CM, algumas palavras-chave devem ser definidas:

- **Esqueleto:** a estrutura completa do humanóide representado pelo arquivo.
- **Osso:** também chamado de segmento, é a entidade básica para a representação do esqueleto. São os menores segmentos do esqueleto, sujeitos à translações e rotações. O esqueleto é composto por vários ossos e, geralmente, são dispostos de forma hierárquica.
- **Canais:** são os atributos que descrevem a posição, orientação e escala de um osso. A animação ocorre com a variação desses valores.
- ***Frame*:** toda animação é composta de um determinado número de *frames* – ou quadros – e para cada *frame* os dados dos canais são definidos. Quando estes *frames* são exibidos seqüencialmente, é obtida a sensação de animação.

### 2.5.2 BVA

O formato BVA desenvolvido pela BioVision pode ser considerado o mais fácil de se implementar e é suportado pela maioria dos *softwares* de animação (LANDER, 1998). Por não ser disposto em hierarquia, cada segmento deve ser definido na posição desejada em determinado *frame* sem herdar a posição do segmento precedente. Isto causa redundância no

arquivo, pois algumas informações de posição e orientação acabam sendo definidas sem necessidade.

A Figura 27 mostra um fragmento de um arquivo BVA (UWa, 2007). Na primeira linha do arquivo encontra-se a palavra-chave `Segment:` seguida pelo segmento que será definido – no caso `Hips`, ou quadril. A segunda linha é composta pela palavra-chave `Frames:` seguida pelo número 2, que é o número total de *frames* do movimento do segmento. Na terceira linha há a palavra-chave `Frame Time:`, que define o tempo em segundos que cada *frame* deve ser exibido. Para se obter uma animação com 30 fps, deve-se definir esta propriedade com o valor 0.03333.

```
01 Segment:   Hips
02 Frames:    2
03 Frame Time: 0.033333
04 XTRAN       YTRAN   ZTRAN   XROT    YROT    ZROT    XSCALE  YSCALE  ZSCALE
05 INCHES      INCHES  INCHES  DEGREES DEGREES DEGREES INCHES  INCHES  INCHES
06  8.03        35.01   88.36   14.78  -164.35 -3.41    5.21    5.21    5.21
07  7.81        35.10   86.47   12.94  -166.97 -3.78    5.21    5.21    5.21
08 Segment:   Chest
09 Frames:    2
10 Frame Time: 0.033333
11 XTRAN       YTRAN   ZTRAN   XROT    YROT    ZROT    XSCALE  YSCALE  ZSCALE
12 INCHES      INCHES  INCHES  DEGREES DEGREES DEGREES INCHES  INCHES  INCHES
13  8.33        40.04   89.69  -27.24   175.94 -2.88    18.65   18.65   18.65
14  8.15        40.16   87.63  -31.12   175.58 -4.08    18.65   18.65   18.65
... (para todos os segmentos)
```

Figura 27 – Fragmento de um arquivo BVA

A quarta linha contém os canais para cada eixo na ordem `X`, `Y` e `Z`, que serão utilizados na animação. Para a translação, utiliza-se as palavras-chave `XTRANS`, `YTRANS` e `ZTRANS`. Para a rotação, `XROT`, `YROT` e `ZROR`. E para a escala, `XSCALE`, `YSCALE` e `ZSCALE`. Na quinta linha há a definição da unidade de cada canal.

As linhas seguintes são os dados do movimento, sendo que cada linha define o valor de um canal em um determinado *frame*.

Para os outros segmentos do esqueleto deve-se repetir estes passos. Nessa etapa também há a redundância, já que é necessário especificar novamente a quantidade de *frames* e a duração dos mesmos.

### 2.5.3 BVH

O formato BVH, também desenvolvido pela BioVision, foi criado para substituir o formato BVA. A sua maior vantagem sobre este é a definição do esqueleto de forma hierárquica, ou seja, o movimento de um segmento filho é diretamente dependente do movimento do osso pai (LANDER, 1998).

Um arquivo BVH é dividido em duas partes: a primeira define o esqueleto, sua hierarquia e posição inicial, e a segunda contém os dados do movimento. A Figura 28 é um fragmento de um arquivo BVH (UWb, 2007).

```
01 HIERARCHY
02 ROOT Hips
03 {
04      OFFSET  0.00    0.00    0.00
05      CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
06      JOINT Chest
07      {
08              OFFSET   0.00    5.21    0.00
09              CHANNELS 3 Zrotation Xrotation Yrotation
10              JOINT Neck
11              {
12                      OFFSET   0.00    18.65   0.00
13                      CHANNELS 3 Zrotation Xrotation Yrotation
14                      JOINT Head
15                      {
16                              OFFSET   0.00    5.45    0.00
17                              CHANNELS 3 Zrotation Xrotation Yrotation
18                              End Site
19                              {
20                                      OFFSET   0.00    3.87    0.00
21                              }
22                      }
23              }
24...
25 }
26 MOTION
27 Frames:    2
28 Frame Time: 0.033333
29 8.03  35.01   88.36  -3.41    14.78  -164.35  13.09   40.30  -24.60  ...
30 7.81  35.10   86.47  -3.78    12.94  -166.97  12.64   42.57  -22.34  ...
```

Figura 28 – Fragmento de um arquivo BVH

A primeira palavra do arquivo é a palavra-chave `HIERARCHY`, indicando o início da hierarquia do esqueleto. Na segunda linha encontra-se a palavra-chave `ROOT` seguida pelo nome do segmento raiz – no caso `Hips`. Após o "{" na terceira linha, encontra-se na quarta linha a palavra-chave `OFFSET` seguida dos valores do deslocamento dos eixos X, Y e Z em

relação ao segmento pai – no caso do segmento `ROOT` este valor é zero. A quinta linha se inicia com a palavra-chave `CHANNELS` seguida pela quantidade de canais e pela lista dos tipos de canal. A linha seguinte inicia-se com a palavra-chave `JOINT` seguida do nome do segmento. Este segmento filho é definido da mesma maneira que se definiu o segmento pai. Porém, ao invés da palavra-chave `JOINT`, pode-se encontrar a palavra-chave `End Site`, que define um segmento do tipo *end-effector*, ou seja, não tem filho. Neste caso, a linha seguinte após o "`{`" deve conter a palavra-chave `OFFSET` e os valores que definem a orientação e o tamanho deste segmento.

Após definir a hierarquia, define-se o movimento do humanóide, iniciando com a palavra-chave `MOTION`. A próxima linha contém a palavra-chave `Frames:` seguida do número de *frames* que a animação contém. A linha seguinte inicia-se com a palavra-chave `Frame Time:` que, assim como no formato BVA, define tempo em segundos que cada *frame* deve ser exibido. A próxima linha contém os dados dos movimentos realizados no primeiro *frame*, escritos na ordem em que os canais são definidos. Por exemplo, os seis primeiros números se referem aos canais do segmento `Hips`. Os próximos três números, aos canais do segmento `Chest`, e assim por diante. As linhas seguintes contêm os dados dos respectivos *frames*.

Apesar de a hierarquia dificultar a renderização, ela traz algumas vantagens ao usuário. Entre elas, é possível combinar dois movimentos distintos sem correr o risco de separar os segmentos durante o movimento (LANDER, 1998).

Como se pode notar, o canal referente à escala existente no BVA foi eliminado do BVH. Outra diferença em relação àquele formato é a possibilidade de variar a ordem dos canais, permitindo a definição da ordem da execução dos movimentos. Além disso, este formato permite a representação dos movimentos de vários esqueletos no mesmo arquivo.

### 2.5.4 ASF/AMC

Este formato é composto por dois arquivos: o ASF (Figura 29), que descreve o esqueleto e a sua hierarquia e o AMC (Figura 30), que contém a descrição do movimento. A separação desses dois arquivos permite que se tenha um arquivo ASF definindo o esqueleto e vários AMC para o movimento.

```
01 :version 1.10
02 :name BioSkeleton
03 :units
04    mass 1.0
05    length 1.0
06    angle deg
07 :documentation
08    Example of an Acclaim skeleton
09    To be used with "Walk.amc"
10 :root
11    axis XYZ
12    order TX TY TZ RZ RY RX
13    position 0.0 0.0 0.0
14    orientation 0.0 0.0 0.0
15 :bonedata
16 begin
17    id 1
18    name hips
19    direction 0.000000 1.000000 0.000000
20    length 0.000000
21    axis 0.00000 0.00000 0.00000 XYZ
22    dof rx ry rz
23    limits (-180.0 180.0)
24       (-180.0 180.0)
25       (-180.0 180.0)
26 end
...
27:hierarchy
28 begin
29    root hips
30    hips hips1 hips2 hips3
...
31 end
```

Figura 29 – Fragmento de um arquivo ASF

```
01 :FULLY-SPECIFIED
02 :DEGREES
03 1
04 root -1.244205 36.710186 -1.148101 0.958161 4.190043 -18.282991
05 hips 0.000000 0.000000 0.000000
06 chest 15.511776 -2.804996 -0.725314
07 neck 48.559605 0.000000 0.014236
...
08 2
09 root -0.227361 37.620358 1.672587 0.204373 -4.264866 -12.155879
10 hips 0.000000 0.000000 0.000000
11 chest 14.747641 2.858763 -1.345236
12 neck 44.651531 0.000000 -0.099206
...
```

Figura 30 – Fragmento de um arquivo AMC

O arquivo ASF é dividido em várias seções, e cada uma começa com uma palavra-chave iniciada por “:” e termina quando o início de outra seção é encontrado. As seções contidas no arquivo são:

- **`:version:`** a versão da definição do esqueleto
- **`:name:`** o nome do esqueleto
- **`:units:`** as unidades utilizadas por vários tipos de dados, como massa, tamanho e ângulo
- **`:documentation:`** permite armazenar informações sobre o projeto
- **`:root:`** permite definir características do segmento raiz, como a ordem de rotação - com a palavra-chave `axis` –, quais canais serão aplicados à raiz e em qual ordem eles deverão aparecer no arquivo AMC – com a palavra-chave `order` – e a posição e orientação iniciais do segmento raiz – com as palavras-chave `position` e `orientation`
- **`:bonedata:`** nesta seção ocorre a definição dos outros segmentos do esqueleto, divididos em blocos iniciados com a palavra-chave `begin` e terminados com `end`. Dentro desses blocos são informados o número de identificação do segmento pela palavra-chave `id` seguida do numero de identificação, o nome por `name` seguida do nome, a direção na qual o segmento deve ser desenhado por `direction` seguida das coordenadas, o tamanho do segmento por `length` seguida do tamanho, o eixo de

rotação do segmento por `axis` seguida das coordenadas e da ordem dos eixos, os canais de movimento na ordem em que eles aparecem no arquivo AMC por `dof` seguida dos canais, e os limites de cada canal especificado em `dof` por `limits` seguida dos ângulos. Para todos os segmentos deve haver um bloco `begin` – `end` na seção `:bonedata`

- **`:hierarchy:`** é definida a hierarquia do esqueleto em um bloco `begin` – `end`, e em cada linha deve conter o nome do segmento pai seguido por seus filhos

No arquivo AMC cada *frame* de animação começa com uma linha declarando o número do *frame*. As linhas seguintes contêm o nome dos segmentos, seguidos pelos canais definidos na seção *`dof`* de cada segmento no arquivo ASF. Cada linha deve conter um segmento.

## 2.6 Comparação dos formatos de arquivo

Para a escolha do formato de arquivo a ser utilizado para armazenar os movimentos realizados, devem ser levados em conta os pontos fortes e fracos de cada formato. O formato BVA deve ser a última escolha, já que, apesar de simples, não há uma hierarquia no esqueleto, aumentando a redundância no arquivo.

O formato BVH, como mencionado anteriormente foi criado para substituir o BVA, provendo uma estrutura hierárquica para o esqueleto, permitindo a recursividade na realização dos movimentos. Entretanto, este formato ainda apresenta alguns pontos fracos, como a falta de informações explícitas sobre como desenhar os segmentos, incluindo a sua orientação, não há unidades de calibração, como a escala da medida do deslocamento das juntas, e detalhes sobre o ambiente, como a orientação.

O ASF/AMC, apesar de ser mais complicado que os dois formatos de arquivos anteriores, é o mais compreensível para o ser humano. Além de oferecer a estrutura hierárquica, este formato permite informar dados sobre o ambiente, como as unidades de

massa, tamanho e ângulo para as rotações. Além disso, permite informar as limitações para a rotação de cada junta, diminuindo a ocorrência de erros na captura.

O formato BVH é mais versátil do que os outros dois formatos, pois permite especificar mais do que um esqueleto por arquivo.

Os formato de arquivo BVH e ASF/AMC podem ser importados para *softwares* como o Maya (AUTODESK, 2006) por meio de ferramentas como *Life Forms MoCap Conversion* (CREDO, 2006) e o PolyTrans (OKINO, 2006), que também está disponível para o 3Ds Max (AUTODESK, 2006).

## 2.7 Considerações finais

Conforme visto neste capítulo, sistemas de CM transformam os movimentos realizados por um ator em dados computacionais. Existem vários tipos de sistemas de CM e várias formas de classificá-los: de acordo com a tecnologia aplicada, pelo posicionamento dos sensores e marcadores, entre outras.

Na classificação por tecnologia, os quatro tipos mais conhecidos são: ópticos, mecânicos, magnéticos e acústicos. Outra forma de classificar é de acordo com a posição dos sensores e transmissores:

- **Ópticos:** Sistemas ópticos de CM, no qual os emissores de luz/marcadores são fixados no corpo do ator e câmeras são posicionadas para capturar os movimentos, são classificados como *outside-in*
- **Mecânicos:** Nos sistemas mecânicos, o ator utiliza um exoesqueleto em que potenciômetros localizados próximo às juntas capturam as rotações realizadas. Estes sistemas são classificados como *inside-in*

- **Magnéticos:** Nos sistemas magnéticos, receptores são fixados nas articulações do ator e transmissores de pulsos eletromagnéticos são posicionados no espaço a ser rastreado. São classificados como *inside-out*
- **Acústicos:** Nos sistemas de CM acústicos, emissores de sinais sonoros são fixados nas articulações do ator e três receptores são posicionados no espaço em que o ator irá se movimentar. Assim como os sistemas magnéticos de CM, são classificados como *inside-in*

Os formatos de arquivo abordados foram BVA, BVH e ASF/AMC. A principal diferença entre os dois últimos e o primeiro é que estes possuem uma estrutura hierárquica, e uma das diferenças entre o formato BVH e ASF/AMC é que este permite informar as limitações de cada articulação, reduzindo erros de captura.

As diversas tecnologias e formatos de arquivo existentes permitem que o usuário desenvolva um sistema de CM de acordo com a sua necessidade, levando em consideração os pontos positivos e negativos de cada tipo.

# CAPÍTULO 3 - SISTEMAS ÓPTICOS DE RASTREAMENTO

Conforme já especificado neste trabalho, o rastreamento de movimentos pode ser aplicado em várias áreas, desde o rastreamento da trajetória de objetos até o rastreamento dos movimentos do corpo humano. Neste Capítulo serão descritos alguns sistemas de rastreamento encontrados na literatura.

## 3.1 *MagicMouse*

O projeto *MagicMouse* (WOODS et al., 2003) descreve um dispositivo de interação não-convencional desenvolvido com a biblioteca ARToolkit que permite ao usuário uma interação mais intuitiva tanto em ambientes bidimensionais (2D) como em ambientes 3D, simplesmente movendo e rotacionando o pulso.

Para que os movimentos realizados sejam entendidos pelo sistema como dados de entrada, o usuário deve vestir uma luva com um marcador do ARToolkit fixado na parte frontal, conforme a Figura 31. Os movimentos realizados nos eixos X, Y e Z permitem a captura dos dados de entrada em seis graus de liberdade.

Figura 31 – Luva com marcador fixado utilizada pelo MagicMouse

A maior vantagem do *MagicMouse* em relação aos outros dispositivos de interação em ambientes 3D é o baixo custo – o preço de uma simples câmera USB. Além disso, muitos

dispositivos disponíveis no mercado não permitem uma interação intuitiva com o ambiente 3D ou só operam com três graus de liberdade, tornando necessária a utilização de botões extras.

Entretanto, o *MagicMouse* também apresenta desvantagens, algumas intrínsecas da biblioteca ARToolkit como a oclusão, e questões relacionadas à iluminação da área de trabalho. Outra desvantagem é a fadiga que o dispositivo pode causar ao usuário após um longo tempo de utilização.

## 3.2 Rastreamento ótico baseado em marcadores passivos utilizando uma câmera

Devido ao alto custo dos rastreadores de movimentos disponíveis no mercado, Lourenço (2004) desenvolveu um sistema de rastreamento que utiliza a biblioteca ARToolkit como ferramenta para capturar os movimentos corporais de um ator. Além de ser gratuita para fins não comerciais, a biblioteca ARToolkit permite “utilizar objetos virtuais para facilitar o acompanhamento do processo de detecção dos marcadores acoplados ao corpo do usuário”. Os marcadores, impressos em papéis e aplicados sobre cartões resistentes para evitar deformações conforme os movimentos são realizados, e presos ao corpo do ator por meio de fitas feitas de velcro, servem como pontos de referência para a aquisição das coordenadas das articulações do ator. O único problema encontrado nessa configuração foi em relação à cabeça, pois devido ao seu formato irregular, a fita se deslocava conforme o movimento.

Outra vantagem da utilização do ARToolkit para o rastreamento dos movimentos corporais é que, ao contrário dos outros sistemas ópticos de captura de movimento, não há necessidade de rastreamento do marcador *frame*-a-*frame* para saber a qual junta ele pertence, já que cada marcador tem o seu padrão que pode ser atribuído a uma determinada junta. Estes padrões foram escolhidos por meio de testes realizados para verificar o desempenho do rastreamento.

O sistema é dividido em três módulos: Módulo de Captura e Detecção, Módulo de Exibição da Captura em Tempo Real e Módulo de Edição e Visualização dos Dados Capturados. A Figura 32 mostra a estrutura detalhada do sistema.

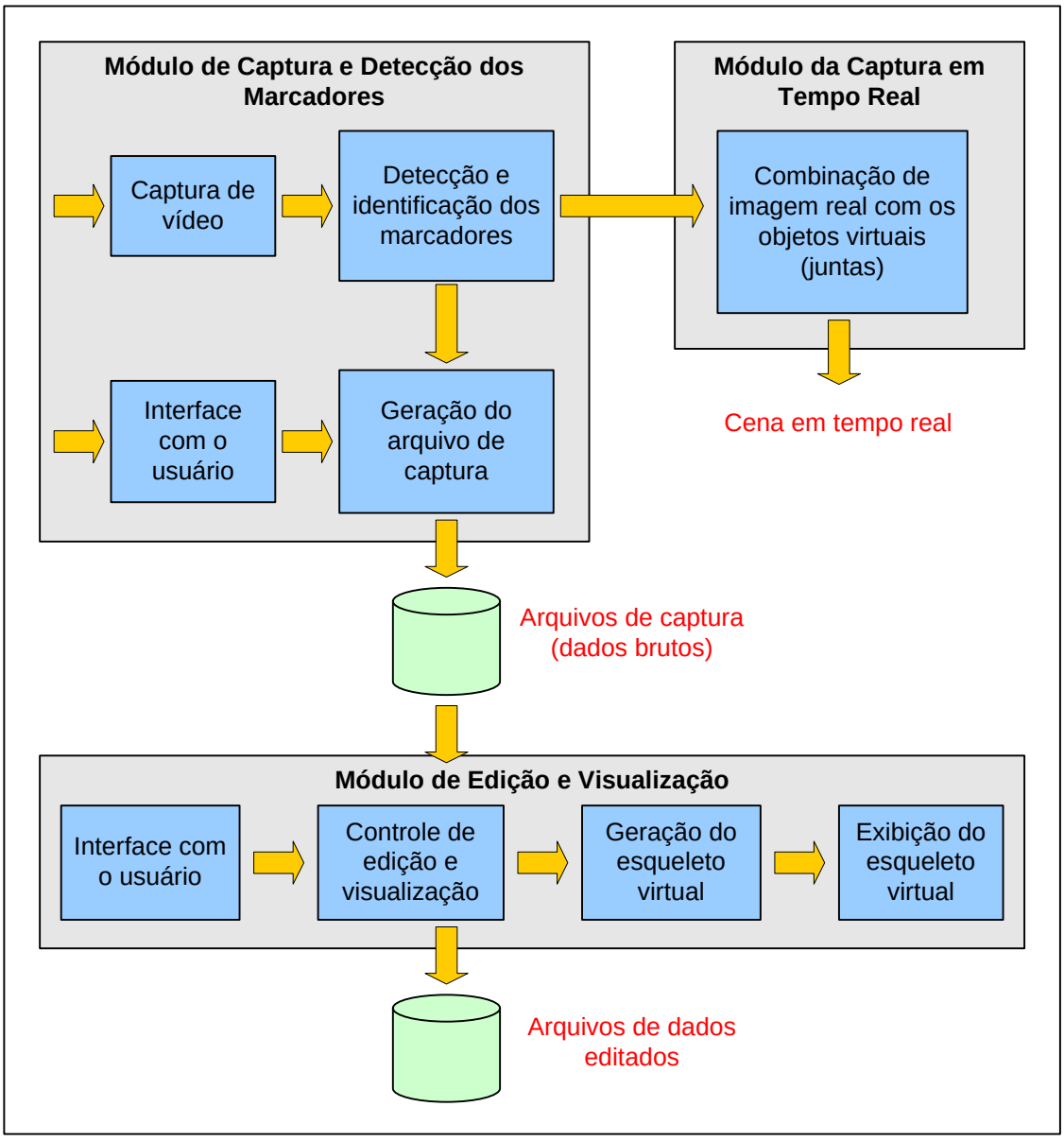


Figura 32 – Estrutura dos módulos do sistema de rastreamento de Lourenço (2004)

O Módulo de Captura e Detecção tem como objetivo detectar e rastrear os marcadores presentes nos quadros capturados pela câmera de vídeo e armazenar as suas coordenadas e orientações.

