INTERFACE NATURAL EM 3D PARA ANIMAÇÕES INTERATIVAS

Andréa Lins e Lins Souza

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientador: Ricardo Guerra Marroquim

Rio de Janeiro
Setembro de 2017

INTERFACE NATURAL EM 3D PARA ANIMAÇÕES INTERATIVAS

Andréa Lins e Lins Souza

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Ricardo Guerra Marroquim, D.Sc.

Prof. Claudio Esperança, Ph.D.

Prof. Paulo Roma Cavalcanti, D.Sc.

Prof. Luiz Carlos Pacheco Rodrigues Velho, Ph.D.

Prof. Esteban Walter Gonzalez Clua, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2017

Souza, Andréa Lins e Lins

Interface Natural em 3D para Animações Interativas/Andréa Lins e Lins Souza. – Rio de Janeiro: UFRJ/COPPE, 2017.

XVI, 87 p.: il.; 29, 7cm.

Orientador: Ricardo Guerra Marroquim

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2017.

Referências Bibliográficas: p. 81 – 87.

1. Interface natural. 2. Animação 3D. 3. Manipulação baseada em gestos. I. Marroquim, Ricardo Guerra. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Agradecimentos

Depois de uma jornada tão intensa é chegada a hora de agradecer a todos aqueles que contribuiram, seja de forma direta ou indireta, para que esse trabalho se concretizasse.

Uma enorme gratidão tenho pelo meu orientador, Ricardo Marroquim, profissional exemplar e pessoa tão generosa, daquelas que duvidamos que ainda existam assim. Obrigada por ter aceitado ser meu orientador e por sua orientação, sempre disposto a ajudar, ensinar, ouvir, entender.

Agradeço aos professores do LCG, em especial, aqueles dos quais fui aluna em alguma disciplina. Ao professor Claudio Esperança, pelo exemplo e ensinamentos, pela paciência e disponibilidade para ajudar e por todas as dicas preciosas e ao professor Antonio Oliveira (*in memoriam*) pelos grandes ensinamentos.

Agradeço ao professor Luis Velho (IMPA), os seus conselhos e ensinamentos durante o tempo que fui "aluna-visitante" no VISGRAF também ajudaram a direcionar este trabalho. E também, ao Djalma, pelo suporte e dicas computacionais.

Agradeço aos meus colegas do LCG, pelos momentos de estudo e de descontração, em especial, aqueles com os quais trabalhei diretamente, Pedro e Bernardo.

Agradeço ao animador Vitor pelas construções das poses-chaves das animações.

Agradeço aos professores da banca, por terem aceitado o convite e pelas contribuições.

A minha querida amiga Maria Andrade (UFS), que mesmo distante sempre esteve presente me ajudando e me apoiando quando mais precisei.

Aos meus queridos pais, João e Marinalva, por terem me concebido e por me mostrarem desde sempre que o caminho do bem é a melhor escolha a seguir. Mesmo distante, sinto que estão presentes em todos os momentos.

Ao meu esposo Cláudio Lins, por sempre me dar forças e me apoiar durante toda a minha vida de estudos.

Aos meus avós, em especial a minha vó Laura, por todo o seu exemplo de força. Em nossas longas conversas sempre descobrimos o quanto somos semelhantes.

Agradeço ao meu querido sobrinho e afilhado Davi, por todos os momentos de descontração e de felicidade verdadeira que tivemos juntos. E a minha sobrinha Laura, por sempre demonstrar tamanha bondade em sua inocência de criança.

Aos meus queridos irmãos, Alex e Adelson e suas respectivas esposas, Carla e Edvania pelos momentos de descontração em família.

Ao meus parentes, afilhadas, amigos e conterrâneos que sempre torceram pelo meu sucesso.

Agradeço a UFRJ, a COPPE, ao PESC e ao LCG por terem me proporcionado momentos de intenso aprendizado e aos funcionários da secretaria do PESC, por todo o suporte acadêmico.

Agradeço ainda, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo suporte financeiro durante os meus estudos.

Agradeço a Deus, por ser sempre o meu refúgio e a minha fortaleza.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

INTERFACE NATURAL EM 3D PARA ANIMAÇÕES INTERATIVAS

Andréa Lins e Lins Souza

Setembro/2017

Orientador: Ricardo Guerra Marroquim

Programa: Engenharia de Sistemas e Computação

A tarefa de produzir animações de personagens virtuais articulados 3D pode ser árdua e maçante para o animador, quando ele não dispõe de técnicas e ferramentas para facilitar e agilizar o resultado do próprio trabalho. A escolha de métodos e de ferramentas apropriadas, tanto de hardwares quanto de softwares, muitas vezes é crucial para garantir o êxito.

Nesse contexto, apresentamos um sistema para produzir animações de personagens articulados 3D, através da performance do usuário. A nossa abordagem fundamenta-se na técnica de quadros-chaves espaciais, a qual associa um marcador no espaço para cada pose-chave de um movimento ou ação e permite que o usuário produza animações ao mover o controlador com um mouse 2D. Em nosso sistema, a interação com o ambiente virtual se dá através de ferramentas mais sofisticadas, capazes de capturar gestos e movimentos da mão em tempo real. Os dados capturados são utilizados pelo algoritmo para interpolar as poses-chaves e construir, automaticamente, novas poses da sequência de animação durante a performance do usuário.

Ademais, dispomos esses marcadores sobre superfícies com o intuito de mapear determinadas animações sobre as mesmas. Assim, é possível facilitar a localização espacial dos objetos presentes na cena, como os marcadores e o controlador, durante a performance. Embora estejamos trabalhando com movimentos e gestos, o nosso objetivo não é desenvolver uma interface puramente gestual. Com isso, ela também dispõe de controles acessíveis através do mouse, os quais conferem características adicionais ao sistema, como gravar a animação para visualização posterior, controlar a velocidade durante a visualização de uma animação gravada, dentre outras.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

NATURAL INTERFACE FOR 3D INTERACTIVE ANIMATIONS

Andréa Lins e Lins Souza

September/2017

Advisor: Ricardo Guerra Marroquim

Department: Systems Engineering and Computer Science

The task of producing 3D character animation may be arduous and tedious for the animator when he lacks the techniques and tools to facilitate and streamline the outcome of his work. Choosing appropriate methods and tools, both hardware and software, have been often crucial to success.