Com as informações obtidas no módulo anterior, o Módulo de Exibição da Captura em Tempo Real exibe as imagens capturadas pela câmera, acrescidas de esferas representando

as juntas do ator. O objetivo deste módulo é fornecer uma resposta ao rastreamento em tempo de execução, permitindo ao usuário avaliar quais marcadores estão sendo rastreados, e caso o resultado não seja satisfatório, é possível alterar o valor do limiar de binarização.

O Módulo de Edição e Visualização dos Dados Capturados realiza o pós-processamento, visando reconstruir o movimento capturado no primeiro módulo e armazenado no segundo. O usuário pode escolher quais juntas ele deseja visualizar, além de controlar a animação, pausando-a em determinado *frame* ou visualizá-la passo-a-passo.

Para este tipo de aplicação, o ARToolkit se mostrou satisfatório em relação à confiabilidade e precisão. Entretanto, conforme se aumentou a distância entre a câmera e os marcadores, as variações de posicionamento se tornaram mais constantes. O sistema também apresentou problemas em relação à taxa de captura conforme se aumentava o número de marcadores rastreados. Porém, a utilização de somente uma câmera para a captura das imagens do mundo real foi o grande problema do sistema, o que aumentou a ocorrência de oclusão devido à pequena área coberta pela câmera. O autor destaca a implementação do um sistema de rastreamento similar, porém, utilizando múltiplas câmeras de vídeo como um dos trabalhos futuros, além da utilização de marcadores diferentes para o rastreamento da mesma junta. As duas medidas serviriam para diminuir os problemas de oclusão.

## 3.3 Captura de movimentos baseado em marcadores passivos e múltiplas câmeras

Desenvolvido por Dias (2005), este suporte é composto por um conjunto de quatro módulos de serviços “o qual implementa a captura e sincronização de dados de posicionamento associados a marcadores passivos, utilizando diversas câmeras”. Estes módulos são:

- **Módulo de Captura:** recebe as informações geradas pelo ARToolkit (matriz de transformação) a partir das imagens capturadas e adiciona algumas informações de controle, como o número do frame
- **Módulo de Conversão:** organiza e verifica os tipos das informações de rastreamento enviadas pelo Módulo de Captura, adiciona mais informações de controle e converte estes dados em uma *string* de posicionamento. Além disso, este módulo faz a conversão das *strings* recebidas via rede para matriz de posicionamento e informações de controle
- **Módulo de Comunicação:** recebe e organiza a *string* do Módulo de Conversão no formato utilizado para comunicação e realiza a troca de informações via *socket* por meio do protocolo *User Datagram Protocol* (UDP)
- **Módulo de Matrizes:** recebe as informações dos marcadores capturados pela câmera secundária e converte a sua coordenada para a coordenada da câmera principal. Esta conversão só ocorre quando a câmera principal está sob oclusão.

A estrutura em módulos permite que o sistema desenvolvido possa ser implementado para várias câmeras. Esta configuração também permite que o trabalho desenvolvido seja facilmente utilizado por outras aplicações.

O sistema de captura é formado por no mínimo três computadores, também chamados de *Hosts*: um *Host* de Visualização e dois ou mais *Hosts* de Câmera. Cada *Host* de Câmera, composto pelos Módulos de Captura, Conversão e Comunicação, é responsável pela captura dos dados dos movimentos por meio de uma câmera digital e pelo envio destes dados pela rede para o *Host* de Visualização. O *Host* de Visualização recebe as informações enviada pelos *Hosts* de Câmera e exibe os movimentos capturados. A Figura 33 descreve as atividades realizadas pelos *Hosts* de câmera, e a Figura 34 as atividades realizadas pelo *Host* de visualização.

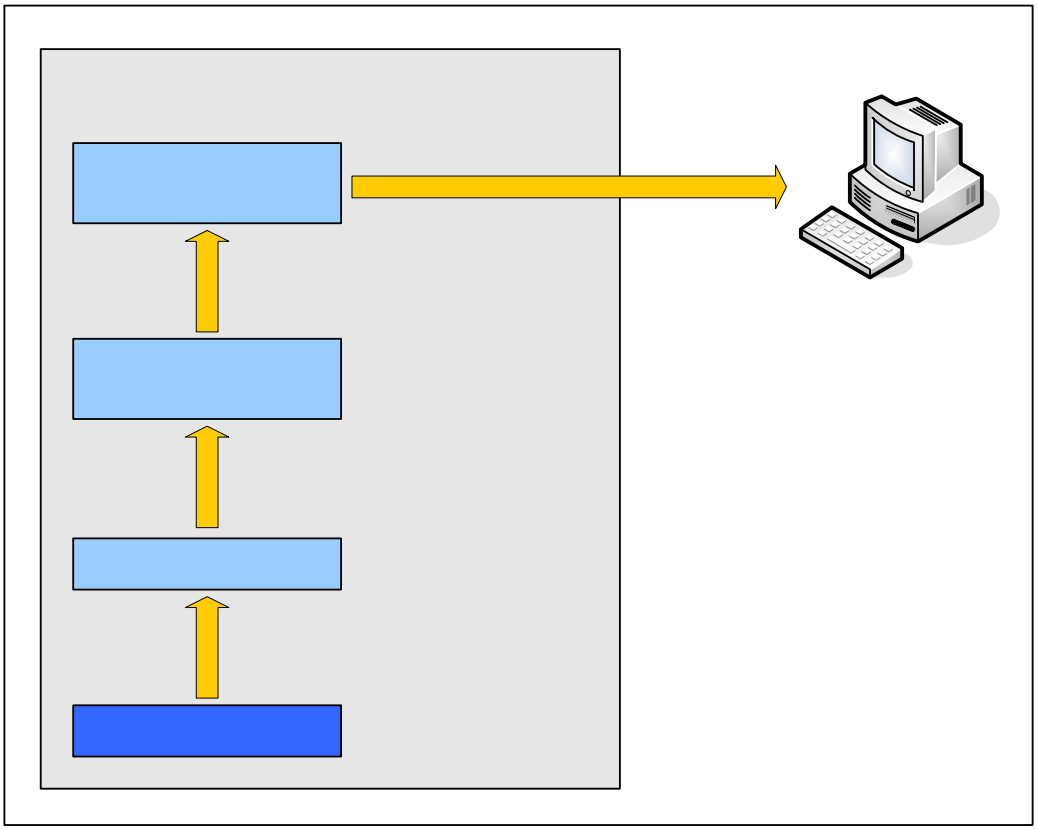

Figura 33 – Atividades realizadas pelos Hosts de câmera

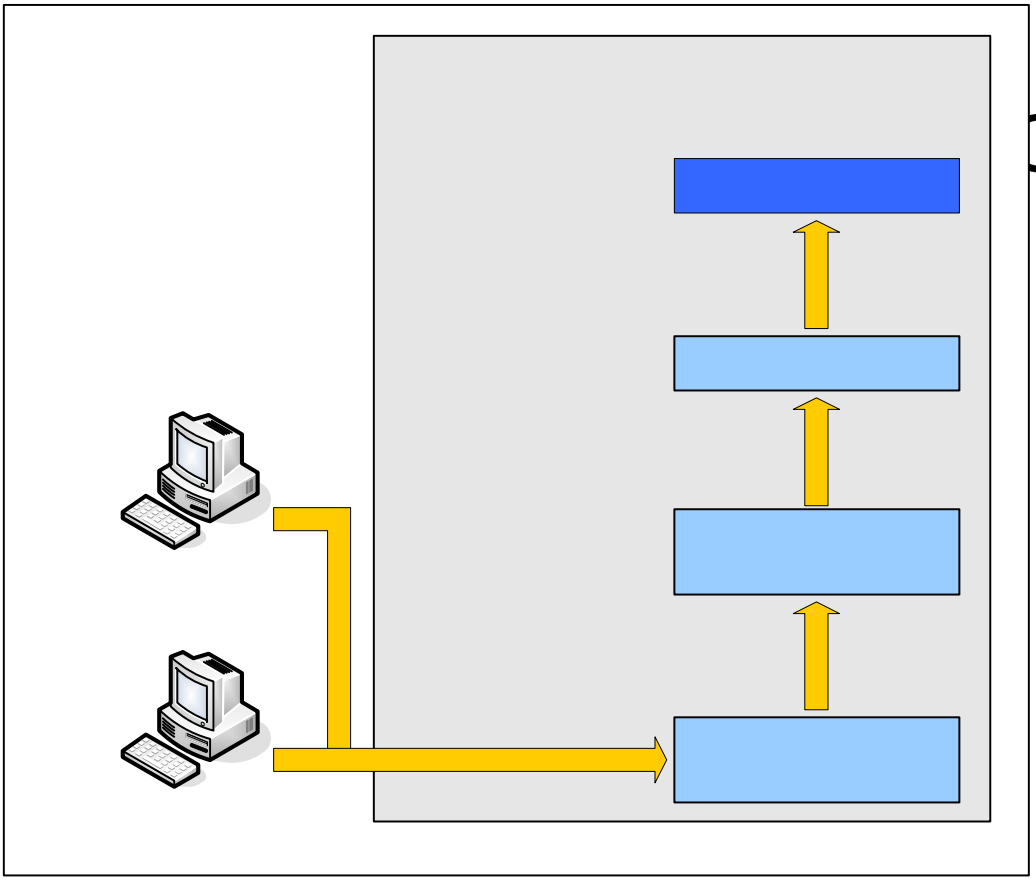


Figura 34 – Atividades realizadas pelo Host de visualização

Nos testes realizados por Dias (2005) foram utilizados marcadores do ARToolkit medindo 5 centímetros por 5 centímetros. Os testes também foram realizados com duas câmeras, mas como citado anteriormente, a estrutura modular permite que o sistema seja implementado para várias câmeras. Os resultados obtidos pelo autor foram satisfatórios, já que a variação entre os valores capturados pelas câmeras primária e secundária foi considerada pequena (aproximadamente 1% no raio e desvio padrão médio de 1mm).

## 3.4 Interfaces para utilização em sistemas de Realidade Aumentada

Devido à importância da IHC e visando implementar um suporte à interação em sistemas de RA, Lopes (2005) apresenta um conjunto de três módulos que provê funções para a criação desse tipo de interface utilizando a biblioteca ARToolkit. Esses módulos são:

- **Módulo de Ambiente:** com as funções oferecidas por este módulo, é possível carregar imagens nos formatos *Bit Map* (BMP) e *Joint Photographic Experts Group* (JPEG) como imagens de *backgroud* ou texturas nos objetos 3D. Também é possível controlar a iluminação do objeto virtual visando oferecer aspectos mais reais ao AV, aplicar transparência para evitar que um objeto obstrua a visibilidade a outro objeto, adição de sons no formato de arquivo *Waveform Audio Format* (WAV) e vídeo no formato *Audio Video Interleave* (AVI)
- **Módulo de Formas Geométricas:** este módulo oferece funções para a criação de formas geométricas 2D e 3D com características úteis para aplicações de RA. Além disso, permite a adição de textos, objetos 3D desenvolvidos em ambientes gráficos profissionais e a visualização de objetos em *wireframe*, ou seja, somente as linhas de estrutura do objeto sem textura ou superfície
- **Módulo de Interação:** disponibiliza funções matemáticas para interação com o AV. Estas funções são a função de detecção de colisão entre duas esferas (a forma mais

simples e que exige menor esforço computacional de detecção de colisão), a função que calcula distância entre dois pontos e a função que calcula a taxa de fps

Utilizando os módulos descritos anteriormente, o autor desenvolveu quatro protótipos para a interação em RA. O primeiro protótipo desenvolvido é uma Interface de Controle 2D. Esta interface permite que, por meio de controles em uma interface 2D comum, o usuário possa alterar as características de um objeto virtual. Essas características são: forma (é possível escolher entre em cubo, uma chaleira, umas esfera e um cone), cor, escala, posição e orientação.

O segundo protótipo apresentado foi o *Augmented Reality Mouse* (ARMouse), que "permite a alteração em tempo real do objeto virtual a partir de um *mouse* composto de marcadores" (LOPES, 2005). O ARMouse, definido como um dispositivo de interação não-convencional, é composto por sete marcadores dispostos em um placa, conforme a Figura 35. De acordo com o autor, "três marcadores centrais são responsáveis pelo eixo X, Y e Z". Outros dois marcadores são responsáveis pela translação e outros dois pela rotação.

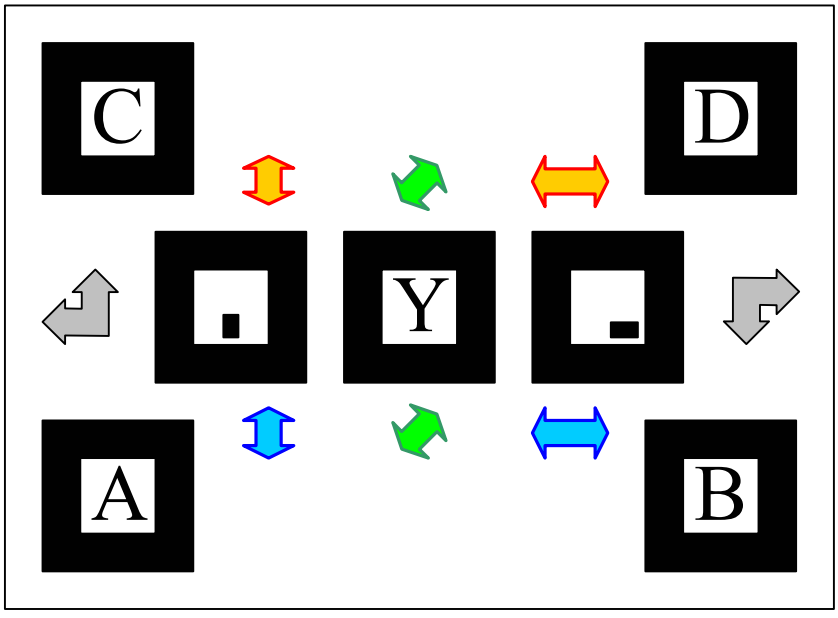


Figura 35 – Imagem da placa do ARMouse

A interação com o AV ocorre quando é realizada uma seqüência lógica de oclusões dos marcadores da placa. Por exemplo, se o usuário deseja deslocar um objeto virtual, deverá

ocultar um marcador específico de direcionamento e depois um marcador para qual eixo haverá esse deslocamento.

Outro protótipo apresentado por Lopes (2005) é a criação de teclados virtuais. Estes teclados virtuais são compostos por “vários marcadores espalhados no mesmo plano a uma distância pré-determinada entre eles”. A utilização de vários marcadores diminui a oclusão, pois caso um marcador não esteja visível, o programa irá procurar outro marcador para projetar o teclado virtual sobre o plano.

Para a interação com esses teclados virtuais, são utilizados marcadores fixados nos dedos do usuário ou um bastão com um marcador fixado na ponta. O reconhecimento da interação entre os dedos ou o bastão e o teclado é feita com a função de detecção de colisão e a função que calcula a distância entre dois pontos (que verifica se os dedos ou o bastão interagiram com o teclado), oferecidas pelo Módulo de Interação.

O último protótipo apresentado pelo autor são as Pinças Virtuais. Neste protótipo, “dois marcadores representam objetos de interação e um marcador representa um objeto fixo”. Sobre os marcadores foram colocadas esferas virtuais, e a colisão entre as esferas virtuais de interação (que podem estar fixadas nas pontas dos dedos do usuário) e a esfera virtual fixa faz com que esta acompanhe os movimentos daquelas, simulando uma interação realizada pelo usuário no mundo real.

## 3.5 *Finger tracking for interaction in Augmented Reality environments*

Um AV não deve somente apresentar os objetos virtuais da forma mais real possível, mas também oferecer uma forma de interação que seja rápida, precisa e natural. A manipulação direta dos dados (por exemplo, por meio de uma varinha eletrônica), diferentemente da manipulação indireta (realizada por *mouse*) permite a movimentação em seis graus de liberdade. Entretanto, as varinhas eletrônicas utilizam rastreadores magnéticos, que limitam o usuário devido aos fios, falta de precisão e susceptibilidade a interferências

eletromagnéticas. Devido a essas características dos rastreadores magnéticas, os rastreadores ópticos são mais atrativos para o rastreamento dos movimentos humanos.

Dorfmüller-Ulhaas e Schmalstieg (2001) descrevem um sistema para rastreamento óptico dos dedos das mãos para ser utilizado em ambientes 3D, tanto de RA quanto de RV. O sistema proposto pelos autores visa fornecer uma forma de interface “espontânea e intuitiva, através de gestos, e que seja simples, de baixo custo, rápida e não oclusiva”. Além disso, o sistema não necessita de um processo de inicialização para o rastreamento, além de não haver necessidade de adaptar o sistema para diferentes usuários.

O sistema proposto consiste em marcadores retroflexivos fixados em uma luva e iluminados por luzes infravermelhas. Um par de câmeras de vídeo equipadas com lentes infravermelhas é posicionado voltado para a área onde ocorrerá o movimento. As lentes infravermelhas irão filtrar os marcadores infravermelhos, fazendo com que estes fiquem mais visíveis para as câmeras.

A luva utilizada no sistema (Figura 36) é composta de um material flexível, possibilitando que se adapte a usuários com mãos de tamanhos diferentes. As esferas retroreflexivas são fixadas entre si por um arame a uma distância conhecida, fazendo com que as distâncias entre as esferas permaneçam as mesmas.

Figura 36 – Luva com esferas retroreflexivas

Após a captura de cada *frame* pelas duas câmeras, estes são processados e as posições 2D dos marcadores são encontradas. Os dados obtidos pelo processamento das imagens são relacionados entre si e a algoritmos de estimação de movimentos, e são determinadas as posições 3D dos marcadores, permitindo que o “dedo virtual” seja renderizado na tela.

A interação entre o usuário e o AV é feita flexionando o dedo para pegar o objeto virtual. Permanecendo com o dedo flexionado, o usuário pode transladar ou rotacionar o objeto de forma análoga ao mundo real. Quando todas as operações desejadas forem realizadas, o usuário pode esticar o dedo, soltando assim o objeto virtual.

## 3.6 Considerações finais

Neste Capítulo foram apresentados sistemas de rastreamento desenvolvidos com tecnologia óptica que utilizam câmeras de vídeo como sensor de captura. Quatro dos cinco sistemas apresentados utilizam a biblioteca ARToolkit, apresentada no Capítulo 1. Conforme apontado no subseção 1.6, a biblioteca ARToolkit possibilita o desenvolvimento de aplicativos em em diversas áreas.

Os trabalhos citados foram: o *MagicMouse*, proposto por Woods et al. (2003); um rastreador de movimentos corporais utilizando a biblioteca ARToolkit, desenvolvido por Lourenço (2004); módulos para auxiliar o desenvolvimento de aplicações com a biblioteca ARToolkit utilizando múltiplas câmeras, apresentado por Dias (2005); a implementação de três módulos para auxiliar o desenvolvimento de interfaces para RA, descrito por Lopes (2005), e um rastreador para os dedos das mãos, de Dorfmüller-Ulhaas e Schmalstieg (2001).

# CAPÍTULO 4 - IMPLEMENTAÇÃO

A Captura de Movimentos Humanos tem por objetivo reproduzir com a máxima precisão possível o movimento realizado por atores, permitindo maior grau de realismo para as animações tridimensionais.

Conforme salientado no Capítulo 2, existem várias tecnologias disponíveis para realizar a captura dos movimentos de um ator. Para escolher qual tecnologia de CM adotar, deve ser levado em consideração alguns aspectos, tais como o tipo de movimento que será realizado (rápido ou lento), o valor que poderá ser gasto no equipamento ou em um estúdio de animação, já que é necessário um pós-processamento dos dados a fim de obter um movimento mais próximo ao real, a parte do corpo que será capturada, o tempo que o ator deverá usar a roupa ou o exoesqueleto, visto que este pode causar fadigas no ator, entre outros.

Infelizmente ainda não há um sistema de CM perfeito que reúna todas as características apresentadas por Welch e Foxlin (2002), porém, o ARToolkit permite que se construa sistemas mais baratos, mais leves (por utilizar marcadores de papel) e com menor necessidade de pós-processamento devido à singularidade de cada marcador.

A aplicabilidade da biblioteca ARToolkit para sistemas de CM foi demonstrada em Lourenço (2004). No trabalho apresentado foi utilizada uma câmera de vídeo para registrar a posição dos marcadores presos no corpo de um ator, capturando o movimento deste. O autor, porém, propôs algumas melhorias como trabalhos futuros, dentre as quais melhorias propostas está a utilização de um esqueleto hierarquizado que facilite a representação dos dados capturados a fim de que possam ser armazenados em um formato padronizado.

Além desta melhoria, também é de grande importância que o sistema de captura possibilite a calibração entre as juntas do ator e as juntas de sua representação virtual, tornando o movimento capturado mais fiel ao realizado.

Dessa forma, é apresentado um sistema óptico de rastreamento de movimentos humanos utilizando a biblioteca ARToolkit que possibilita a calibração das juntas do ator e o armazenamento dos movimentos realizados em um formato padronizado e hierárquico.

## 4.1 Visão geral do sistema

Para a implementação desse sistema, foi necessário desenvolver seis bibliotecas, ou módulos, de apoio: **libArq**, **libMoCap**, **libBvh**, **libMcp**, **libMcpToBvh** e **libMat**. Estes módulos fornecem as funções e estruturas de dados utilizadas na captura e conversão de formatos. Os cinco principais módulos (libArq, libMoCap, libBvh, libMcp e libMcpToBvh) serão descritos detalhadamente no subitem 4.3

### 4.1.1 Captura de movimentos

O protótipo de Captura de Movimentos utiliza cinco Módulos de Suporte para realizar a calibração do esqueleto, captura do movimento, visualização em tempo real e a geração dos arquivos BVH e *Motion Capture File* (MCP), formato de arquivo que foi criado para esse trabalho e será explicado adiante no subitem 4.3.4.

A fase de calibração tem por objetivo manter as proporções entre as juntas representadas no programa e as juntas do ator, visando extrair as informações necessárias para que a captura dos movimentos ocorra de forma mais próxima à realidade. Esse trabalho é realizado pelo usuário por meio de elementos de GUI que permitem selecionar qual junta será calibrada e realizar a translação nos três eixos dos cubos que representam as juntas dos atores. A Figura 37 mostra a articulação do ombro direito antes e depois da calibração, em que é possível observar o deslocamento do cubo que representa a articulação. Dentre as informações retiradas desta fase, tem-se:

- a posição de cada marcador em relação à câmera;

- o deslocamento entre a posição dos marcadores e à articulação que cada marcador representa.

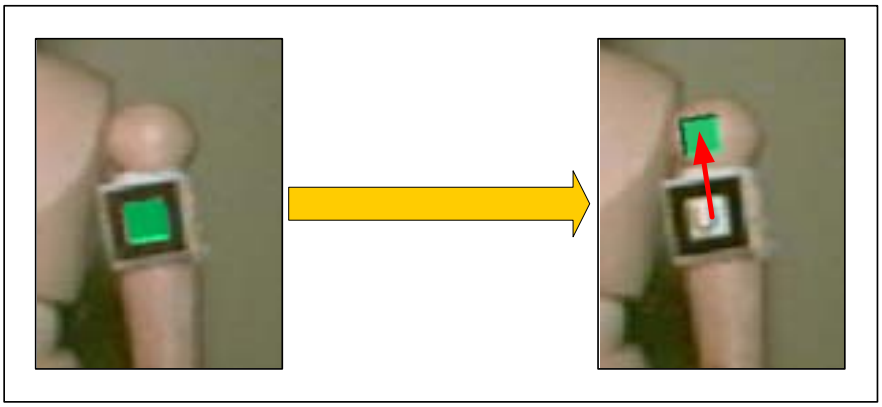

Figura 37 – Janela de calibração e captura

A Figura 38 mostra a janela do módulo de captura e calibração. O painel localizado no lado direito desta janela contém os componentes de GUI do GLUI necessários para a calibragem. Na parte superior deste painel há um painel contendo nove *radio buttons* que compõem o conjunto de articulações, onde é escolhida a junta que será calibrada (Figura 39). Abaixo desse painel, há dois controles de translação – um para os eixos x e y e outra para o z (Figura 40). Estes controles permitem movimentar o objeto virtual, inicialmente posicionado sobre o marcador, para a posição real da junta.

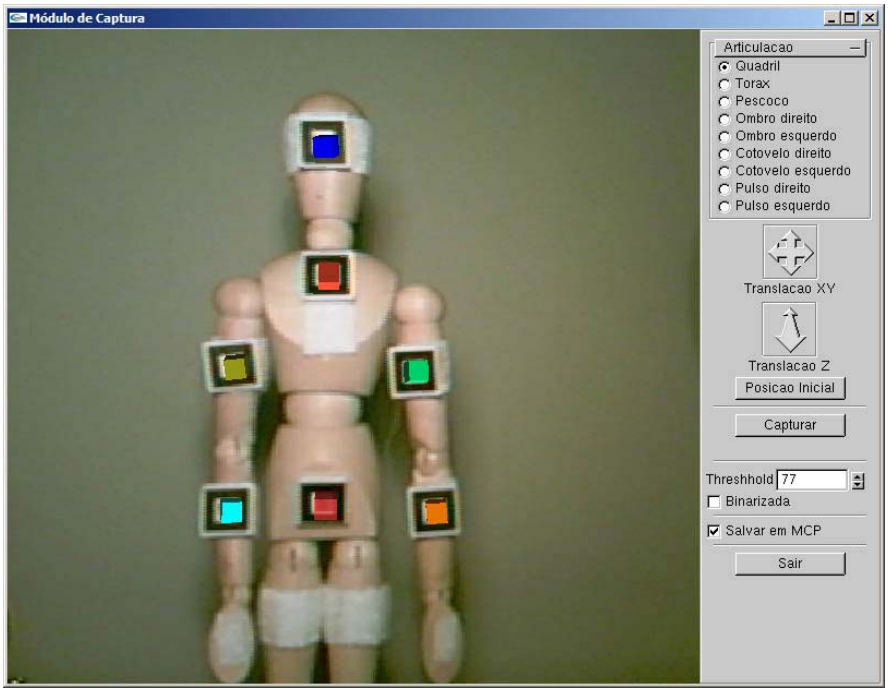


Figura 38 – Janela de calibração e captura

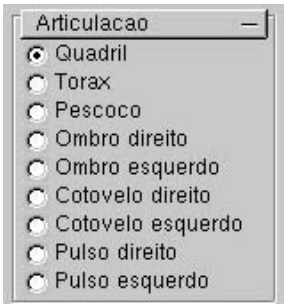


Figura 39 – Painel com *radio buttons* para a seleção da junta

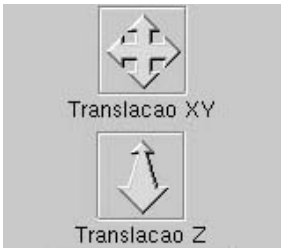


Figura 40 – Controles de translação

A interface também apresenta um *spinner* para alterar o valor do *threshold*, com o objetivo de diminuir os problemas de iluminação, e um *checkbox* para exibir a imagem capturada de forma binarizada para que o usuário possa visualizar o resultado provocado pelo valor escolhido para o *threshold* (Figura 41). Estas alterações podem ser feitas e visualizadas em tempo de execução do programa.

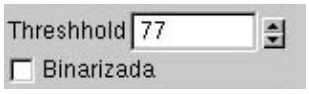


Figura 41 – *Spinner* e *checkbox* para controle da binarização

Os dados obtidos na calibragem são salvos em um arquivo temporário chamado de inicial.tmp. A estrutura de dados utilizada para isso está no Módulo de Suporte libMoCap.

Depois de realizada a calibração, é iniciada a captura do movimento e a geração de mais dois arquivos temporários: um para os dados de fps e número de *frames* e um para armazenar o movimento realizado.

A Figura 42 mostra o fluxograma simplificado da captura do movimento realizado pelo ator. Os processos de captura e calibração resultam em três arquivos binários temporários que contêm a posição inicial do esqueleto do ator, o movimento realizado durante a captura e

os dados sobre esse movimento, como taxa de fps, duração do movimento e número de *frames* capturados.

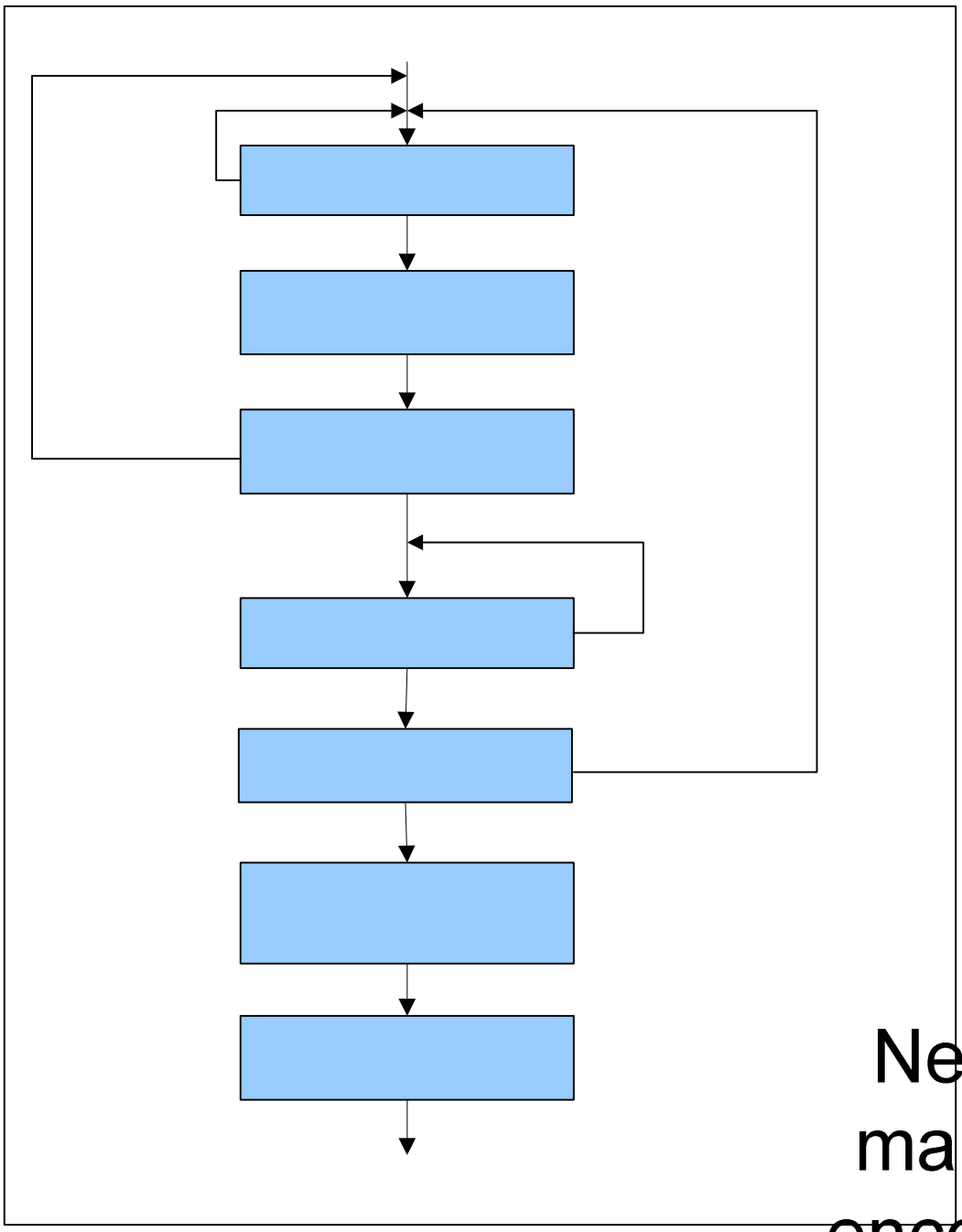

Figura 42 – Fluxograma simplificado da captura

Nenhum marcador encontrado

Os arquivos temporários obtidos no processo de captura são utilizados como entrada para gerar o arquivo BVH. A Figura 43 mostra o fluxograma simplificado da geração do arquivo BVH.

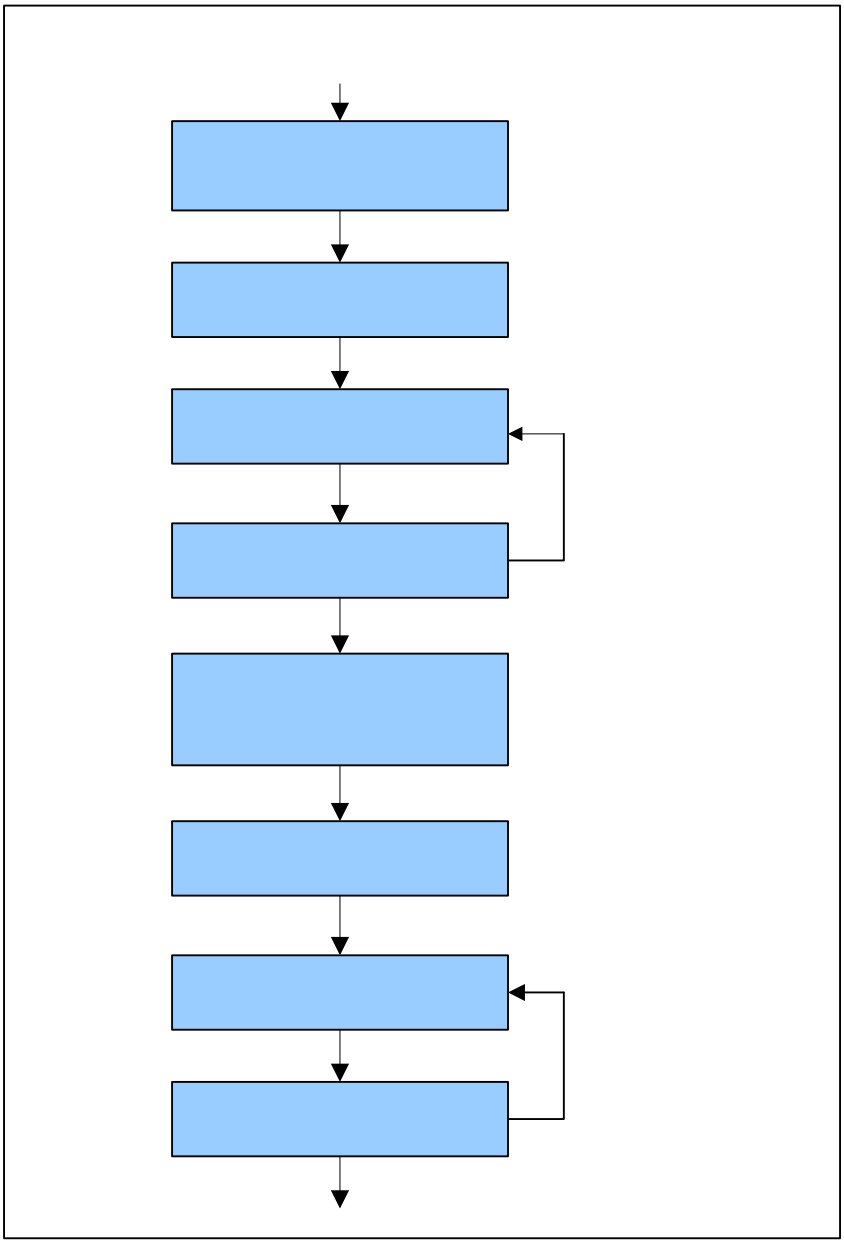

Figura 43 – Fluxograma da geração do arquivo BVH

O arquivo BVH tem como vantagem a descrição hierárquica do esqueleto em fora de grafo de cena. Grafo de cena é um grafo onde cada nó (no caso, uma articulação do ator), com exceção do nó raiz, está ligado a um nó pai, e as transformações aplicadas sobre um nó pai são conseqüentemente aplicadas nos nós filhos. Conforme citado anteriormente, a utilização de grafo para representar o esqueleto humano permite que dois movimentos sejam combinados através da multiplicação das matrizes. A Figura 44 mostra a estrutura convencional de um esqueleto simplificado (com apenas quinze articulações) e a Figura 45 mostra o mesmo esqueleto representado como grafo.

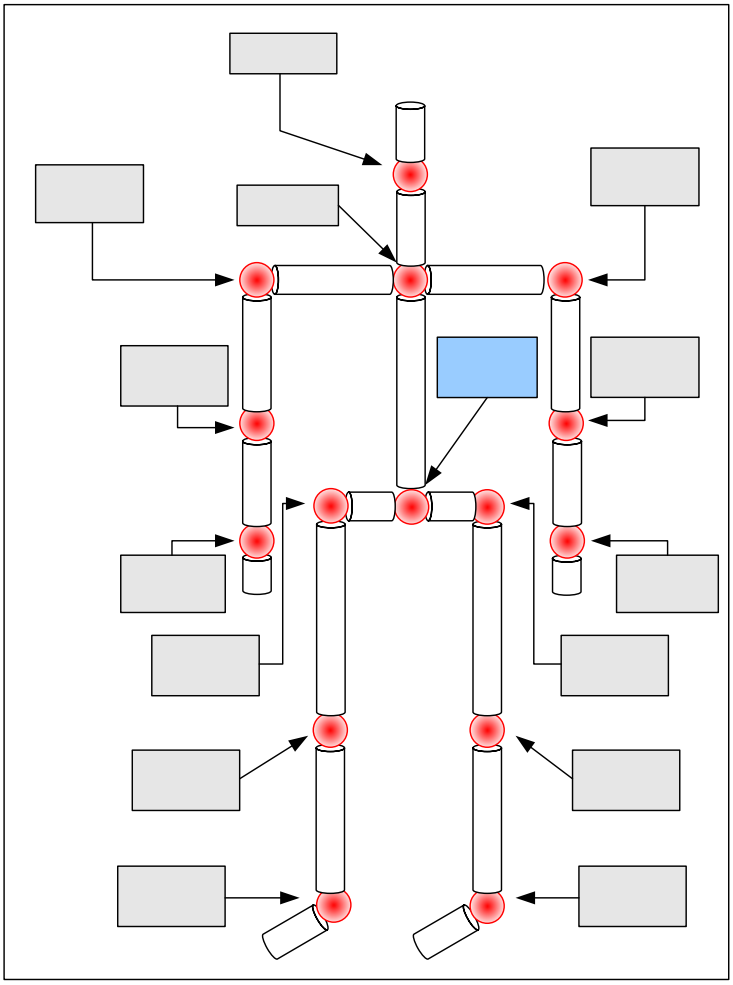

Figura 44 – Esqueleto em estrutura convencional

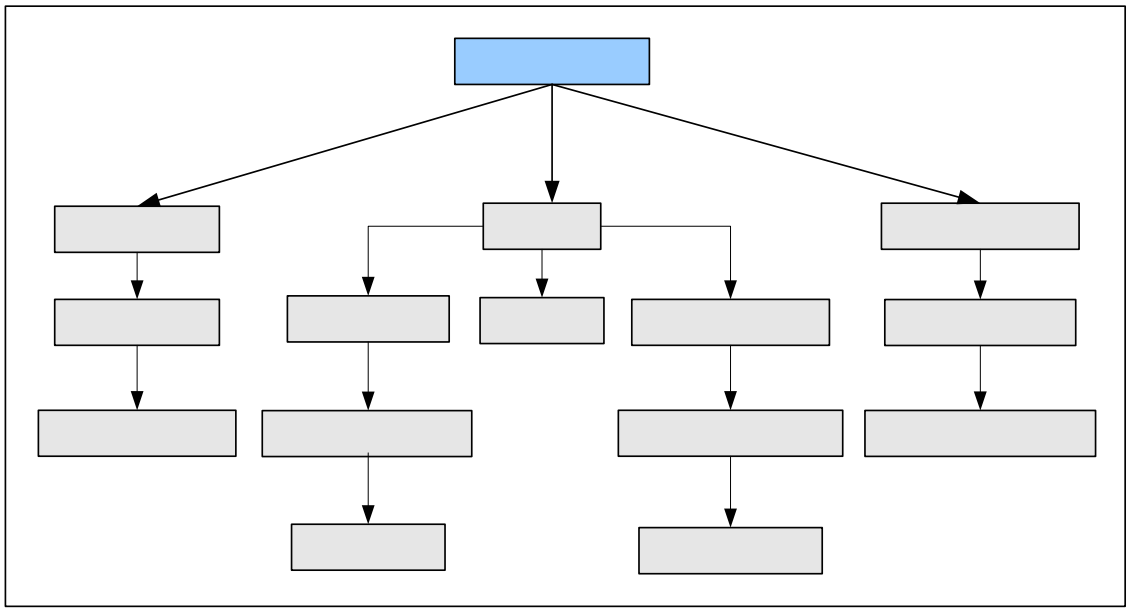

Figura 45 – Esqueleto em estrutura de grafo

### 4.1.2 Conversão de formato

O protótipo de Conversão de Formatos tem como entrada um arquivo MCP e como saída um arquivo BVH. Para fazer essa conversão, é necessário obter os dados do arquivo MCP e escrevê-los no arquivo BVH de acordo com a sintaxe deste formato.