In this context, we present a system for performance-driven articulated 3D character animation. Our approach is based on the technique of spatial keyframing, that associates a marker in space for each key pose of a movement or action and allows the user to provide animations by moving the control cursor in the 3D space with a 2D mouse. In our system, interaction with the virtual environment takes place through more sophisticated tools capable of capturing gestures and hand movements in real time. The captured data is used by the algorithm to automatically construct new poses of the animation sequence during the user's performance.

In addition, we have these markers on surfaces with the intention of mapping certain animations on them. Thus, it is possible to facilitate the spatial localization of the objects present in the scene, such as the markers and the controller, during the performance. Although we are working with movements and gestures, our goal is not to develop a purely gestural interface. It also has controls that may be accessed by mouse and give additional features to the system, such as recording the animation for later viewing, controlling the speed when viewing a recorded animation, and more.

Sumário

Lista de Figuras

Lista de Símbolos

€	Euro, p. 31

Lista de Abreviaturas

2D	Bidimensional, p. 5
3D	Tridimensional, p. 1, 5
API	Interface de Programação de Aplicativos, p. 6
HMD	*Head-mounted display*, p. 29
ICP	*Iteractive Closest Point*, p. 27
LBS	*Linear Blend Skinning*, p. 12
NUI	Interface Natural de Usuário, p. 5
OpenGL	*Open Graphics Library*, p. 31
PLI	Problema Linear Inteiro, p. 22
RBF	Função de Base Radial, p. 4, 44, 60
SDK	*Kit* de Desenvolvimento de *Software*, p. 6

Capítulo 1

Introdução

Neste capítulo vamos discorrer brevemente sobre temas inerentes ao nosso trabalho, iniciando pela animação, incluindo animação por computador, bem como o processo de animação utilizando a técnica de quadros-chaves. Também apresentaremos, sucintamente, um histórico do surgimento das interfaces naturais, até chegar aos recentes dispositivos eletrônicos de captura 3D. Ademais, faremos uma abordagem a respeito das técnicas de interação 3D. E, por fim, apresentaremos o objetivo e a proposta do nosso trabalho.

1.1 Motivação

Desde os primórdios da comunicação humana, quando o homem tentava exprimir pensamentos e ações através de desenhos rupestres, percebe-se o desejo de tentar realizar a ação de animar. Atualmente, pensar em animação nos remete aos desenhos animados clássicos que assistimos pela televisão, aos jogos eletrônicos, ou às grandes produções de estúdios como *Disney, Pixar, Hollywood*, etc., geradas com o auxílio de toda a sofisticação de aparatos tecnológicos, tanto de softwares quanto de hardwares, disponíveis até então.

Mas o que exatamente é uma animação? A palavra animação vem do termo latim *animare*, que significa dar vida, ânimo, movimento [1]. De acordo com DENSLOW [2] há diversas definições para o termo animação e elas variam de acordo com a aplicação. Nesse sentido, podemos considerá-la como uma sequência de quadros dispostos lado-a-lado, de modo que objetos e/ou personagens possam locomover-se ao serem projetados.

Conforme WILLIAMS [3], há dois elementos básicos em uma animação: o tempo da ação e o espaço necessário para realizá-la. A sincronização desses dois elementos é de importância fundamental para obter um resultado agradável. Para que uma animação seja coerente, é indispensável que haja alguma alteração nos quadros da sequência, de acordo com o tempo. Sem isso, ela tornar-se-ia estática, o que

não faria sentido, já que o próprio significado da palavra animação nos remete à ideia de movimento. Em se tratando de animações realistas, também é importante sincronizar a velocidade da ação de acordo com o tempo. Animar um personagem andando ou correndo, por exemplo, geraria quadros diferentes ao longo do tempo; assim como em uma animação de corrida leve ou corrida mais intensa.

Existem diversas formas de se produzir uma animação, dentre as quais destacamos a animação digital 3D. Ela é produzida através de sistemas computacionais, os quais viabilizam movimentar corpos rígidos em uma cena, de acordo com movimentos específicos, usando recursos da computação gráfica. Através de transformações lineares, novos quadros são produzidos e renderizados, os quais adicionam novas cenas na animação. Pode-se, inclusive, controlar o movimento ou o ponto de vista de câmeras virtuais [4], recurso fundamental para a navegação em ambientes de realidade virtual.

Segundo THALMANN e THALMANN[5], o computador é capaz de assumir diversas funções na produção de uma sequência animada. Por exemplo, na construção dos quadros que compõem a sequência, o computador pode ser empregado para digitalizá-los e pintá-los, ou para construí-los totalmente através de editores de imagens. Ele também é capaz de gerar objetos complexos utilizando recursos de programação e pode, inclusive, ser adotado para criar o movimento, ao calcular quadros intermediários automaticamente, ou ainda, produzir movimentos complexos diretamente. Por fim, também é possível empregá-lo na fase de pós-produção para controle de edição e sincronização.

Com o advento e a disponibilização de novos aparatos tecnológicos e ferramentas para modelagem 3D, mapeamento de textura, iluminação, animação e renderização 3D, os filmes, jogos e desenhos animados vêm se modernizando a cada dia. E hoje, a tecnologia digital está cada vez mais presente nas animações, sejam elas 2D ou 3D, tornando possível produzí-las inteiramente através de imagens geradas por computador.

Um dos grandes desafios da animação digital é, justamente, criar animações de movimentos complexos, tão realistas quanto possível, utilizando as ferramentas computacionais disponíveis até então, como a computação gráfica, a visão computacional e, ainda, a interação homem-máquina. Contudo, deve-se observar que esse realismo não pode comprometer a qualidade e o desempenho da ferramenta utilizada, que, em muitas casos, deve processar as ações em tempo real.

As ferramentas de animação digital facilitam e agilizam o trabalho do animador, pois reduzem, consideravelmente, o tempo e o esforço necessário para produzir as animações, além de viabilizar resultados mais agradáveis visualmente. No entanto, mesmo que o animador disponha de diversos recursos computacionais, o trabalho ainda pode ser bastante fatigante e as ferramentas disponíveis, em muitos casos, são

bastante onerosas. Encontrar técnicas e ferramentas simples que tornem o trabalho mais simples e divertido é uma das barreiras a ser vencida durante a produção de uma animação.

Atualmente, existem diversas técnicas para produzir animações virtuais, dentre as quais destacam-se aquelas baseadas em captura de movimentos, simulação baseada em física e quadros-chaves.

Captura de movimentos: Um sistema de captura de movimentos coleta dados de movimentos, geralmente de atores reais, em um cenário controlado. Pode-se utilizar marcadores posicionados sobre o corpo do ator e, posteriormente, os dados são capturados por câmeras. Os resultados advindos da performance do ator são associados a avatares digitais. Com isso, é possível mapear o movimento do ator diretamente nos movimentos de personagens virtuais [6].

Simulação baseada em física: Com esse método, constrói-se os movimentos da animação e o controle do ambiente virtual como resultados de processos de simulação, através da aplicação de forças e torques baseados em modelos físicos reais [7, 8].

Quadros-chaves (*keyframes*): Essa técnica pode ocorrer tanto na forma tradicional, quanto computacional [9]. No método tradicional, um animador experiente desenha os quadros-chaves à mão e, posteriormente, animadores intervalistas completam a animação, preenchendo os quadros intermediários (*inbetweens*). Já na animação por computador prevalece a mesma premissa da tradicional, o que as diferem, no entanto, é que o animador especifica os quadros-chaves manual ou computacionalmente e os quadros intermediários são gerados pelo computador, mediante a interpolação de tais quadros-chaves. Em se tratando de animação de esqueletos virtuais 3D, os quadros-chaves são compostos por posições 3D de um modelo tridimensional [10].

A captura de movimentos, é uma técnica muito utilizada, principalmente pela indústria cinematográfica, pois ela produz movimentos realistas de alta qualidade. No entanto, para a fluidez do trabalho, é imprescindível que atores estejam disponíveis para executar os movimentos sempre que necessário, o que pode tornar-se inviável em determinados momentos e acabar por retardar a produção do animador. Além disso, faz-se necessário o uso de equipamentos sofisticados que, em sua grande maioria, são dispendiosos e de difícil acesso. O controle da iluminação também é essencial para garantir bons resultados e, quando utiliza-se câmeras 2D, deve-se dispor de pelo menos duas delas capturando a posição de um marcador ao mesmo tempo, para que se possa fazer a correspondência dessa posição em 3D. Outra des-

vantagem desse método é que ele não permite mapear diretamente os movimentos em personagens não bípedes, como cavalos, peixes, dentre outros.

Com a simulação baseada em física, o animador não necessita de dados adicionais dos movimentos, porém, ele não terá um controle preciso dos movimentos dos personagens virtuais durante a animação, pois tal controle é regido puramente por leis da física. Muitas vezes, essas leis são um tanto quanto complexas e requerem a execução de diversos cálculos, tornando esse método ineficaz e pouco utilizado para animar personagens articulados 3D. Ademais, esse método não é confiável para reproduzir movimentos humanos realistas.

Uma das vantagens do método de quadros-chaves tradicional é prover a liberdade de criação e o controle total do resultado final da animação. No entanto, retratar todos os quadros torna o trabalho um tanto quanto maçante; além do que, o método não é adequado para produzir animações tridimensionais. Já com a técnica de quadros-chaves digitais gera-se resultados mais realistas do que os produzidos manualmente em 2D, além de reduzir o esforço do animador. Assim como a captura de movimentos, essa técnica é bastante aplicada para animar personagens virtuais articulados 3D.

Em nosso trabalho estamos interessados em animar um personagem virtual articulado 3D e utilizamos a técnica de animação através de quadros-chaves. Ele fundamenta-se no trabalho proposto por IGARASHI *et al.* [11], os quais desenvolveram um sistema para criar animações de personagens articulados 3D baseadas na performance do usuário. Nesse trabalho, eles apresentaram um novo conceito, o qual denominaram *Spatial Keyframing*, ou *Quadros-Chaves Espaciais*. Com esse sistema, é possível gerar animações através da interpolação de quadros-chaves espaciais, usando como interpoladora uma função de base radial (RBF). Para isso, o animador deve definir as poses-chaves digitais manualmente, com um mouse, e associá-las a uma posição no espaço tridimensional, por meio de marcadores. Assim, a animação poderá ser realizada através da performance do usuário, o qual utiliza um dispositivo de entrada, no caso um mouse, para interagir com o sistema. Esse, por sua vez, realiza a interpolação dos quadros-chaves e retorna novas poses da animação. Mais detalhes deste sistema serão apresentados no Capítulo 4.

Ainda que seja um dos dispositivos mais utilizados em se tratando de interação com o computador, o mouse não é o meio mais apropriado quando estamos lidando com o espaço tridimensional. Criar animações 3D baseadas na performance do usuário torna-se uma tarefa dramática quando os dispositivos de interação não correspondem ao espaço em questão e estão limitados ao espaço 2D, como o mouse e a tela. Utilizando alguns recursos computacionais, é possível mapear os movimentos 2D do mouse para o espaço 3D, mas essa percepção não é direta e natural para muitos usuários. Uma forma de diminuir os impactos causados pelo uso de dispositivos

inapropriados ao espaço tridimensional, seria utilizar dispositivos com mais graus de liberdade, que permitissem ao usuário interagir de forma natural com o sistema.