Para a escrita no formato BVH, são utilizadas as funções da biblioteca libBvh. Entretanto, as funções para escrever a posição inicial do esqueleto e a sua hierarquia foram reimplementadas, visto que o formato MCP já apresenta a hierarquia, não sendo necessário fazer a conversão de sistema de coordenadas globais para sistema de coordenadas locais. A Figura 46 mostra o fluxograma simplificado da conversão de formatos.

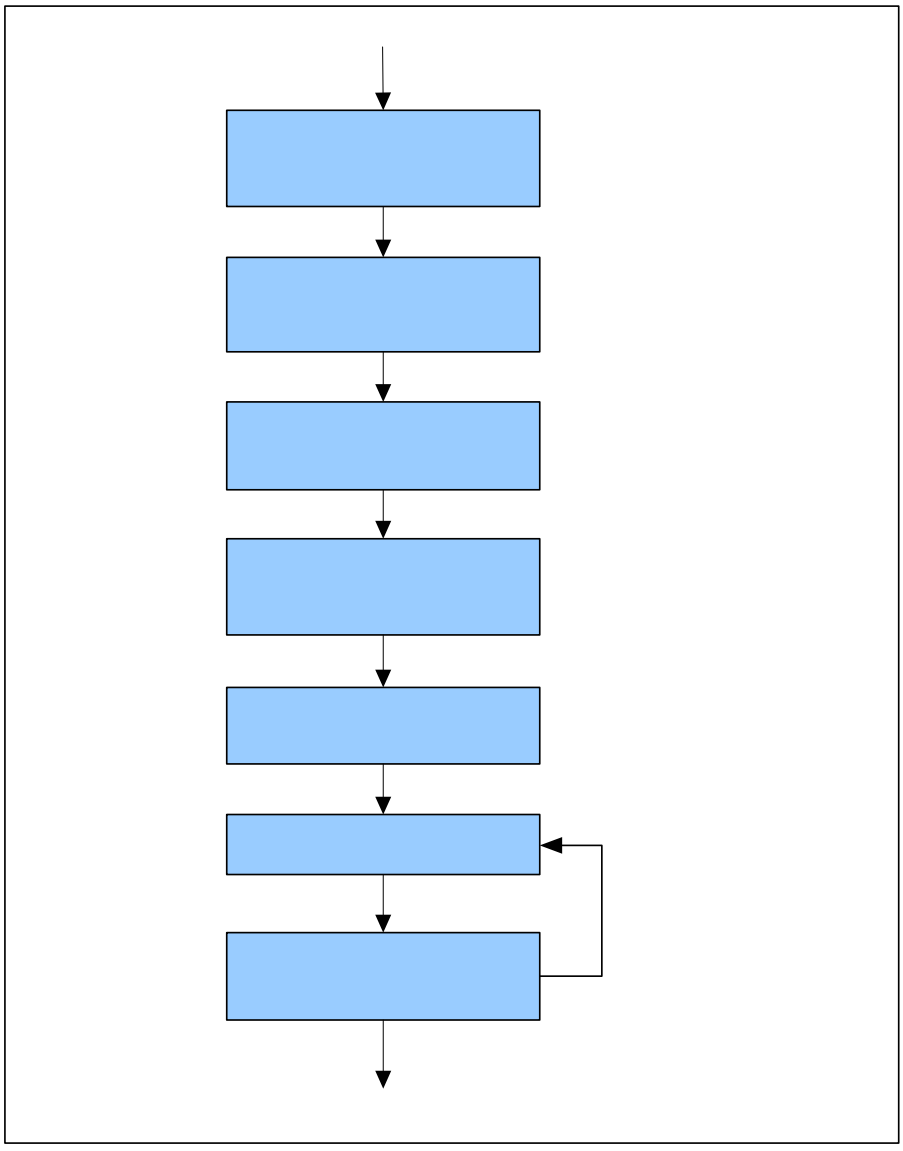

Figura 46 – Fluxograma da conversão de formatos

A Figura 47 mostra a interface do protótipo de conversão de formatos.

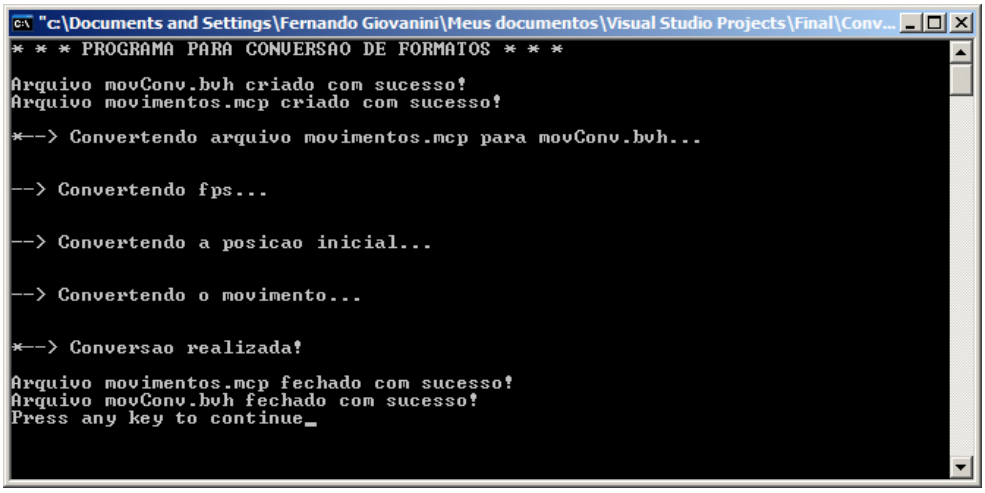


Figura 47 – Interface do protótipo de conversão de formatos

## 4.2 Estrutura

As funções e estruturas de dados foram divididas em seis módulos principais. Esses módulos interagem diretamente com outras bibliotecas indispensáveis para o projeto, como ARToolkit, *Open Graphics Library* (OpenGL) (OPENGL, 2007) , *Graphical Library User Interface* (GLUI) (GLUI, 2007) e OpenGL *Utility Toolkit* (GLUT) (GLUT, 2007). Além disso, o ARToolkit ainda utiliza funções da biblioteca *DirectShow Vídeo Processing Library* (DSVL) para a captura e processamento da imagem da câmera. O desenvolvimento em módulos, em que funções relacionadas e as estruturas utilizadas por elas são agrupadas no mesmo arquivo, foi escolhido devido às vantagens desta abordagem, como:

- facilidade para a reutilização do código
- maior rapidez na busca de erros de programação e
- possibilidade de compilar os módulos separadamente, reduzindo o tempo gasto na compilação.

A Figura 48 apresenta a estrutura do projeto, que possui 3 camadas principais. A camada de mais alto nível é a Camada de Aplicação, onde estão os protótipos apresentados anteriormente. Esses protótipos são:

- **Programa para Captura de Movimentos:** onde é realizada a calibração do esqueleto virtual, a captura do movimento, o armazenamento em formato BVH e MCP;
- **Programa para a Conversão do formato MCP para BVH:** neste programa é realizada a conversão do formato MCP para o formato BVH;

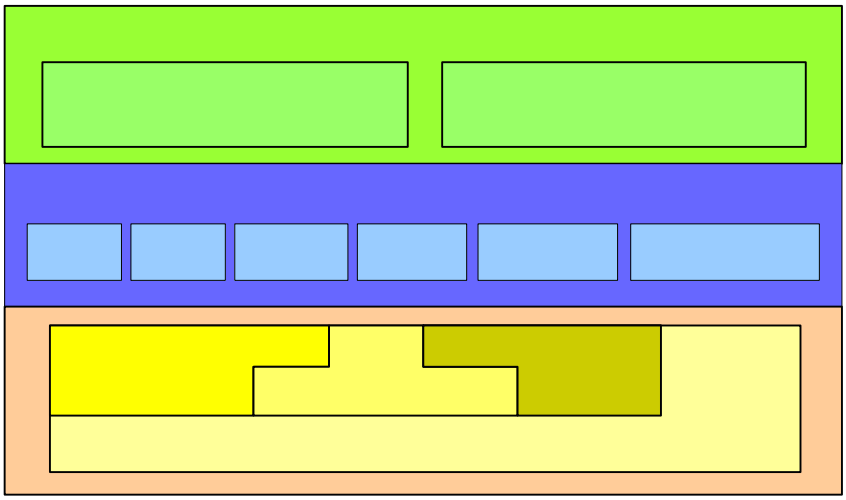

Figura 48 – Estrutura do projeto

Na Camada de Suporte à Captura de Movimentos estão os módulos que provêem as funções e estruturas de dados utilizadas na calibração do esqueleto, na captura dos movimentos, na geração e no armazenamento dos arquivos de movimento e na conversão de formatos de arquivo. Devido ao desenvolvimento modular, algumas das bibliotecas utilizam funções de outras, conforme descrito na Figura 49. Esses módulos implementados são:

- **libArq:** biblioteca em que estão as funções e estruturas de dados responsáveis pela manipulação dos arquivos do tipo texto e binário;
- **libMoCap:** biblioteca com funções e estruturas de dados para captura do movimento, da posição inicial e da taxa de fps;
- **libBvh:** encapsula as funções que geram e escrevem os dados no arquivo BVH;
- **libMcp:** nessa biblioteca estão as funções e estruturas para manipular os arquivos MCP;
- **libMat:** na biblioteca libMat estão as funções que realizam cálculos matemáticos;

Camada

Progr

Camada

- **libMcpToBvh:** biblioteca com as funções de conversão do formato MCP para BVH

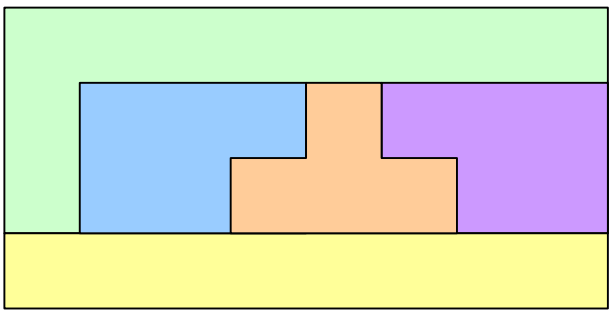

Figura 49 – Estrutura dos módulos no programa de CM

E por fim, na terceira camada estão as funções disponibilizadas pelo Sistema Operacional e pelas bibliotecas necessárias para o projeto, como ARToolkit, OpenGL, entre outras.

## 4.3 Módulos de suporte

Conforme citado anteriormente, os módulos de suporte oferecem aos protótipos as funções e estruturas de dados necessárias para a captura de movimentos e a conversão entre MCP e BVH.

### 4.3.1 Módulo LibArq

A biblioteca **libArq** encapsula as funções e a estrutura de dados responsáveis pela manipulação dos arquivos criados e abertos pelos protótipos desenvolvidos. A estrutura que contém as informações dos arquivos é a *struct* `infoArquivo` representada na Figura 50.

```
01 struct infoArquivo{
02     char caminho[100];
03     char nome[23];
04     FILE **ponteiro;
05 };
```

Figura 50 – *struct* infoArquivo

A *struct* `infoArquivo` contém as variáveis:

- `caminho`: um vetor de 100 elementos do tipo `char` que armazena o caminho do arquivo que será manipulado;

- `nome`: vetor que armazena o nome do arquivo;
- `ponteiro`: um ponteiro para um ponteiro do tipo `FILE` que se refere ao local da memória onde será alocado o arquivo.

As funções da biblioteca **libArq** permitem: fechar, apagar, abrir ou criar um arquivo, e inicializar a *struct* `infoArquivo`. As funções para abrir/criar, apagar e fechar um arquivo fazem uso de funções cujas interfaces estão descritas no cabeçalho stdio.h, mas que verificam se a operação foi realizada com sucesso e retorna o resultado para o usuário. A Figura 51 contém o código da função `criaArquivo`.

```
01 void criaArquivo(struct infoArquivo arquivo, char *permissão){
02     char caminhoArquivo[123];
03
04     strcpy(caminhoArquivo,arquivo.caminho);
05     strcat(caminhoArquivo,arquivo.nome);
06
07     if((*arquivo.ponteiro = fopen(caminhoArquivo,permissão)) == NULL){
08         printf("ERRO ao criar o arquivo %s!\n",arquivo.nome);
09         exit(1);
10     }
11     else
12         printf("Arquivo %s criado com sucesso!\n",arquivo.nome);
13 }
```

Figura 51 – Função criaArquivo

A função `criaArquivo` utiliza como parâmetros uma variável do tipo `infoArquivo` e a permissão para a criação ou abertura do arquivo. A variável `caminhoArquivo`, criada na linha 2, irá armazenar o caminho completo para o arquivo que será manipulado, que é a concatenação das variáveis caminho e nome contidas na *struct* `infoArquivo`. Na linha 7 o programa tenta criar ou abrir o arquivo e verifica o valor retornado por `fopen`. Caso não seja possível realizar essa operação, um erro será retornado para o usuário e o programa será encerrado.

As outras funções contidas na biblioteca **libArq** funcionam de maneira parecida, sempre testando as operações realizadas e retornando a mensagem para o usuário.

### 4.3.2 Módulo LibMoCap

O módulo **libMoCap** encapsula as funções e *structs* relacionadas à captura de movimento. Essa biblioteca pode ser dividida em quatro partes: ler e armazenar os dados sobre a posição inicial do esqueleto, ler e armazenar os dados sobre o movimento do ator, ler e armazenar os dados sobre taxa de fps, número de *frames* e cálculos relacionados à hierarquia do esqueleto.

Para a posição inicial do esqueleto são utilizadas as *structs* `articulacaoInicial` e `posicaoInicial`, representadas na Figura 52. A struct `posicaoInicial` é composta por um vetor do tipo `articulacaoInicial` com o seu número de elementos definido pela constante `MAX_JUNTAS`. A *struct* `articulacao` é composta pelos campos:

- **`posicao`:** a posição do marcador em relação à câmera nos eixos X, Y e Z;
- **`deslocamento`:** o deslocamento entre o marcador e a articulação representada por ele;
- **`pai`:** identifica qual o pai do nó;
- **`endEffector`:** utilizado pelo arquivo BVH para determinar se o nó é uma folha da árvore;
- **`visitado`:** utilizado quando a árvore hierárquica é percorrida na geração do arquivo BVH para verificar se o nó já foi escrito no arquivo e;
- **`nome`:** o nome do segmento, utilizado no arquivo BVH para identificar os nós.

```
01 struct articulacaoInicial{
02     double posição[3];
03     double deslocamento[3];
04     int pai;
05     int endEffector;
06     int visitado;
07     char nome[20];
08 };
09     }
10 struct posicaoInicial{
11     articulacaoInicial junta[MAX_JUNTAS];
12 };
```

Figura 52 – Estruturas de dados para a posição inicial do ator

O valor do vetor `posicao`, pertencente à *struct* `articulacaoInicial` é obtido no início do processo de captura dos movimentos, que será explicado mais adiante, na matriz de transformação fornecida pelas funções `arGetTransMat()` ou `arGetTransMatConv()`da biblioteca ARToolkit. A diferença entre essas funções é que a matriz de transformação fornecida pela segunda função é calculada com base na posição anterior do marcador, fazendo com que aumente a estabilidade da posição da junta. A Figura 53 mostra o processo de captura do valor do vetor `posicao`.

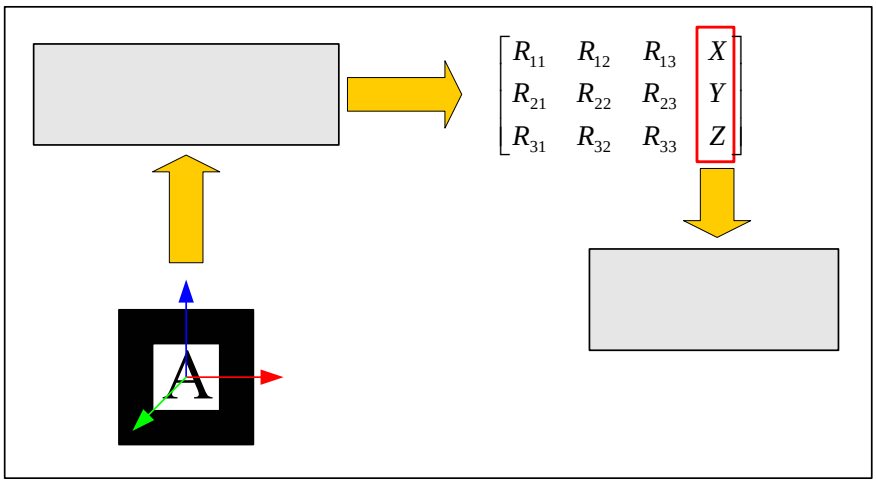


Figura 53 – Captura da posição inicial das juntas do esqueleto

A variável `pai`, pertencente à *struct* `articulacaoInicial`, conforme explicado, armazena o valor do índice do elemento `pai` de determinada articulação no vetor `junta`. Esse valor é necessário para montar a estrutura do grafo de cena que representa o esqueleto do ator.

Os outros valores armazenados nas variáveis do tipo `articulacaoInicial` também são obtidos nos protótipos, que serão descritos adiante. Após esta etapa, os valores são armazenados em um arquivo temporário. A função responsável por escrever os valores nesse arquivo temporário é a `escrevePosicaoInicial`, apresentada na Figura 54. Os parâmetros passados para essa função são uma variável do tipo `infoArquivo` (*struct* descrita no módulo libArq) e uma variável do tipo `posicaoInicial`. Na linha 4 o arquivo temporário é criado com permissão “`w+b`”, ou seja, um arquivo do tipo binário com permissão de escrita,

utilizando a função `criaArquivo` do módulo libArq, e nas linhas de 6 a 9 a variável do tipo `posicaoInicial` é gravada no arquivo especificado na variável `infoArquivo`.

```
01 int escrevePosicaoInicial(struct infoArquivo arquivo,
02                           struct posicaoInicial frame){
03
04     criaArquivo(arquivo, “w+b”);
05
06     if(fwrite(&frame,sizeof(struct posicaoInicial),
07               1,*arquivo.ponteiro)!=1){
08         printf(“ERRO ao escrever arquivo de posição inicial!\n”);
09         return 0;
10     }
11
12     fechaArquivo(arquivo);
13
14 return 1;
15 }
```

Figura 54 – Função escrevePosicaoInicial

Conforme citado anteriormente, o módulo **libMoCap** também manipula os dados relacionados ao movimento realizado pelo ator. As estruturas de dados necessárias para o armazenamento desses movimentos são `articulacaoMovimentos` e `posicaoMovimentos`, mostradas na Figura 55.

```
01 struct articulacaoMovimentos{
02     double posição[3];
03     double rotação[3];
04 };
05
06 struct posicaoMovimentos{
07     articulacaoMovimentos junta[MAX_JUNTAS];
08 }
```

Figura 55 – Estrutura de dados para o movimento realizado pelo ator

A *struct* `articulacaoInicial` contém dois vetores de três elementos do tipo *double*. O vetor `posicao` armazena a posição da junta do ator em um frame capturado e a *struct* `posicaoMovimentos` é composta por um vetor do tipo `articulacaoMovimentos` e seu tamanho é definido pela constante `MAX_JUNTAS`, assim como a *struct* `posicaoInicial`.

Os valores das variáveis *structs* acima são obtidos no programa para captura dos movimentos do ator. A cada *frame* capturado, esses dados são obtidos e armazenados em um arquivo binário temporário onde cada registro desse arquivo corresponde a um *frame* da captura, representado na Figura 56.