1.1.1 Interfaces Naturais

Obter grandes avanços na forma de interação do homem com ambientes virtuais é um desejo constante nos dias atuais. Cada vez mais, os movimentos naturais do ser humano são requisitados nessa forma de interação, a qual vem se modificando intensamente ao longo dos últimos anos. Os primeiros relatos de interação do mundo real com o virtual surgiram durante os anos 60 e foram descritos por Ivan Edward Sutherland[1].

Em 1963, SUTHERLAND apresentou um sistema chamado *Sketchpad* [12]. Tal sistema permitia que o usuário interagisse diretamente com a tela do computador através de desenhos, tais como quadrados, triângulos e outras formas geométricas. Para isso, ele utilizou como dispositivo de entrada uma caneta óptica. Com ela, podia-se desenhar na própria tela e, também, selecionar e manipular desenhos exibidos nessa mesma tela.

Já em 1965, ele idealizou um dispositivo conectado a um computador digital para visualização e interação com objetos virtuais [13]. O dispositivo permitiria a exibição de objetos de realidade virtual de uma maneira realista e funcionaria como um receptor sensorial de estímulos físicos: sons, movimentos dos olhos e posições do corpo.

Finalmente, em 1968, ele exibiu um capacete para visualização tridimensional através de imagens em perspectiva [14]. Tal capacete era equipado com dois pequenos monitores e com sensores de movimento. Os monitores projetavam imagens 2D diretamente no olho do usuário, enquanto os sensores de movimento captavam a posição e a orientação a ser adotada pelo sistema de visualização. Assim, o usuário podia enxergar um objeto virtual 3D com movimentos semelhantes aos de um objeto real.

Desde então, muitos avanços foram apresentados e a indústria de jogos virtuais vem acompanhando esse progresso. Recentemente, foram criados dispositivos de baixo custo que permitem uma imersão total ou parcial no mundo virtual, através de Interfaces Natural de Usuário (NUI). Eles utilizam somente os movimentos do próprio corpo e, também, podem estar associados a comandos de voz. As interfaces por toque, por voz e de gestos, em alguns casos diminuiram ou até mesmo abandonaram completamente o uso do mouse. Com isso, para que haja interação com o mundo virtual, não se faz necessário qualquer tipo de contato físico do usuário com

[1]Engenheiro eletricista e cientista da computação americano, nascido em 1938. Em 1988, ganhou o prêmio Turing por sua contribuição visionária e pioneira para a computação gráfica, começando com o Sketchpad, e continuação posterior. É considerado como o "pai da computação gráfica".

tais dispositivos.

1.1.2 Dispositivos de Captura 3D

Um exemplo bem conhecido de dispositivo de captura 3D é o *Kinect* da *Microsoft* ([15–17]). Lançado em 2010 com o intuito de revolucionar a forma de interação com os jogos virtuais, o *Kinect* foi criado originalmente como um acessório do console de videogame *Xbox 360*. Mas, com o lançamento do seu *Kit* de Desenvolvimento de *Software* (SDK), a sua aplicação não ficou restrita tão somente aos jogos. Ele possui uma interface 3D gestual e por comando de voz, além de outros recursos, tais como uma câmera RGB e um sensor infravermelho que fornece dados de profundidade em tempo real. Tais recursos permitem utilizá-lo em diversas áreas de pesquisas, como Computação Gráfica, Processamento de Imagem, Visão Computacional e Interação Homem-Máquina [18, 19].

Com a popularização do *Kinect*, outros dispositivos 3D com tecnologias semelhantes foram concebidos. A exemplo do *Xtion PRO LIVE* da Asus [20], o *Senz3D* da *Creative* [21], as câmeras 3D com a tecnologia *Realsense* da Intel [22] e o *Structure Sensor* da Occipital [23]. Este último, tendo sido o primeiro scanner 3D móvel acoplado diretamente em dispositivos móveis, como *tablets*.

Diferentemente de outros dispositivos, o *Leap Motion* [24] não foi produzido como acessório de videogames. Lançado no mercado em 2013, ele é um dispositivo portátil 3D que vem ganhando espaço em pesquisas recentes. O *Leap* emite um feixe de luzes infravermelhas a partir de três fontes para fazer o rastreamento 3D da posição de objetos em tempo real. Assim, quando o objeto está no campo de visão do sensor ele reflete a luz de volta e uma ou ambas as câmeras de infravermelho acabam por detectá-lo. Através das imagens capturadas, ele é capaz de identificar movimentos naturais das mãos, de cada um dos dedos e de ferramentas pontiagudas, como pincel e caneta, além de dispor de gestos pré-configurados. A sua Interface de Programação de Aplicativos (*API*) permite acessar diversas informações, como a posição de cada um dos ossos das mãos, das pontas dos dedos, etc., além de movimentos e gestos. Ele fornece dados a uma taxa de até 300 quadros por segundo, com precisão de apenas 0,01 milímetros na posição dos dedos das duas mãos. A sua robustez o levou a ser adotado na criação de diversos aplicativos, inclusive com o intuito de substituir o mouse e o teclado, embora esse não seja o uso recomendado por seus criadores. Uma das vantagens do *Leap Motion*, em relação aos demais dispositivos, é a sua precisão e robustez no rastreamento das mãos e dos dedos. Além disso, por ser portátil, pode facilmente ser acoplado a outros dispositivos, como os de realidade virtual, com um diferencial de exibir as mãos do usuário, tornando a experiência de imersão mais agradável [25].

Além dos citados anteriormente, há outros sensores que utilizam tecnologias vestíveis, como o *Myo* [26], em forma de bracelete e o *Ring* [27], que tem o formato de um anel. Esses dispositivos limitam-se à detecção de gestos e movimentos das mãos e dos dedos, através de sensores de movimentos dos músculos. Possuem gestos pré-configurados que podem ser associados a comandos para controle de *smartphones, tablets, Smart* TV, computadores e drones, dentre outros equipamentos eletrônicos.

1.1.3 Técnicas de interação 3D

Segundo Bowman *et al.* [28, 29], as técnicas de interação em um ambiente virtual 3D podem ser divididas em três categorias: 1) navegação; 2) seleção e manipulação e 3) controle do sistema.

1) A *navegação* permite que o usuário movimente-se pelo ambiente virtual. Por sua vez, essa técnica também pode ser classificada em três categorias: *Exploração*, que é a navegação sem uma meta explícita, ou seja, o usuário apenas investiga o ambiente. *Busca*, que permite movimentar-se para um local de destino específico. E *Manobras*, que utilizam movimentos de alta precisão com um curto alcance, usados para colocar o ponto de vista numa localização mais vantajosa para a realização de uma determinada tarefa.

2) Já a *manipulação* permite ao usuário modificar o ambiente virtual e manipular objetos dentro desse mesmo ambiente. Ela deve fornecer meios para realizar pelo menos uma das três tarefas básicas: seleção, posicionamento e rotação do objeto. Nesse sentido, um recurso bastante utilizado é fornecer uma mão virtual ao usuário. Com a presença dessa mão no ambiente virtual a interação torna-se mais natural; assim, ela facilita a realização das tarefas. Um dos problemas que pode-se encontrar ao utilizar esse recurso é que, muitas vezes, a área de manipulação está limitada ao alcance de visão do dispositivo de entrada utilizado.

3) Por fim, o *controle do sistema* é a tarefa de mudar o estado do sistema ou o modo de interação. A eficácia na execução de tarefas que seriam simples em ambientes uni ou bidimensionais tradicionais, pode ser bem menor quando se trata de ambientes virtuais 3D. As técnicas de controle do sistema em ambientes virtuais imersivos podem, ainda, ser classificadas em quatro grupos: *menus gráficos*, que são representações visuais de comandos; *comandos de voz*, menus acessados via voz; *interação gestual*, conjuntos de comandos acessado através de gestos e *ferramentas*, que são objetos virtuais com uma função implícita ou um modo.

1.2 Objetivo

Dar vida a um personagem virtual 3D pode ser uma tarefa árdua e tediosa para o animador, quando o mesmo não dispõe de técnicas e ferramentas adequadas para simplificá-la e torná-la divertida.

Utilizar equações de movimento em um sistema de animação baseado em física, pode ser uma solução para calcular automaticamente a locomoção, diminuindo assim, o trabalho entediante do animador. No entanto, esse deverá, em alguns casos, dispor de cálculos complexos para obter resultados consideráveis e, também, perderá o controle preciso dos movimentos e possivelmente uma componente e liberdade artística. Uma outra alternativa seria utilizar a captura de movimentos, visto que ela permite representar e controlar os movimentos humanos com altivez. No entanto, ela fica limitada às restrições dos movimentos das articulações própria dos humanos, o que torna inviável a liberdade de criação dos animadores. Além disso, deve-se levar em conta o custo e a disponibilidade de recursos humanos e equipamentos adequados, que muitas vezes são sofisticados e caros, bem como a demanda de tempo para modelar e animar.

Nesse contexto, a nossa proposta de trabalho é promover uma alternativa para produzir animações computacionais de personagens articulados 3D, através de quadros-chaves espaciais, baseadas na performance do usuário. Elaboramos uma interface compacta e simples de ser manipulada e controlada, composta tão somente de um desktop com uma tela para visualização, ainda que essa seja 2D, e um sensor de profundidade acessível para detecção de gestos e movimentos das mãos. Confortavelmente sentado, o usuário poderá interagir com o ambiente virtual em tempo real e produzir as animações através do seu desempenho.

As animações serão controladas e produzidas em tempo real pelo usuário. Esse, por sua vez, deverá, através da sua produção, realizar gestos naturais, cujos movimentos serão associados a modelos de superfícies regulares. Como vimos na Subseção 1.1.3, muitas vezes, a manipulação em um ambiente virtual 3D requer o uso de um recurso visual. Com isso, em nosso sistema as superfícies terão a funcionalidade de apoiar os marcadores que serão associados a cada um dos quadros-chaves no espaço 3D, com o intuito de facilitar a localização espacial dos mesmos, durante a performance do usuário.

Um exemplo gerado com esse sistema, que tem como dados de entrada do algoritmo, os quadros-chaves espaciais, a posição de um controlador e a superfície e retorna uma pose intermediária, pode ser visualizado na Figura 1.1. Ela contém quatro quadros-chaves espaciais, com poses e marcadores, à esquerda e, à direita, uma amostra do sistema em execução, exibindo a superfície adotada para apoiar os marcadores no espaço, o controlador e uma nova pose.