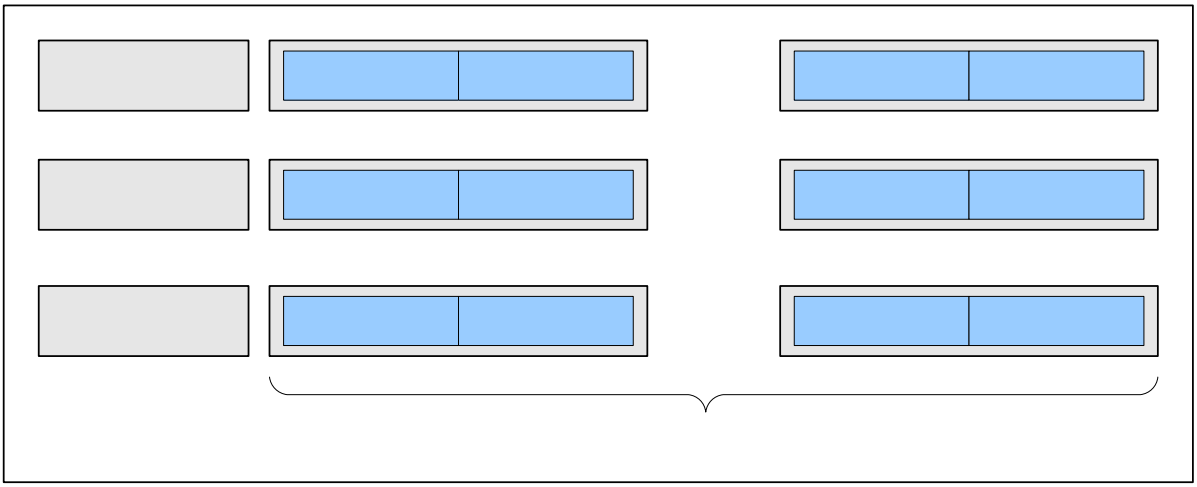

Figura 56 – Representação do arquivo temporário de movimentos

Os dados armazenados nos vetores `posicao` e `rotacao` da *struct* `articulacaoMovimento`, que por sua vez é armazenada no arquivo temporário de movimento, descrito na Figura 56, assim como os dados do vetor `posicao` da *struct* `posicaoInicial`, são obtidos na matriz de transformação fornecida pelas funções `arGetTransMat` e `arGetTransMatConv`, disponibilizadas pela biblioteca ARToolkit. Além do vetor de translação, essa matriz de transformação fornece uma matriz de rotação (Figura 57) que equivale à multiplicação das matrizes de rotação nos eixos X (eq. 1), Y (eq. 2) e Z (eq. 3). As eq. 4 descreve a ordem utilizada pela biblioteca ARToolkit para a multiplicação das matrizes, e as eq. 5 e eq. 6 apresentam o resultado dessa multiplicação

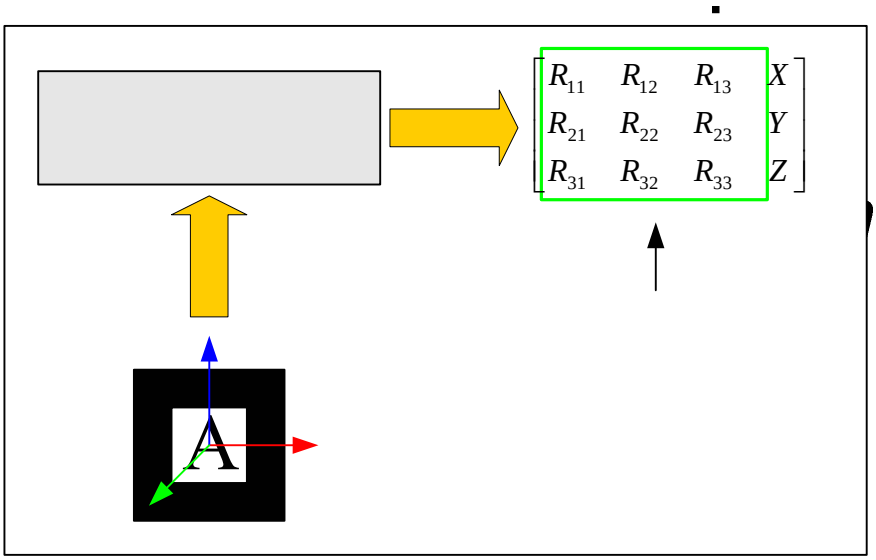


Figura 57 – Obtenção da matriz de rotação

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \quad \text{(eq. 1)}$$

$$M_Y = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \quad \text{(eq. 2)}$$

$$M_Z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{(eq. 3)}$$

$$M_R = M_X M_Y M_Z \quad \text{(eq. 4)}$$

$$M_R = \begin{bmatrix} \cos(\alpha)\cos(\beta) & \cos(\alpha)\sin(\beta)\sin(\theta) - \sin(\alpha)\cos(\theta) & \cos(\alpha)\sin(\beta)\cos(\theta) + \sin(\alpha)\sin(\theta) \\ \sin(\alpha)\cos(\beta) & \sin(\alpha)\sin(\beta)\sin(\theta) + \cos(\alpha)\cos(\theta) & \sin(\alpha)\sin(\beta)\cos(\theta) - \cos(\alpha)\sin(\theta) \\ -\sin(\beta) & \cos(\beta)\sin(\theta) & \cos(\theta)\cos(\theta) \end{bmatrix} \quad \text{(eq. 5)}$$

$$M_R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \quad \text{(eq. 6)}$$

Para obter as rotações de cada eixo em ângulos de Euler, a matriz de transformação disponibilizada pelo ARToolkit é passada como parâmetro para a função `arUtilMat2QuatPos`, também disponibilizada pela biblioteca. Essa função converte a matriz de transformação em um vetor com a posição do marcador e um quatérnio (eq. 7) que armazena a rotação.

$$Q = [q_0, q_1, q_2, q_3] \quad \text{(eq. 7)}$$

Após a obtenção do quatérnio, é necessário extrair os ângulos de Euler. Para isso, foi utilizada a função `quaternioParaEuler`, implementada no módulo libMat. Essa função recebe o quatérnio como parâmetro e retorna, por referência, a rotação nos três eixos. As equações utilizadas para obter os ângulos das rotações em X, Y e Z são demonstradas nas eq. 8, eq. 9 e eq. 10, respectivamente.

$$X = \arctan(2 \cdot (q_0 \cdot q_1 + q_2 \cdot q_3), 1 - 2 \cdot (q_1^{\,2} + q_2^{\,2})) \quad \text{(eq. 8)}$$

$$Y = \arcsin(2 \cdot (q_0 \cdot q_2 - q_3 \cdot q_1)) \quad \text{(eq. 9)}$$

$$Z = \arctan(2 \cdot (q_0 \cdot q_3 + q_1 \cdot q_2), 1 - 2 \cdot (q_2^{\,2} + q_3^{\,2})) \qquad \text{(eq. 10)}$$

Os valores obtidos são os ângulos em radianos das rotações nos três eixos. Por fim, esses valores são transformados em graus e armazenados no vetor rotação da *struct* `articulacaoMovimentos`.

A biblioteca **libMoCap** também é responsável por armazenar os dados referentes à taxa de *frames* por segundo e o número de *frames* capturados, e para isso utiliza a *struct* `infoFps`, representada na Figura 58. Essa *struct* contém uma variável do tipo inteiro que armazena o número de *frames* capturados e uma variável do tipo *float* que armazena a taxa de fps.

```
01 struct infoFps{
02     int frames;
03     float fps[3];
04 };
```

Figura 58 – Representação da *struct* infoFps

Assim como na posição inicial do ator e nos movimentos realizados, a variável `infoFps` é armazenada em um arquivo temporário que será utilizado na geração dos arquivos BVH e MCP.

O módulo **libMoCap** também é responsável pelas funções que trabalham com a hierarquia do esqueleto, visto que a biblioteca OpenGL não trabalha com grafos de cena. A hierarquia é definida na *struct* `articulacaoInicial`, armazenando qual é o nó pai de determinado marcador. Entretanto, para que o arquivo BVH fosse criado de acordo com as suas especificações, tornou-se necessário percorrer o grafo, transformando as rotações de sistema de coordenadas globais para sistema de coordenadas locais.

Para obter a rotação de determinado eixo de uma junta em coordenadas locais, subtrai-se do valor da rotação neste eixo a soma das rotações nos mesmos eixos das juntas pais, em sistema de coordenadas locais. Para obter a rotação dos outros eixos em coordenadas locais, aplica-se o mesmo cálculo. A Figura 59 ilustra esse cálculo.

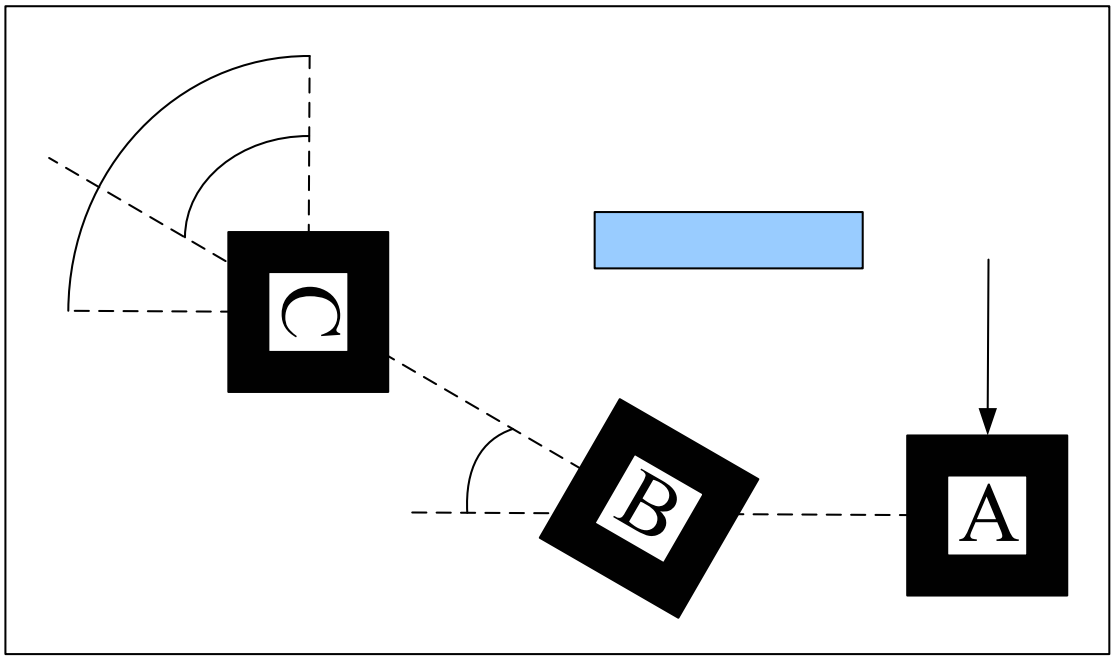


Figura 59 – Cálculo para transformação de sistemas dos coordenadas

### 4.3.3 Módulo LibBvh

O módulo **libBvh** é responsável pela criação do arquivo BVH. As funções contidas neste módulo possibilitam:

- escrever variáveis do tipo inteiro, char, *double*, *float*;
- identar o arquivo, conforme recomendado por Meredith e Maddock (2001);
- escrever os movimentos realizados;
- escrever a posição inicial do ator;
- escrever os dados de fps e número de frame.

Dentre as funções pertencentes ao módulo **libBvh** está a `escreveBVH`, que recebe como parâmetros quatro *structs* do tipo `infoArquivo`: arquivo BVH, arquivo da posição inicial do ator, arquivo com o movimento realizado e o arquivo com a taxa de fps e o número de *frames*. A partir destes três últimos arquivos, a função irá gerar o arquivo BVH.

Devido à característica hierárquica do arquivo BVH, durante a escrita da posição inicial capturada é necessário calcular a posição dos nós filhos em relação aos pais, pois assim como na rotação, estes dados estão em coordenadas globais. Este cálculo deve ser realizado

para os três eixos cartesianos. A eq. 11 demonstra como foram feitos os cálculos para o eixo x em uma articulação identificada por idNo.

$$eixoX = junta[idNo].posicao[0] + junta[idNo].deslocamento[0] - junta[idPai].posicao[0] + junta[idPai].deslocamento[0] \quad \text{(eq. 11)}$$

### 4.3.4 Módulo LibMcp

Este módulo foi desenvolvido para manipular os arquivos de formato MCP criado para este trabalho. Este formato de arquivo tem como principais vantagens:

- possuir estrutura hierárquica para facilitar a exportação para o formato BVH e
- ser um arquivo binário, que facilita a sua leitura quando realizada por outro programa, como, por exemplo, um visualizador ou editor de movimentos

Assim como BVH, o arquivo MCP é composto por informações sobre número de *frames*, hierarquia e posição das juntas no esqueleto e movimentos realizados. Porém, diferentemente do BVH, esse arquivo tem informações do número de juntas do esqueleto e a taxa de *frames* por segundo.

Entre as estruturas que compõem o arquivo MCP estão a `jointMCP` e `juntaMCP`, representadas na Figura 60. A *struct* `juntaMCP` é um vetor do tipo `jointMCP`, com o número de elementos igual ao número máximo de juntas. A *struct* `jointMCP` tem os seguintes campos:

- **`nome:`** nome da junta
- **`id:`** identificação da junta
- **`paiId:`** identificação do nó pai
- **`posicao:`** posição da junta em relação ao nó pai
- **`endSite:`** caso o nó contenha um *end site*, o valor desse campo é 1
- **`endSiteDeslocamento:`** posição do *end site* em relação ao nó pai

```
01 struct jointMCP{
02     char nome[20];
03     int id;
04     int paiId;
05     double posição[3];
06     int endSite;
07     double endSiteDeslocamento[3];
08 };
09
10 struct juntaMCP{
11     jointMCP juntas[MAX_JUNTAS];
12 };
```

Figura 60 – Representação das *structs* jointMCP e juntaMCP

Para armazenar o movimento realizado pelo ator, foram criadas *structs* descritas na Figura 61. A estrutura `movimentoMCP` é composta por um campo do tipo `rootMCP` e um vetor do tipo `nodeMCP` com o número de elementos igual a `MAX_JUNTAS` menos um elemento. A *struct* `rootMCP` armazena a posição e a rotação da junta raiz, mas a *struct* `nodeMCP`, que armazena a rotação dos demais nós, armazena somente a rotação dos nós devido à característica hierárquica do formato MCP. Para cada *frame* capturado, um registro da *struct* `movimentoMCP` é escrito no arquivo.

```
01 struct rootMCP{
02     double posicao[3];
03     double rotacao[3];
04 };
05
06 struct nodeMCP{
07     double rotacao[3];
08 };
09
10 struct movimentoMCP{
11     rootMCP raiz;
12     nodeMCP no[MAX_JUNTAS-1];
13 };
```

Figura 61 – Representação das *structs* jointMCP e juntaMCP

O arquivo MCP ainda armazena informações sobre o número de juntas capturadas (tipo inteiro), a quantidade de *frames* (tipo inteiro) e a taxa de fps (tipo *float*). A Figura 62 ilustra a estrutura do arquivo MCP.

As funções oferecidas pela biblioteca **libMcp** são similares às da libBvh, porém, em vez de manipular um arquivo tipo texto, elas lidam com arquivo binário.

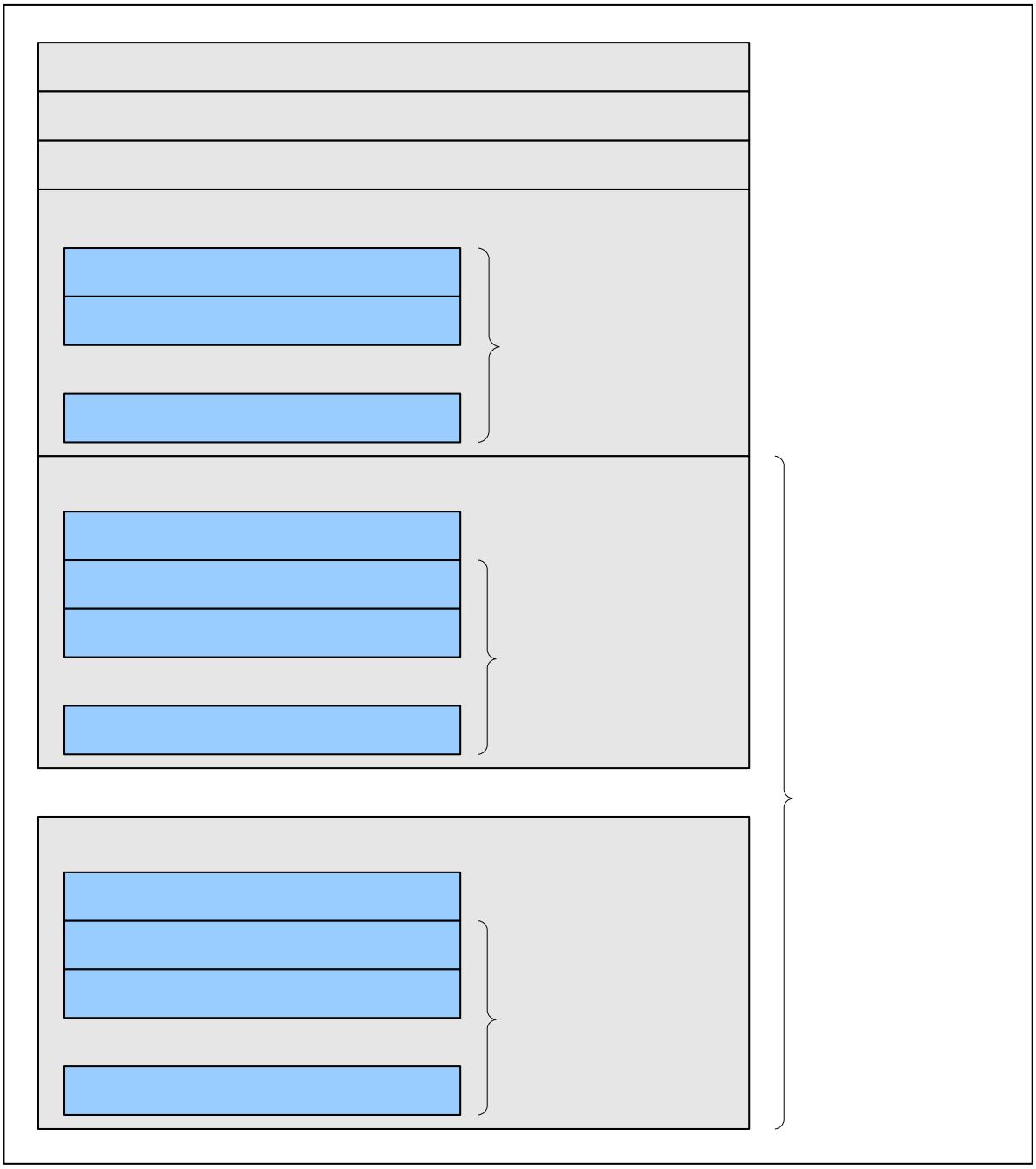

Figura 62 – Estrutura do arquivo MCP

### 4.3.5 Módulo LibMcpToBvh

Conforme o nome indica, esta biblioteca provê funções para realizar a conversão do formato MCP para BVH, entretanto, somente uma função, representada na Figura 63, é disponibilizada no cabeçalho libMcpToBvh.h para facilitar o seu uso. Essa função, `converteMcpToBvh`, realiza basicamente os seguintes passos:

- linhas 13 a 16: lê do arquivo MCP o número de juntas do esqueleto, o número de *frames* do movimento e a taxa de fps, e armazena em uma variável do tipo `infoFps`;
- linha 18 e 19: lê as informações da posição inicial contidas no arquivo MCP, armazena em uma variável do tipo `juntaMCP` e escreve a posição inicial no arquivo BVH ;
- linha 21: escreve no arquivo BVH as informações de número de *frames* e o tempo duração de cada *frame*;
- linha 23 a 26: lê os movimentos no arquivo MCP e escreve no arquivo BVH.

```
01 void converteMcpToBvh(struct infoArquivo arquivoMCP,
02                       struct infoArquivo arquivoBVH){
03     struct infoFps informacaoFps;
04     struct juntaMCP esqueletoMCP;
05     struct movimentoMCP movMCP;
06
07     int juntas,
08         frames;
09     float tempoFrame;
10
11     int contador;
12
13     juntas = lerIntMCP(arquivoMCP);
14     frames = lerIntMCP(arquivoMCP);
15     fps = lerFloatMCP(arquivoMCP);
16     converteInfoFps(&informacaoFps, frames, fps);
17
18     lerJuntaMCP(arquivoMCP,&esqueletoMCP);
19     escrevePosIniMcpToBvh(&arquivoBVH,esqueletoMCP,juntas);
20
21     escreveInfoFpsBVH(&arquivoBVH,informacaoFps);
22
23     for(contador = 0;contador < frames;contador++){
24         lerMovimentoMCP(arquivoMCP,&movMCP);
25         escreveMovIniMcpToBvh(&arquivoBVH,movMCP,esqueletoMCP,juntas);
26     }
27 }
```

Figura 63 – Função converteMcpToBvh

As estruturas de dados são as mesmas dos módulos libMcp e libMoCap, visto que são necessárias para a leitura do arquivo MCP e para a criação do arquivo BVH, ambas tarefas realizadas por funções disponíveis nas bibliotecas libMcp e libBvh.

## 4.4 Considerações finais

Neste Capítulo foi apresentada a descrição do sistema e implementação dos módulos e protótipo do sistema de CM.

Os dois protótipos apresentados foram o de Captura de Movimentos e o de Conversão de Formatos. No primeiro protótipo, as funções e estruturas de dados implementadas nos módulos de suporte foram utilizadas para realizar a captura dos movimentos do ator e armazená-las em formato BVH e MCP. O segundo protótipo realiza a conversão entre os formatos também utilizando as funções e estruturas de dados disponibilizadas nos módulos já citados.

Foram desenvolvidos seis módulos para o suporte à CM e conversão de formatos: **libBvh**, **libMcp**, **libMoCap**, **libArq**, **libMat** e **libMcpToBvh**. Em cada um desses módulos foram implementadas as funções e estruturas de dados relacionadas à tarefas específicas.

Os módulos **libBvh** e **libMcp** são responsáveis pela criação e edição dos arquivos BVH e MCP, respectivamente. O módulo **libMoCap** é responsável pelas funções e estruturas de dados da captura dos movimentos do ator. No módulo **libArq** estão as funções e estruturas de dados específicas para a criação, abertura e remoção dos arquivos temporários, BVH e MCP. No módulo **libMat** estão as funções matemáticas utilizadas nas conversões dos ângulos e sistemas de coordenadas, e no módulo **libMcpToBvh** estão as funções e estruturas para a conversão do arquivo em formato MCP para BVH.

# CAPÍTULO 5 - TESTES E ANÁLISE DOS RESULTADOS

Neste Capítulo são descritos os testes realizados com o protótipo de CM, o ambiente em que foram realizados e os *softwares* e *hardwares* utilizados, além de analisados os resultados obtidos. Os testes foram realizados a fim de obter as taxas de fps e a taxa de reconhecimento fornecida pelo ARToolkit, variando o número de marcadores.

## 5.1 Recursos utilizados

Para o desenvolvimento dos módulos e dos protótipos, foram utilizados os softwares e hardwares descritos nas subseções seguintes.

A Tabela 4 descreve os principais aplicativos utilizados na implementação dos módulos e protótipos implementados desenvolvidos.

Tabela 4 – Principais *Softwares* utilizados na implementação das bibliotecas e protótipos

| ***Software*** | **Versão** | **Descrição** |
|---|---|---|
| Microsoft Windows XP Professional SP2 | 2002 | Sistema Operacional |
| Microsoft Visual Studio .NET 2003 | 7.1.3088 | Ambiente de programação para linguagem C e C++ |
| ARToolkit | 2.72 | Biblioteca de RA |
| GLUI | 2.35 | Biblioteca gráfica para o desenvolvimento de Interfaces Gráficas em linguagem C e C++ |
| GLUT | 3.7.6 | Biblioteca gráfica com funções de criação e manipulação de janelas e monitoramento do teclado e *mouse* |
| DSVL | 0.0.8 | Biblioteca que controla e processa as imagens capturadas pela câmera de vídeo |
| OpenGL | 2.1 | Biblioteca gráfica para a renderização de objetos |

Os principais recursos de hardware utilizados na implementação do projeto foram:

- *Notebook* ACER TravelMate 4062 com 512 *megabytes* de memória RAM (*Random Access Memory*) sendo 64 megabytes compartilhados com a placa de vídeo *on-board* Intel 915GM e processador Intel Pentium M 1.73 GHz
- Câmera digital com modo *Webcam*, da marca Genius, modelo DV1210, com sensor CMOS de 5.1 *megapixels*, resolução de vídeo de 640 x 480 *pixels*, foco fixo e conexão USB

Além dos recursos de software e hardware já citados, também foram utilizados outros recursos materiais. A Figura 64 mostra o manequim articulado de 30 cm de altura utilizado para testar o sistema de CM implementado. Este manequim contém 14 articulações, é confeccionado em madeira e utilizado geralmente por desenhistas para capturar posições e proporções do corpo humano.

Figura 64 – Manequim articulado

Para a confecção dos marcadores foram utilizadas fitas elásticas de 2 cm de largura (para fixar no manequim articulado) e 3 cm (para fixar em um ator), e nesses elásticos foram

costuradas fitas macias de velcro com a mesma largura. Os marcadores do ARToolkit foram impressos em papel “sulfite” branco e colados em papel “paraná” com 2 mm de espessura, para que o marcador não se deforme durante o movimento. No outro lado do papel “paraná” foram fixadas as fitas ásperas do velcro, permitindo a junção das fitas elásticas com os marcadores. A Figura 65 mostra as fitas utilizadas e os marcadores fixados no manequim articulado, enquanto a Figura 66 mostra as fitas utilizadas no ator.

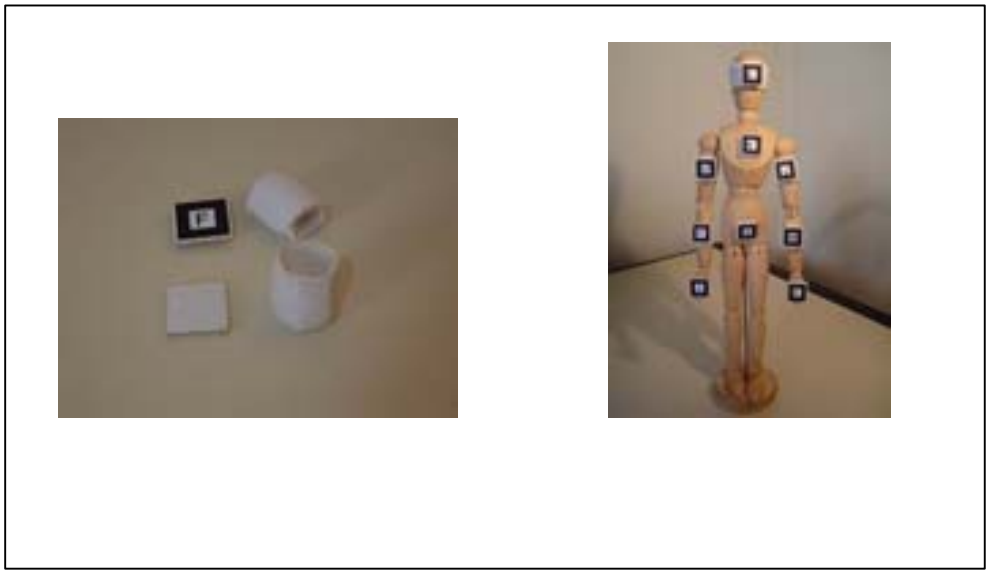

Figura 65 – Marcadores do manequim articulado

Figura 66 – Marcadores do ator

Para os testes, nove marcadores foram escolhidos para representar as juntas do ator e do manequim. Apesar do protótipo de CM permitir a captura dos movimentos do corpo

inteiro, somente os movimentos da parte superior do corpo foram capturados para os testes devido à resolução da câmera utilizada. A Figura 67 mostra os marcadores dispostos em grafo de acordo com a estrutura utilizada nos testes. A Tabela 5 descreve onde cada marcador foi posicionado e qual junta eles representam.

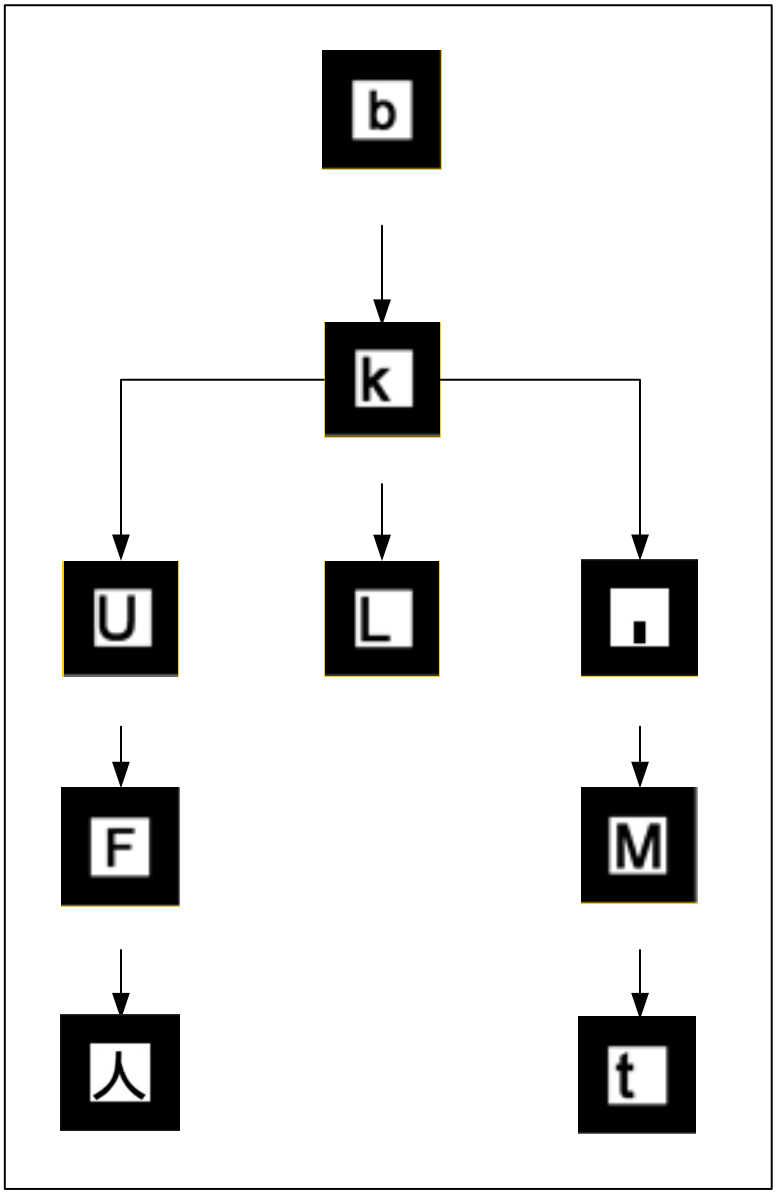


Figura 67 – Marcadores dispostos em forma de grafo de cena

Tabela 5 – Relação entre os marcadores, seus posições e juntas correspondentes

| ID | Marcadores | Localização | Junta correspondente |
|---|---|---|---|
| 1 |  | Quadril | Quadril |
| 2 |  | Tórax | Tórax |
| 3 |  | Cabeça | Pescoço |
| 4 |  | Braço direito | Ombro direito |
| 5 |  | Braço esquerdo | Ombro esquerdo |
| 6 |  | Antebraço direito | Cotovelo direito |
| 7 |  | Antebraço esquerdo | Cotovelo esquerdo |
| 8 |  | Mão direita | Pulso direito |
| 9 |  | Mão esquerda | Pulso esquerdo |

Os marcadores utilizados no manequim articulado e no ator possuem respectivamente 1,5 cm e 7 cm de largura.

## 5.2 Ambientes de teste

Para os testes, foram criados três ambientes, sendo um para os testes no manequim articulado, um para o ator e outro para os testes realizados em superfície plana. Nas subseções abaixo serão descritos os três ambientes.

### 5.2.1 Ambiente de teste para o manequim articulado

Os testes com o manequim articulado foram realizados em uma mesa com 70 cm de comprimento. Para medir as distâncias em que os movimentos do manequim articulado foram capturadas, foi colada uma fita métrica sobre a mesa. A câmera, posicionada diante do manequim, foi apoiada em um tripé a 84 cm do chão. Para a iluminação, uma luminária de 20

watts foi posicionada acima da câmera e voltada para o manequim. A Figura 68 apresenta o ambiente de testes descrito.

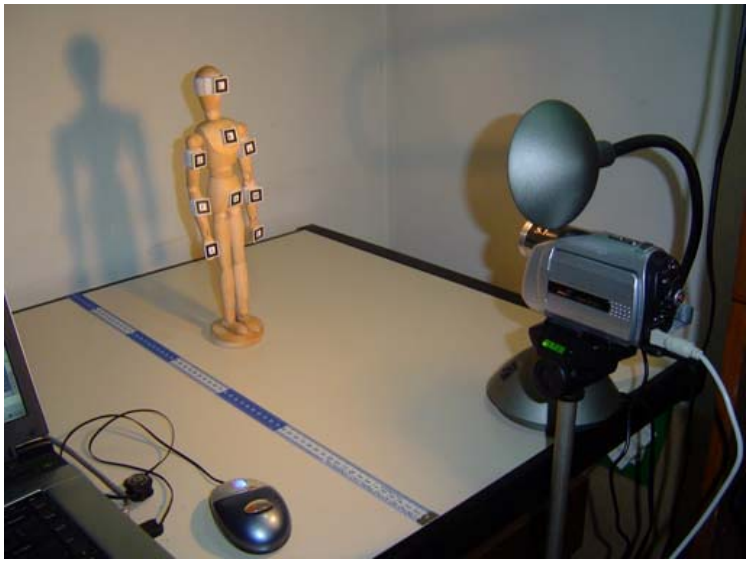

Figura 68 – Ambiente de testes para o manequim articulado

### 5.2.2 Ambiente de teste para o ator

Os testes com o ator foram realizados em uma área aberta com quatro metros de comprimento. Foram feitas marcas no chão para poder controlar a distância entre o ator e a câmera de vídeo, apoiada em um tripé a aproximadamente 120 centímetros do chão. A Figura 69 apresenta o ambiente de teste descrito acima. A iluminação utilizada foi constante e indireta

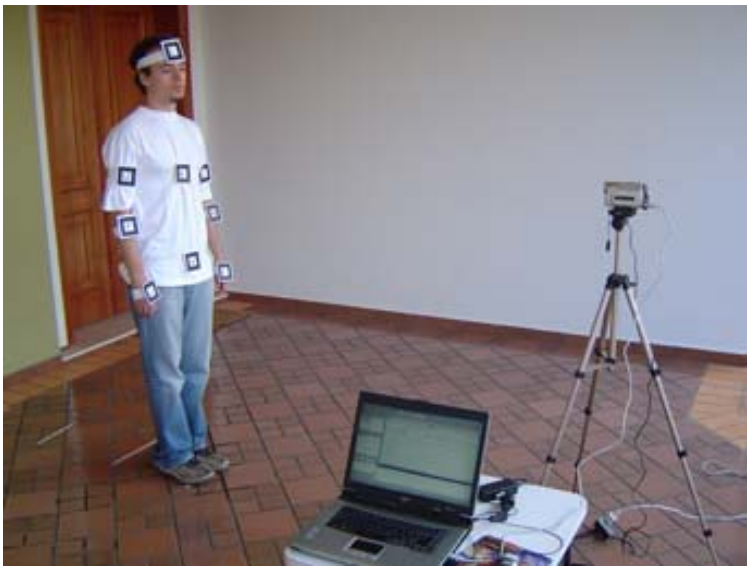

Figura 69 – Ambiente de testes para o ator

### 5.2.3 Ambiente de teste em superfície plana

Neste ambiente a câmera foi fixada a um tripé a 40 cm de altura em cima de uma mesa. A lente da câmera foi direcionada para a superfície da mesa, que foi revestida com uma folha de sulfite branca para aumentar o contraste com os marcadores. A iluminação foi feita com uma luminária de 20 watts.

Foram realizados dois tipos de testes neste ambiente, que serão descritos em subseções seguintes: um para verificar a taxa de fps com até 22 marcadores e outro para obter a velocidade angular máxima que o protótipo pode capturar. A Figura 70 apresenta o ambiente adaptado para os dois testes.

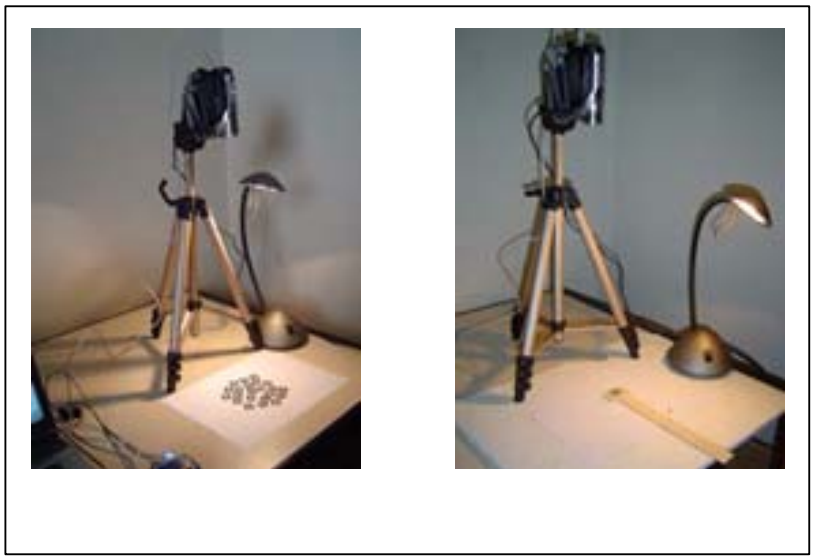

Figura 70 – Ambiente de testes em superfície plana

No teste para obter a taxa de fps para até 22 marcadores, estes foram posicionados sobre a folha de “sulfite” branca. No teste para obter a velocidade angular máxima, o marcador utilizado foi preso sobre um tira de papel “paraná” preso a uma placa de isopor para permitir a rotação. Ambos os ambientes foram iluminados por uma luminária de 20 watts.

## 5.3 Testes realizados

Foram realizados quatro tipos de teste:

- verificação da taxa de frames por segundo;

- precisão na captura da rotação das juntas;
- velocidade angular máxima e;
- reprodução do arquivo BVH gerado.

### 5.3.1 Verificação da taxa de *frames* por segundo variando o número de marcadores

Foram realizados testes para verificar a variação na taxa de fps do movimento capturado, sendo realizadas nove sessões de captura, e em cada sessão, efetuadas 10 capturas de aproximadamente trinta segundos. A cada sessão de captura foi adicionado um marcador, iniciando com um e finalizando com nove.

Os testes foram realizados no manequim articulado e em um ator. Nos testes no manequim articulado, ele foi posicionado a 40 cm da câmera, e nos testes do ator, foi posicionado a 186 cm da câmera. A Figura 71 apresenta o gráfico de desempenho dos testes realizados no manequim articulado e no ator. Os valores apresentados nos gráficos são as médias aritméticas dos valores obtidos nos testes.

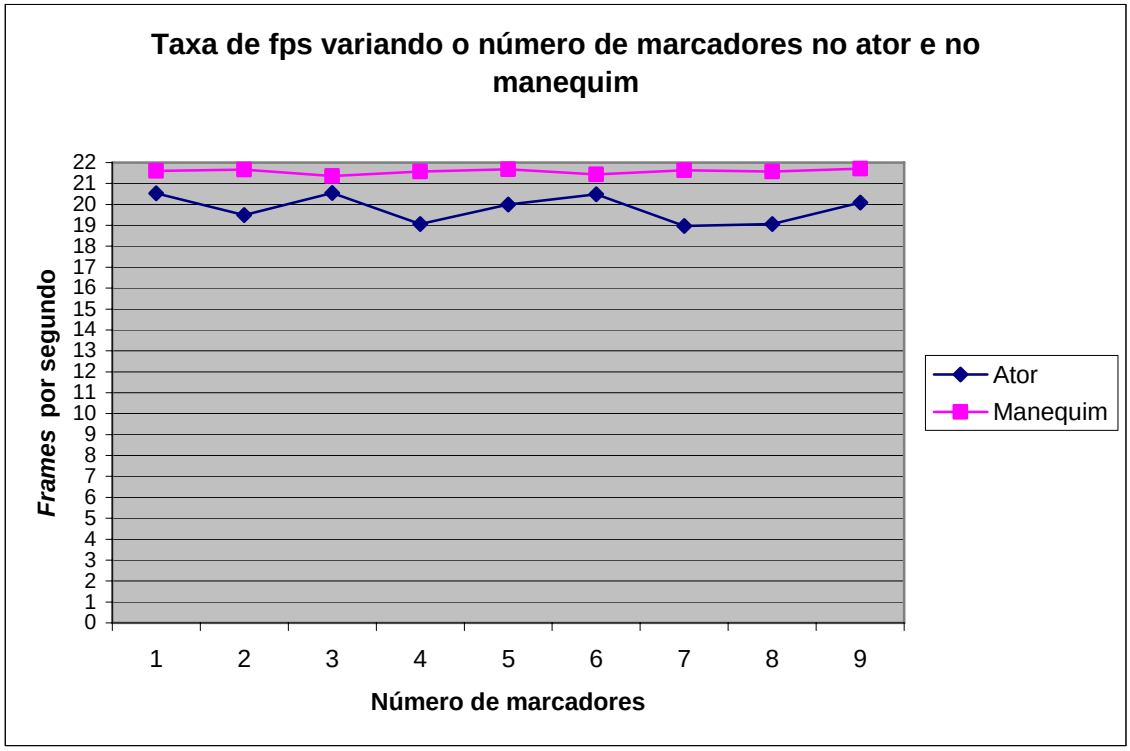


Figura 71 – Variação da taxa de fps de acordo com o número de marcadores

Percebe-se que a variação na taxa de fps do manequim articulado foi menor do que a variação da taxa de fps do ator. Além disso, a taxas de *frames* por segundo do manequim foram maiores do que as taxas obtidas nos testes do ator. A razão encontrada para isso foi a diferença nas condições de iluminação dos dois testes. Em ambos os testes, porém, a variação foi pequena, não passando de pouco mais de 1,56 *frames* por segundo no pior dos casos.