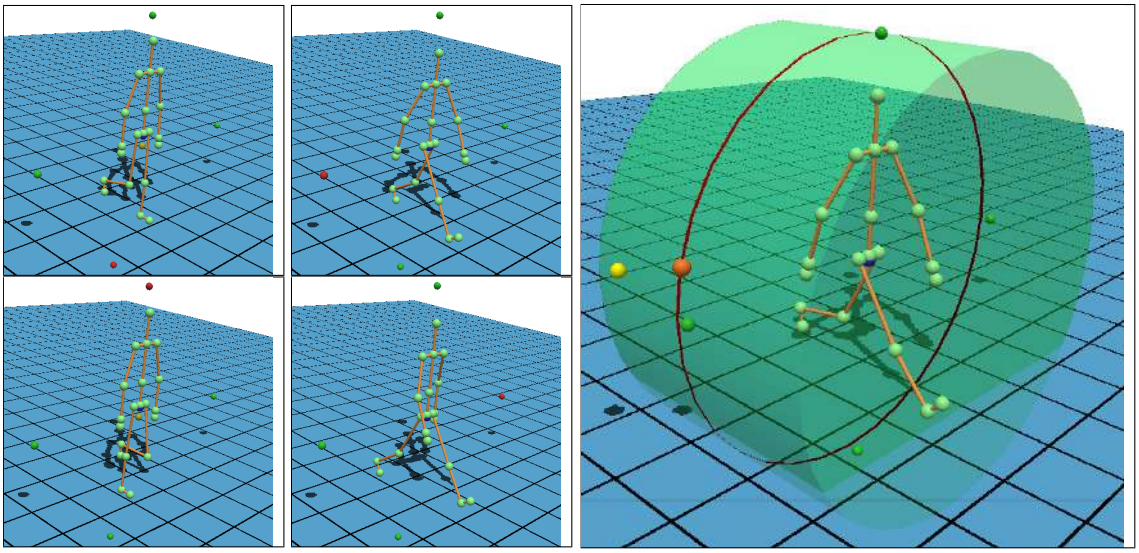


Figura 1.1: Panorama do sistema proposto. À esquerda temos quatro quadros-chaves espaciais. À direita temos uma nova pose gerada com o nosso sistema.

No nosso sistema, os gestos e movimentos serão capturados por sensores de movimento 3D, preferencialmente *handsfree*; visto que, utilizar um dispositivo de entrada 3D livre de contato físico com o usuário proporciona uma liberdade de movimentos e gestos ainda maior. Utilizar sensores 3D, ainda estabelece uma correspondência equivalente na dimensionalidade, já que, embora a visualização seja em uma tela 2D, todo o trabalho será desenvolvido no espaço tridimensional.

Alternativamente, também desejamos mapear ciclos de animações de estilos diferentes, em tempo real, sem que haja a necessidade de parar a animação para alterar os quadros-chaves pré-configurados.

Por fim, implementamos um sistema de animação digital, conforme exposto acima, para validar o método proposto e apresentar os resultados obtidos.

1.3 Organização da tese

Esta tese está organizada da seguinte forma:

Alguns trabalhos relacionados ao tema e que serviram de inspiração ao nosso trabalho são apresentados no capítulo 2. No capítulo 3 apresentamos um embasamento teórica útil ao nosso problema. Dedicamos o capítulo 4 para explicar, em maiores detalhes, o sistema de animação através de quadros-chaves espaciais. No capítulo 5 elucidamos a nossa proposta de trabalho, cujos resultados serão apresentados no capítulo 6. E, por fim, o capítulo 7, que é dedicado às conclusões e considerações finais.

Capítulo 2

Revisão Bibliográfica

Atualmente, as interfaces de animação baseadas na performance do usuário dispõem de uma nova ferramenta para que o utilizador possa interagir com o ambiente virtual, que são os dispositivos de captura de movimento 3D. Tais dispositivos podem ser utilizados para rastrear pessoas e objetos, associando-os a personagens e objetos na cena; para controle e comandos por gestos, dentre outras aplicações. Em alguns dos trabalhos que serão apresentados a seguir, os autores utilizam dispositivos de captura de profundidade para criar e manipular sistemas que permitem uma interação do usuário com o ambiente virtual, dentre os quais destacam-se o *Kinect* e o *Leap Motion*. Outros dispositivos de entrada como *webcam*, luva, etc., também permitem que o usuário possa interagir com os sistemas propostos pelos autores. No decorrer desse capítulo, faremos uma breve descrição desses trabalhos que serviram de inspiração ao longo da nossa pesquisa e foram fundamentais para a concepção e desenvolvimento do nosso método.

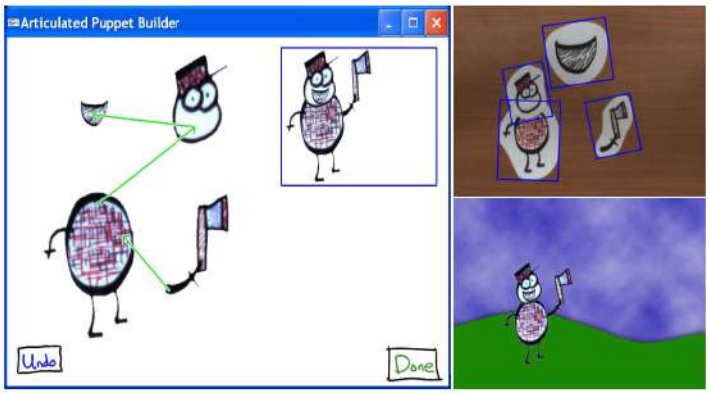
2.1 Animação de objetos articulados

Nessa seção iremos descrever alguns trabalhos que tem como proposta fundamental, animar objetos virtuais articulados. Em alguns casos, a interação do usuário com o sistema pode ocorrer através de gestos, ou de ferramentas físicas.

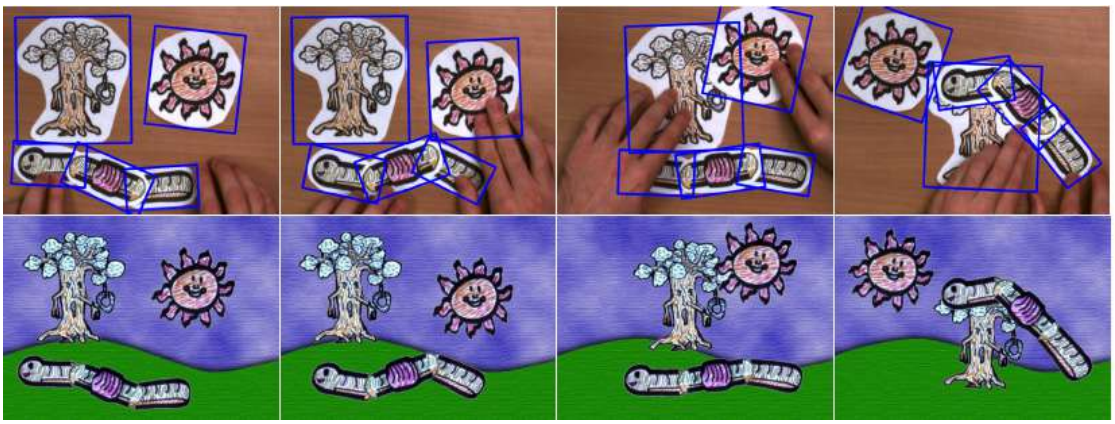
BARNES *et al.* [30] criaram um sistema baseado em vídeo 2D que permite ao usuário produzir animações com personagens de recortes de papel. Para tal fim, o sistema foi dividido em dois módulos: um construtor de fantoches, no qual cria-se um banco de dados de imagens dos personagens; e um teatro de fantoches, no qual os personagens podem ser renderizados, em tempo real, quando o sistema faz o rastreamento dos movimentos realizados pelo usuário e os associa com os personagens. Na Figura 2.1a podemos visualizar a configuração desse sistema e o mesmo sendo manipulado pelo usuário. O sistema também possibilita criar fantoches articulados, como pode ser visto na Figura 2.1b, o que torna o movimento mais

convincente. O usuário deve manipular os recortes de papel na frente da câmera 2D para gerar as animações, como ilustrado pelo Figura 2.1c. Esse sistema apresenta algumas limitações, por exemplo, ele não permite que o personagem seja rotacionado em três dimesões, com isso, os movimentos desses personagens ficam limitados por translação e escala.

(a) Configuração do sistema.



(b) Criação de fantoches articulados.

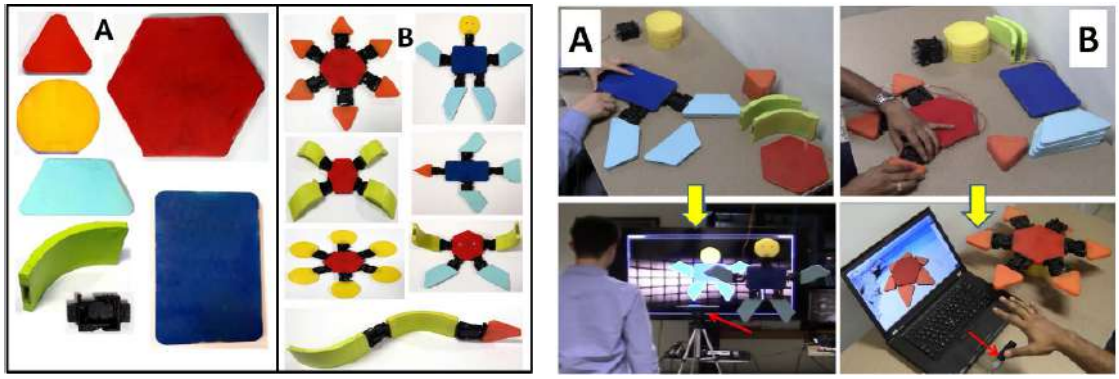


(c) Resultados da animação.

Figura 2.1: Sistema de marionetas baseado em vídeo 2D sendo utilizado por um usuário para gerar animações (a) e que permite criar fantoches articulados (b). Visão da camera (parte superior) e resultado da animação (parte inferior) (c). Figura extraída de [30].

O *framework PuppetX* foi desenvolvido por GUPTA *et al.* [31] para interações através de gestos. Ele permite que o usuário tenha uma certa liberdade para criar seu próprio modelo virtual, por intermédio de modelos reais. Tais modelos são compostos pela junção de diversas partes com formas geométricas básicas, a saber, círculo, triângulo, retângulo, hexágono e trapézio. Essas formas são ligadas fisicamente por um conector, formando assim, uma diversidade de fantoches, como pássaro, caranguejo, serpente, estrela do mar e flor. A Figura 2.2a mostra as partes utilizadas para gerar os modelos (lado esquerdo), bem como alguns modelos criados (lado direito). Para animar os fantoches virtuais, eles fazem a captura de movimentos do usuário por intermédio de dois sensores de profundidade: o *Kinect*, para capturar gestos do

corpo inteiro; e o *Leap Motion*, para capturar os gestos das mãos do usuário, já que ele mostra muito mais precisão nesse sentido. Na Figura 2.2b podemos observar as duas configurações do sistema: no lado direito, um cenário no qual o usuário utiliza os movimentos do corpo todo, que são captados pelo *Kinect*; e no lado esquerdo, um cenário criado com o *Leap Motion*, que faz o rastreamento dos gestos dos dedos do usuário. Algumas limitações são encontradas nesse sistema, como o número de fantoches gerados, que é apenas sete; as rotações nas articulações também são limitadas a apenas um dos eixos.

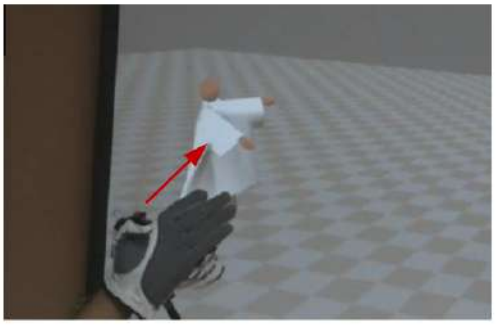


(a) (b)

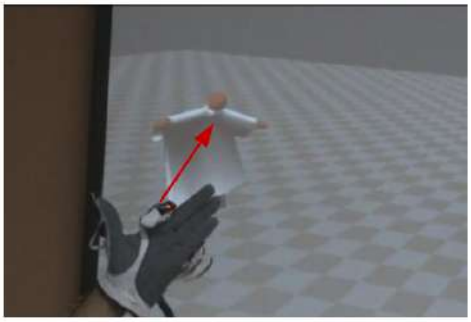
Figura 2.2: Componentes utilizados para gerar modelos físicos (a). Em (b), um modelo humanóide é utilizado para gerar animações com o *Kinect* (lado esquerdo) e um modelo de estrela-do-mar é utilizado para animações utilizando o *Leap Motion*. Figura extraída de [31].