Pode-se também verificar que neste caso não há nenhuma relação entre o número de marcadores e a taxa de frames por segundo, ao contrário dos testes realizados no sistema de Lourenço (2004). Isto ocorre devido ao sistema desenvolvido pelo autor ter sido implementado com *hardwares* com menor capacidade de processamento.

Para comprovar os valores, um terceiro teste foi realizado. Neste, a câmera foi fixada em um tripé a 40 cm da superfície de uma mesa, e para a iluminação foi utilizada uma luminária de 20 watts. Foram realizadas 22 sessões de captura e em cada sessão foram realizadas 10 capturas de aproximadamente 30 segundos. A cada sessão foi adicionado um marcador, iniciando com um, mas ao contrário dos testes descritos anteriormente, finalizando com 22. Os marcadores utilizados mediam 1,5 cm (Figura 72).

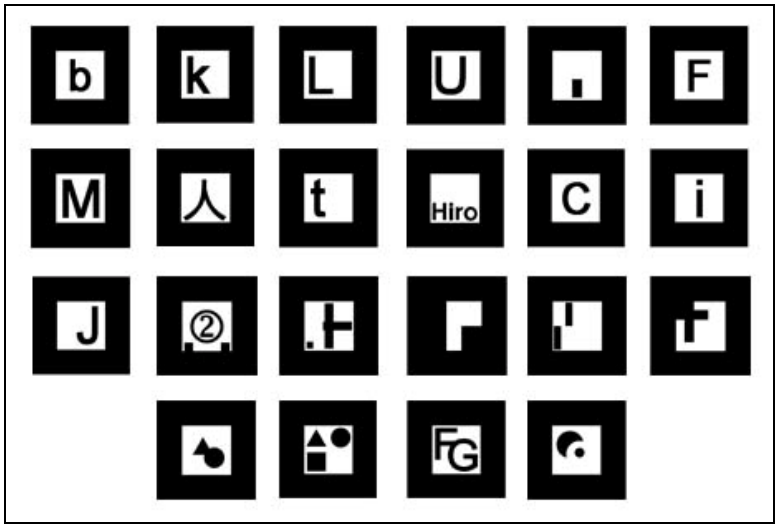


Figura 72 – Marcadores do teste da taxa de fps

O gráfico da Figura 73 apresenta o resultado obtido.

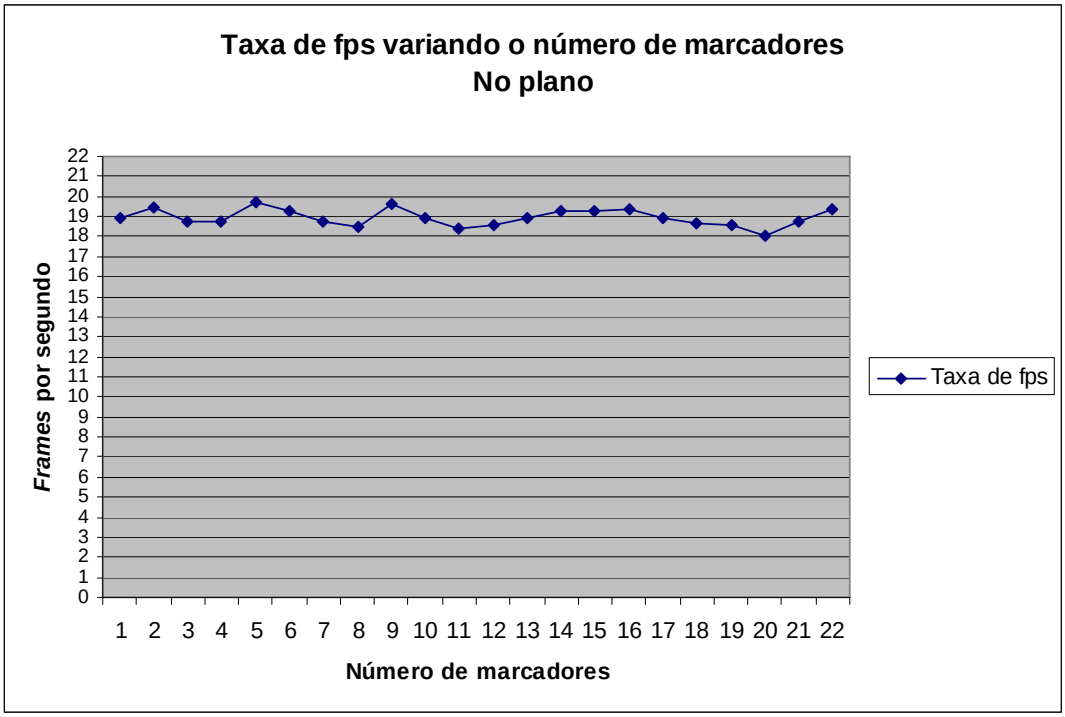


Figura 73 – Marcadores do teste da taxa de fps

Os resultados obtidos foram similares aos anteriores, demonstrando que até 22 marcadores não há mudança que altere a taxa de fps.

### 5.3.2 Precisão na captura da rotação das juntas

Para verificar a precisão na rotação de determinadas articulações, foram realizados testes em que somente determinada articulação era rotacionada no eixo Z e o valor da rotação era armazenado. Os valores obtidos permitem avaliar se a rotação capturada condiz com a realizada.

Foram rotacionadas as juntas correspondentes ao ombro direito, ombro esquerdo e pescoço, tanto no manequim articulado quanto no ator. Nos ombros, o valor do ângulo rotacionado foi de aproximadamente 90 graus, e no pescoço, o valor rotacionado foi de aproximadamente 45 graus para a direita e para a esquerda. As Figuras 74 e 75 descrevem os movimentos realizados para os ombros e o pescoço.

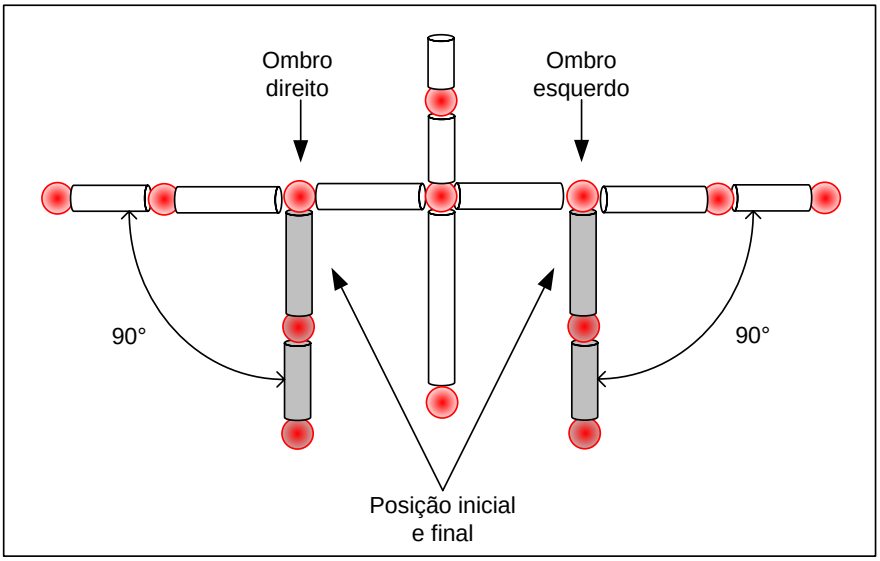


Figura 74 – Rotações realizadas nos ombros do ator e manequim articulado

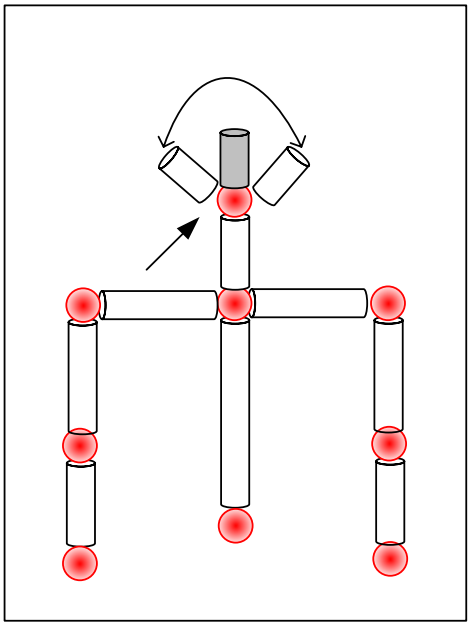

Figura 75 – Rotação realizada pescoço do ator e manequim articulado

Nos testes, o manequim articulado foi posicionado a 40 cm da câmera, enquanto no teste realizados com o ator, este ficou a 186 cm da câmera.

As Figuras 76, 77 e 78 mostram os gráficos com os movimentos das articulações do manequim articulado e do ator.

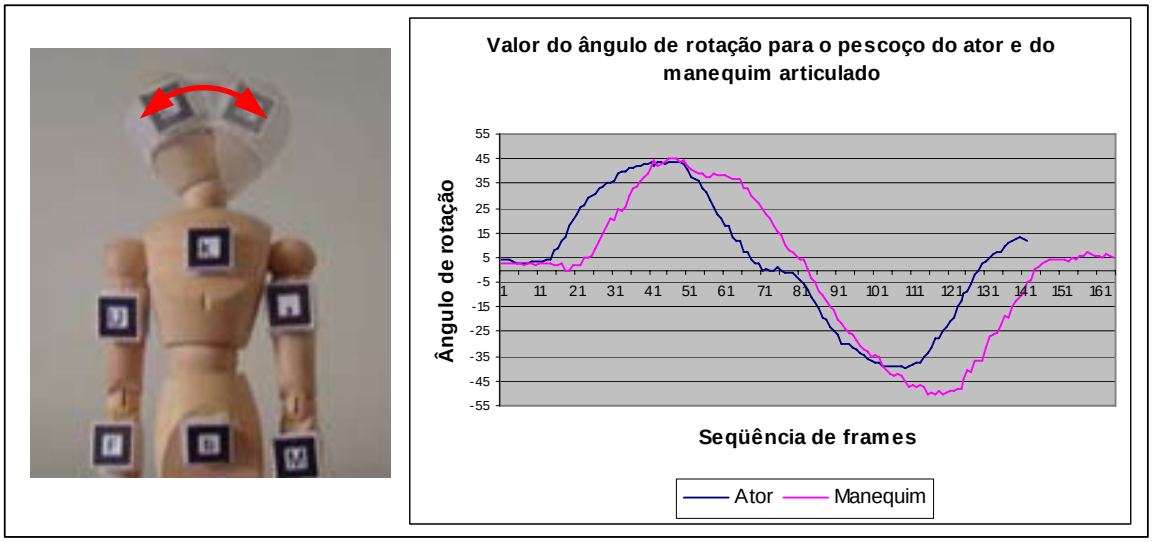


Figura 76 – Movimento do pescoço do ator e do manequim articulado

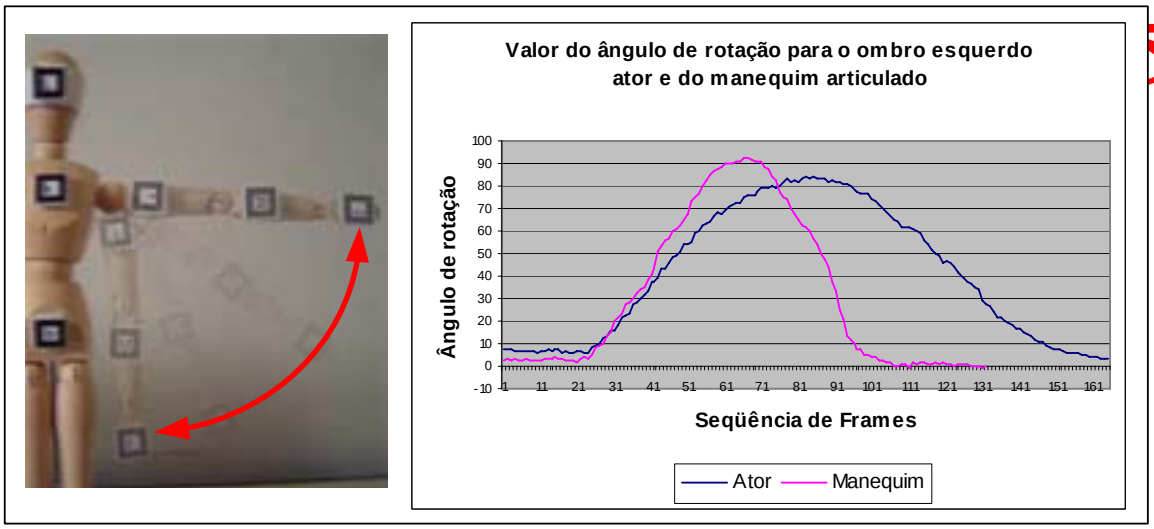


Figura 77 – Movimento do ombro direito do ator e do manequim articulado

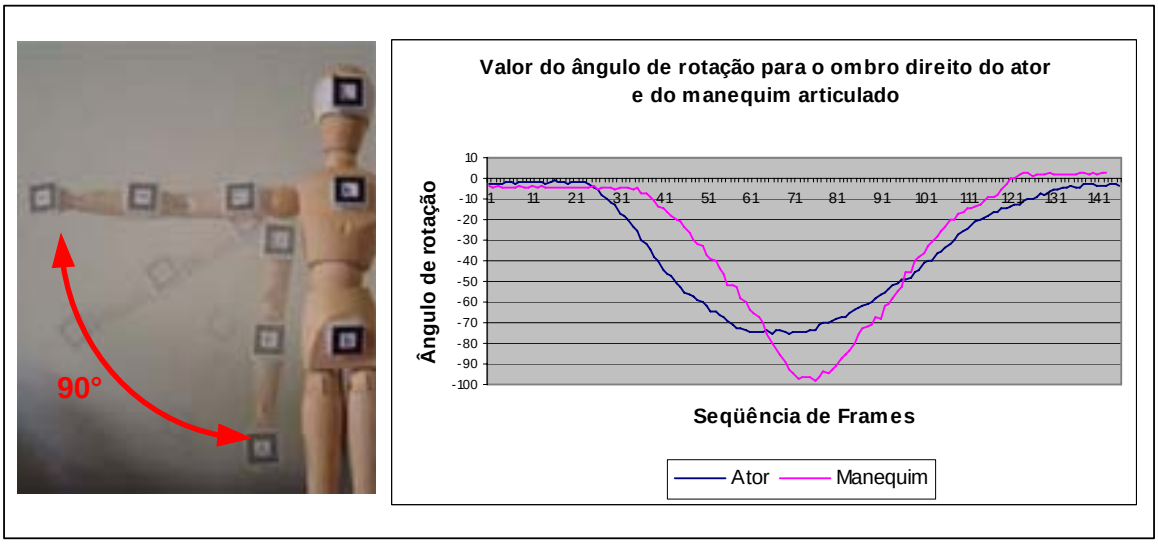


Figura 78 – Movimento do ombro esquerdo do ator e do manequim articulado

Analisando os gráficos apresentados, é possível perceber que os movimentos realizados pelo ator são mais naturais do que os realizados pelo manequim. Isso se deve ao fato de que o manequim é controlado manualmente, o que causa algumas oscilações na rotação.

### 5.3.3 Velocidade angular máxima

O terceiro teste foi realizado para obter a velocidade angular máxima capturada pelo protótipo, obtida pela fórmula descrita na eq. 12, onde $D_q$ é o ângulo rotacionado e $D_t$ o intervalo de tempo. O ângulo rotacionado foi 90° e o intervalo de tempo foi obtido a partir da taxa de fps e do número de *frames* capturados.

$$w = \frac{D_q}{D_t} \qquad \text{(eq. 12)}$$

Foram realizadas cinco capturas, e as velocidades angulares obtidas são descritas na Tabela 6.

Tabela 6 – Velocidades angulares obtidas

| **Captura** | ***Frames* capturados** | **Taxa de fps** | **Tempo de captura** | **Velocidade angular** |
|---|---|---|---|---|
| 1 | 70 | 20,2663 | 3,4540 s | 26,0567 graus/s |
| 2 | 45 | 19,7281 | 2,2810 s | 39,4562 graus/s |
| 3 | 48 | 19,6978 | 1,9291 s | 46,6527 graus/s |
| 4 | 35 | 18,9804 | 1,8440 s | 48,8067 graus/s |
| 5 | 20 | 20,9863 | 0,9530 s | 94,4384 graus/s |

A Figura 85 mostra o gráfico com os movimentos capturados.

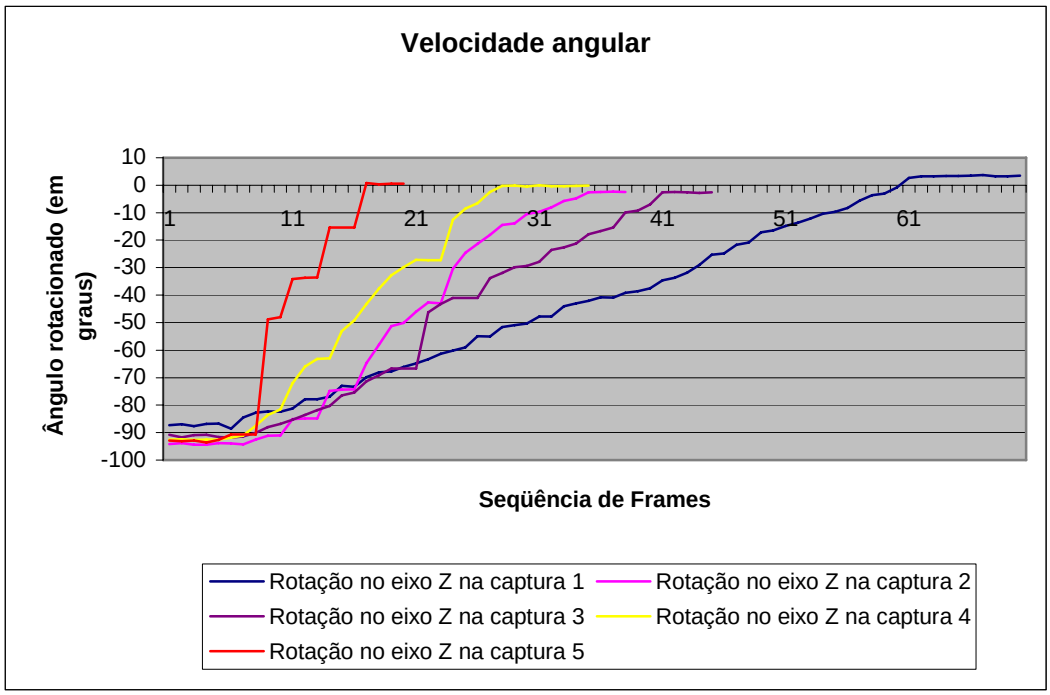


Figura 79 – Movimentos realizados no teste da velocidade angular

É possível verificar no gráfico da Figura 79 que quanto mais lento é o movimento realizado, mais *frames* são capturados. As perdas de *frames* são percebidas no gráfico nos intervalos onde há mudanças bruscas no ângulo rotacionado.

### 5.3.4 Reprodução do arquivo BVH gerado

Para verificar se o arquivo BVH gerado na captura estava de acordo com a especificação do formato, foi necessário reproduzir os movimentos capturados em um ou mais visualizadores de movimentos disponíveis. Foram escolhidos dois visualizadores: *WX Motion Viewer* (WXMV, 2006) e Bioviewer (BIOVIEWER, 2007). As Figuras 80 e 81 mostram, respectivamente, as telas do *WX Motion Viewer* e Bioviewer reproduzindo um movimento capturado.

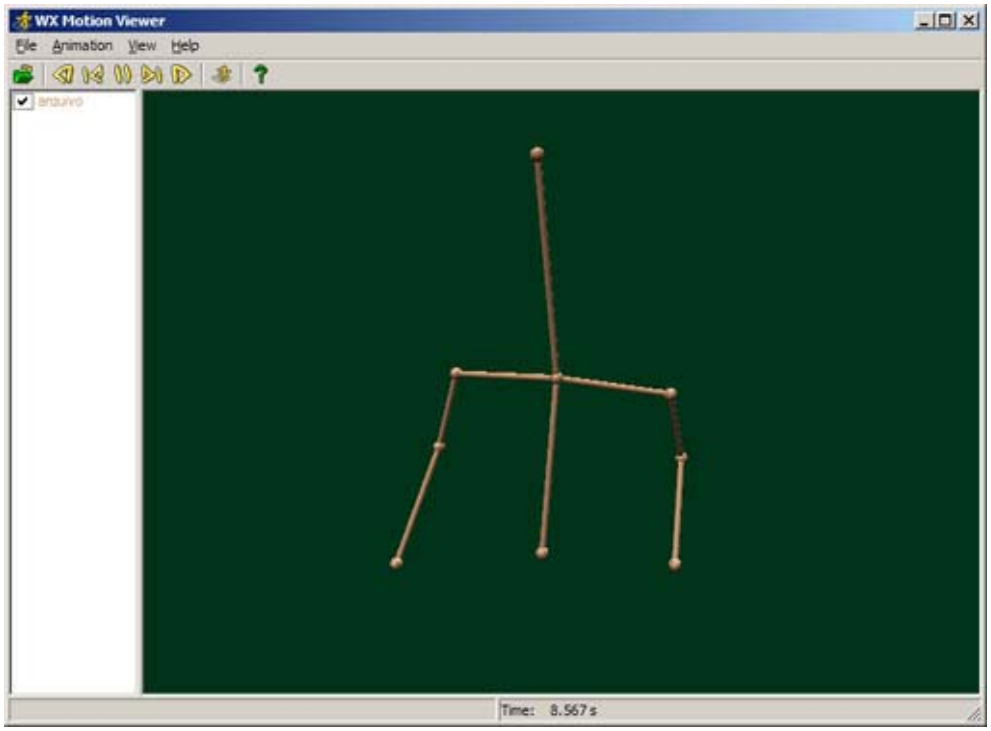


Figura 80 – Reprodução de um arquivo BVH no *WX Motion Viewer*

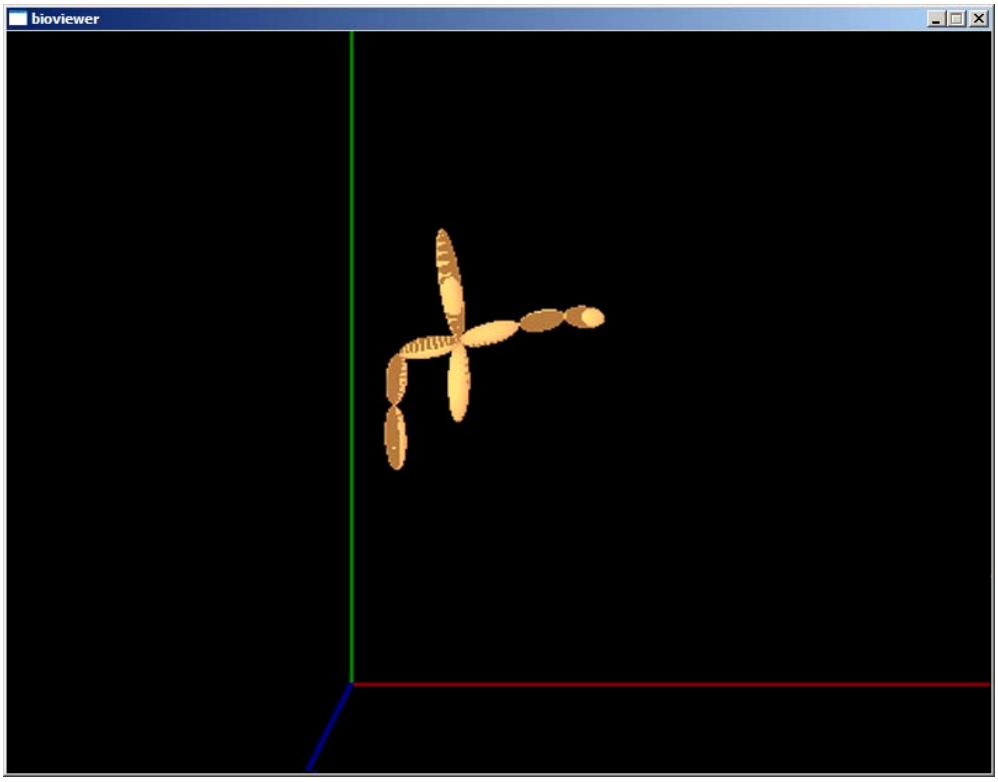


Figura 81 – Reprodução de um arquivo BVH no Bioviewer

Os movimentos capturados foram reproduzidos de acordo com o movimento realizado, comprovando a eficiência da biblioteca ARTookit para o desenvolvimento de rastreadores de movimentos corporais.

# CONCLUSÕES E TRABALHOS FUTUROS

A partir do trabalho apresentado, é possível concluir que a biblioteca ARToolkit é viável para o desenvolvimento de sistemas de captura de movimentos corporais baseados em esqueleto com estrutura hierárquica.

Os testes realizados mostraram que o protótipo de captura de movimentos reproduz os dados obtidos nas rotações das articulações, apresentando leves variações. A utilização de câmeras de vídeo mais adequadas e *hardware* com maior capacidade de processamento deve aumentar a precisão do movimento capturado.

Em relação à taxa de *frames* por segundo, ao contrário de outros trabalhos disponíveis na literatura, ela não decresceu conforme se aumentava o número de marcadores. Ocorreram variações conforme o número de marcadores, porém, sem apresentar um padrão de variação. Um dos motivos para isto pode ser o fato de que o hardware utilizado tem maior poder de processamento do que o utilizado em trabalhos anteriores.

Os movimentos capturados foram reproduzidos em dois visualizadores de arquivo BVH disponíveis, mostrando que os arquivos BVH gerados estavam de acordo com a sua especificação, e o movimento capturado de acordo com o realizado.

Destaca-se como trabalho futuro a utilização de mais de uma câmera de vídeo para o rastreamento dos marcadores, o que diminuiria a oclusão. Dias (2005) apresentou um suporte para rastreamento de marcadores da biblioteca ARToolkit utilizando múltiplas câmeras que poderia ser utilizado como base para este trabalho, já que foi desenvolvido em módulos reutilizáveis.

Visando também diminuir a oclusão, destaca-se a utilização de mais de um marcador para representar determinada articulação do ator. Assim como a utilização de mais de uma

câmera para rastrear o movimento, a utilização de dois marcadores ou mais por articulação ajudaria a reduzir a oclusão. Entretanto, o aumento do número de marcadores tende a aumentar o custo computacional do rastreamento.

Por fim, destaca-se também como trabalho futuro a implementação de um módulo para verificação da precisão do movimento capturado a fim de evitar erros de captura ou calibração.

Os três trabalhos propostos poderiam ser implementados com a biblioteca OSGART (OSGART, 2006), que fornece funções similares às do ARToolkit, porém, com funções de grafo de cena.

# REFERÊNCIAS


ARTOOLKIT. ***Augmented Reality Toolkit Library.*** Disponível em: <http://artoolkit.sourceforge.net/>. Acesso em: 01 Fev 2006.

ASCENSION. ***Ascension Technology Corporatio.*** Disponível em: <http://www.ascension-tech.com/>. Acesso em: 01 Fev 2007.

AUTODESK. ***AutoDesk.*** Disponível em: <http://www.autodesk.com/>. Acesso em: 01 Fev 2006.

AZUMA, R. T. ***A Survey of Augmented Reality***. Teleoperators and Virtual Enviroment v. 355-385. Ago. 1997. Disponível em: <http://www.cs.unc.edu/~azuma/ARpresence.pdf>. Acesso em: 01 Fev. 2006.

AZUMA, R. T. et al. ***Recents Advances in Augmented Reality***. IEICE Computer Graphics and Applications p. 34-47 Nov./Dec. 2001. Disponível em: <http://www.cs.unc.edu/~azuma/cga2001.pdf >. Acesso em: 01 Fev 2006.

BIOVIEWER. ***Biovision BVH Viewer***. Disponível em: <http://bioviewer.sourceforge.net/> Acessado em: 01 Set 2007.

CARDOSO, A.; MACHADO, L dos S. ***Realidade Virtual:*** *Conceitos e Tendências*. São Paulo: J. Garcia Comunicação Visual, 2004. cap 2 p. 21-32.

CREDO, ***Credo Interactive***. Disponível em: <http://www.charactermotion.com> Acesso em: 10 Abr 2006.

DESKSHARE, ***Using the Green Screen Technique***. Disponível em: <http://www.deskshare.com/Resources/articles/vem_greenscreentechnique.aspx> Acesso em: 01 Fev 2007.

DIAS, J. B. D. J. ***Suporte para Captura de Movimentos Baseado em Marcadores Passivos Utilizando Múltiplas Câmeras.*** 2005. f. 113. Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2005.

DORFMÜLLER-ULHAAS, K.; SCHMALSTIEG, D. ***Finger tracking for interaction in augmented environments***. Proceedings of the 2nd IEEE and ACM ISAR'01, International Symposium on Augmented Reality. 2001. Disponível em: < http://citeseer.ist.psu.edu/436795.html>. Acesso em: 10 Set 2007.

ELOP, ***Elbit Systems Electro-Optics ELOP Ltd.*** Disponível em: <http://www.el-op.com/category/head-up_displays.html> Acesso em: 10 Set 2007.

GARBIN, T.R. et al. *Sistema de realidade aumentada para educação de portadores de necessidades especiais*. KIRNER, C.; TORI, R. ***Realidade Virtual:*** *Conceitos e Tendências*. São Paulo: J. Garcia Comunicação Visual, 2004. cap 20, p. 277-282.

GLUI. ***GLUI User Interface Library***. Disponível em: <http://glui.sourceforge.net/> Acesso em: 28 Set 2007.

GLUT. ***GLUT – The OpenGL Utility Toolkit***. Disponível em: <http://www.opengl.org/resources/libraries/glut/> Acesos em: 28 Set 2007.

GREENLEAF, W. J. *Neuro/Orthopedic Rehabilitation and Disability Solutions Using Virtual Reality Technology*. AKAY, M.; MARSH, A***. Information Technologies in Medicine, Volume II***: *Rehabilitation and Treatment.* John Wiley & Sons, Inc. 2001. cap 1, p. 3-18.

H-ANIM. ***Humanoid Animation Working Group***. Disponível em: <http://www.h-anim.org/>. Acesso em: 01 Fez 2006.

KALAWSKY. The Science of Virtual Reality and Virtual Environment. Addison-Wesley. Workingham, England, 1993.

KARMA. ***Knowledge-based Augmented Reality for Maintenance Assistance*** Disponível em: <http://www1.cs.columbia.edu/graphics/projects/karma/karma.html> Data de Acesso: 01 Fev 2006.

KATO, H.; BILLINGHURST, M. ***Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System***. Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality. 1999. p. 85.

KIRNER, C.; TORI, R. ***Realidade Virtual:*** *Conceitos e Tendências*. São Paulo: J. Garcia Comunicação Visual, 2004. cap 1 p. 3-20.

LANDER, J. *Working with motion capture file formats.* ***Games Developer Magazine***, Manhasset, New York, p.30-37, Jan 1998. Disponivel em: <www.darwin3d.com/gamedev/articles/col0198.pdf > Acesso em: 11 Abr 2006.

LEDERMANN, F., et al. ***Dynamically Shared Optical Tracking.*** Proceedings of the IEEE First International Workshop on ARToolKit, 2002. Disponível em: <http://www.ims.tuwien.ac.at/media/documents/publications/SharedTrackingFinal.pdf> Acesso em: 01 Fev 2006.

LIAROKAPISA, F.; WHITEB, M. ***Augmented Reality Techniques for Museum Environments***. The Mediterranean Journal of Computers and Networks, Vol. 1, No. 2, p. 90-96. 2005. Disponível em: < www.soi.city.ac.uk/~fotisl/publications/MEDJCN2005.pdf > Acesso em: 01 Fev 2006.

LITEYE. ***Liteye Systems, Inc. - HMD designer & manufacturer***. Disponível em: <http://www.liteye.com/>Acesso em: 01 Fev 2007.

LOPES, L. F. B. ***O Estudo e a Implementação de Interfaces para a Utilização em Sistemas de Realidade Aumentada.*** 2005. f. 105. Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2005.

LOURENÇO, L. E. ***Um Sistema de Rastreamento Ótico Baseado em Marcadores Passivos Utilizando uma Câmera.*** 2004. f. 115. Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2004.

MACKAY, W. E. ***Augmented Reality:*** *Linking Real and Virtual Worlds*. Proceedings of ACM AVI '98, Conference on Advanced Visual Interfaces p. 9. L'Aquila, Italy, 1998. Disponível em: <http:// citeseer.ist.psu.edu/mackay98augmented.html> Acessado em: 01 Fev 2006.

MBCHB. ***MBChB Portfolio.*** Disponível em: <http://www.portfolio.mvm.ed.ac.uk/studentwebs/session1/group71//Stacsurgery.htm/>. Acesso em: 01 Fev 2007

MEREDITH, M.; MADDOCK, S. ***Motion capture file formats explained***. Department of Computer Science Technical Report CS-01-11, 2001. Disponível em: <www.dcs.shef.ac.uk/intranet/research/resmes/CS0111.pdf > Acesso em: 11 Abr 2006.

MICROVISION. ***Microvision: A World of Display and Imaging Opportunities.*** Disponível em: <http://www.microvision.com/>. Acesso em: 01 Fev 2007.

MILGRAM, P.; KISHINO, F. ***A Taxonomy of Mixed Reality visual displays***. IEICE Transactions on Information and Systems v. E77-D. Dec. 1994. Disponível em: <http://web.cs.wpi.edu/~gogo/hive/papers/Milgram_IEICE_1994.pdf >. Acesso em: 01 Fev 2006.

MILGRAM, P., et al. ***Augmented Reality:*** *A Class of Displays on the Reality-Virutality Continuum*. SPIE Proceedings Telemanipulator and Telepresence Technologies. p. 282-292. 1994. Disponível em: <http://vered.rose.utoronto.ca/people/paul_dir/SPIE94/SPIE94.full.html> Acesso em: 01 Fev 2006.

MILGRAM, P., et al. ***A Telerobotic Control Using Augmented Reality***. Proceedings 4th IEEE International Workshop on Robot and Human Communication (RO-MAN'95), Tokyo 1995. Disponível em: <http://etclab.mie.utoronto.ca/people/paul_dir/RO-MAN95/roman95.html> Acesso em: 01 Set 2007.

MOGILEV, D., et al. ***AR Pad:*** *An interface for face-to-face AR collaboration*. CHI Conference on Human Factors in Computing Systems. Minneapolis, Minnesota, Abr. 2002. Disponível em: <http://citeseer.ist.psu.edu/mogilev02ar.html> Acessado em: 01 Fev 2006.

MULDER, A. ***Human movement tracking technology***. Relatório técnico 94-1, Simon Fraser University, Jul 1994. Disponível em: < http://www.xspasm.com/x/sfu/vmi/HMTT.pub.html > Acesso em: 01 Fev 2006.

OKINO. ***Okino Computer Graphics.*** Disponível em: <http://www.okino.com/conv/imp_bvh.htm>. Acesso em: 10 Abr 2006.

OPENGL. ***OpenGL – The Industry Standard for High Performance Graphics***. Disponível em: <http://www.opengl.org/> Acesso em: 28 Set 2007.

OSGART. ***ARToolkit for OpenSceneGraph.*** Disponível em: <http://www.artoolworks.com/community/osgart/index.html>. Acesso em: 31 Dez 2006.

PHASESPACE. ***Phase Space Full Body Motion Capture System.*** Disponível em: <http://www.phasespace.com/>. Acesso em: 01 Fev 2006.

PIEKARSKI, W., THOMAS, B. ***ARQuake:*** *The Outdoor Augmented Reality Gaming System*. ACM Communications, v. 45, p. 36-38,. Jan 2002.

SEMENTILLE, A. C., et al. ***A Motion Capture System Using Passive Markers***. Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry. p. 440-447. Singapore, 2004.

SILVA, F. W. S. V. da. ***Motion capture:*** *introdução à tecnologia*. Rio de Janeiro, 2002. Disponível em: < http://w3.impa.br/~nando/publ/lcg-02.pdf> Acesso em: 01 Fev 2006.

STUDIERSTUBE. ***Studierstube Augmented Reality Project***. Disponível em: <http://www.studierstube.org/>. Acesso em: 01 Fev 2006.

TERASHIMA, N. *The Definition of Hiperreality*. TIFFIN, J.; TERASHIMA, N. ***Hyperreality****: Paradigm for the third millennium*. New York, NY. Routledge, 2001, p. 4-24.

TRIOVISIO. ***Triovisio***. Disponível em: <http://www.trivisio.com/>Acesso em: 01 Fev 2007.

UMLAUF, E. J., et al. ***ARLib:*** *The Augmented Library*. Proceedings of the First IEEE International Workshop on ARToolKit. 2002. Disponível em: < http://www.ims.tuwien.ac.at/media/documents/publications/ARLibFinal.pdf> Acesso em: 01 Fev 2006.

UVADRIVE. ***UvA SCS - Scientific Visualization and Virtual Reality***. Disponível em: <http://www.science.uva.nl/> Acesso em: 01 Fev 2007.

UWa. ***Madison Computer Sciences Department Home Pag***e. Disponível em: <http://www.cs.wisc.edu/graphics/Courses/cs-838-1999/Jeff/Example1.bva> Acesso em: 01 Fev 2007.

UWb. ***Madison Computer Sciences Department Home Pag***e. Disponível em: <http://www.cs.wisc.edu/graphics/Courses/cs-838-1999/Jeff/Example1.bvh> Acesso em: 01 Fev 2007.

VIIRRE, E., et al. ***The Virtual Retinal Display:*** *A New Technology for Virtual Reality and Augmented Vision in Medicine*. Proceedings of Medicine Meets Virtual Reality, San Diego, California, USA, 1998. p. 252-257. Disponível em: <www.hitl.washington.edu/publications/r-98-21/r-98-21.pdf > Acessado em: 01 Fev 2006.

VALLINO, J. ***Introduction to Augmented Reality***. Ago. 2002. Disponível em: <http://www.se.rit.edu/~jrv/research/ar/introduction.html>. Aceso em: 01 Fev 2006.

WATERWORTH, J. A. ***Virtual Reality in Medicine:*** *A Survey of the State of the Art.* Umeå University, Department of Informatics Research Report #9801, Out. 1998. Disponível em: < http://www.informatik.umu.se/~jwworth/execsum.html.>. Acesso em: 01 Fev 2006.

WATSON, B. et al. ***Evaluation of the Effects of Frame Time Variation on VR Task Performance***. VRAIS'97, IEEE Virtual Reality Annual Symposium, 1997. p. 38-44. Disponível em: <www.cs.northwestern.edu/~watsonb/docs/vr97.pdf >. Acesso em: 01 Fev 2006.

WELCH G., FOXLIN E. ***Motion tracking:*** *No silver bullet, but a respectable arsenal*. IEEE Computer Graphics and Applications, Nov/Dec 2002. v. 22. p. 24-38.Disponível em: < http://www.cs.unc.edu/~tracker/media/pdf/cga02_welch_tracking.pdf > Acesso em: 01 Fev 2006.

WOODS, E. et al. ***MagicMouse:*** *an Inexpensive 6-Degree-of-Freedom Mouse.* Proceedings of Graphite 2003, Melbourne, 11-13 Fev, 2003. Disponível em: <http://www.hitl.washington.edu/artoolkit/Papers/2003-Graphite-MagicMouse-Inexpensive6DOF.pdf > Acesso em: 01 Fev 2006.

WXMV. ***WX Motion Viewer***. Disponível em: <http://cgg.ms.mff.cuni.cz/~semancik/research/wxmv/index.html>. Acesso em: 30 Dez 2006

X-IST. ***X-IST Full Body Tracker.*** Disponível em: <http://www.x-ist.de/>. Acesso: 01 Fev 2006.