O *IMPuppet*, sistema de animação interativa para gerar, planejar e controlar uma animação de fantoches de luva, foi projetado por LUO *et al.* [32, 33]. Nesse sistema, o movimento da mão do usuário é capturado em tempo real e reconhecido através do dispositivo de entrada *SmartGlove*. Diante desse cenário, um usuário pode controlar diretamente os dois braços e os movimentos da cabeça do boneco. A Figura 2.3 ilustra resultados da animação controlada pelo movimento de três dedos: o polegar, controlando os movimentos do braço direito (Figura 2.3a); o dedo indicador, controlando os movimentos da cabeça (Figura 2.3b); e o dedo médio, controlando o movimento do braço esquerdo (Figura 2.3a). Além disso, utiliza-se animação procedural para projetar alguns movimentos nos pés e no corpo do boneco virtual. A integração entre o movimento da mão e a animação procedural garantem um movimento fluido do boneco.

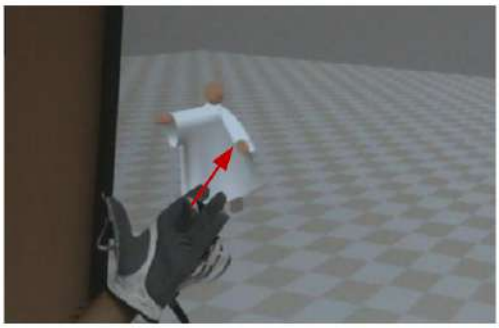
O método proposto por TSOLI *et al.* [34] apresenta uma nova abordagem para modelar deformações corporais, através de uma interface simples e intuitiva. Os autores se baseiam na técnica *Linear Blend Skinning* (LBS), que deforma formas 3D por intermédio de um esqueleto interno, no entanto, eles calculam o peso da LBS automaticamente. Uma das aplicações é deformar personagens virtuais 3D de forma



(a) Controlando a animação do braço direito através do movimento do polegar.



(b) Controlando a animação do braço esquerdo com o movimento do dedo médio.



(c) Controlando a animação do braço esquerdo através do movimento do dedo médio.

Figura 2.3: Usuário controlando o movimento do boneco através da *SmartGlove* utilizando três dedos, o polegar (a), o dedo indicador (b) e o dedo médio (c). Figura extraída de [32].

realista, usando malhas triângulares.

XIA *et al.* [35] apresentam um sistema de animação guiado por dados para síntese e controle de diversos estilos de movimentos humanos em tempo real. Através de uma base de dados de captura de movimentos heterogêneos com estilos não-rotulados, o algoritmo de aprendizagem on-line descreve as diferenças entre os estilos do movimento de entrada e o de saída por meio de uma mistura de modelos autor-regressivos locais. Esses modelos são construídos durante uma busca pelo exemplo de estilo de movimento mais próximo de cada uma das poses de entrada. As poses com estilos não-rotulados de movimentos diversos são, então, modificadas por uma transformação linear simples, ao ajustar os parâmetros previamente estimados pelos dados do treinamento; resultando, assim, em poses de saída com um estilo de movimento pré-definido. Com isso, é possível criar uma sequência de animação com determinadas ações humanas, tais como caminhada, corrida, socos, chutes, saltos e, ainda, transições entre essas ações, associando-as com diferentes estilos. O sistema foi testado com oito diferentes estilos, dentre eles neutro, sexy, deprimido, alegre, etc. É possível, ainda, misturar os tipos de estilos. O sistema ainda não é capaz de

reconhecer as ações e não pode ser utilizado com dados obtidos por dispositivos de captura de movimentos em tempo real. Uma de suas limitações é não produzir a animação do início, o que faz com que ele deva ser associado a sistemas de animação que utilizam técnicas de síntese de movimento.

2.1.1 Marionetes virtuais 3D

A proposta de SHIRATORI *et al.* [36] é criar uma interface para animação de marionetes 3D, na qual cria-se o movimento de um personagem virtual usando *blocking* e efeitos de simulação que seriam controlados por um animador. Fundamentados na ideia de que o animador pode realizar movimentos muito mais rápido quando manipula um objeto real, eles supõem que a interface funcionaria como uma alternativa aos *softwares* de animação comerciais existentes, que são baseados em quadros-chave. A interface foi desenvolvida através de um processo de refinamento interativo em conjunto com sete animadores profissionais, que realizaram diversos testes até se chegar a uma versão mais plausível do ambiente desenvolvido. A Figura 2.4 ilustra a ferramenta sendo utilizada pelo animador profissional para criar a animação para o diálogo "Salário Mínimo", cujos quadros resultantes podem ser vistos na Figura 2.5. Um sistema de captura de movimentos é utilizado para monitorar os movimentos do objeto, o qual foi criado com partes de brinquedos simples e escolhido após serem realizados diversos testes. O sistema permite controlar alguns graus de liberdade do personagem virtual durante a tomada, que necessariamente, não precisa ser única. Com isso, o animador tem a liberdade de redefinir o mapeamento dos movimentos entre as tomadas. A interface não funcionou adequadamente para movimentos que envolvem uma sincronização muito próxima entre os graus de liberdade do personagem ou ainda, quando diversas articulações precisam ser usadas ao mesmo tempo. Movimentos faciais também não funcionaram satisfatoriamente. O posicionamento de algumas partes do boneco em determinados gestos, como os que utilizam toques da mão, também ficou prejudicado.

Por sua vez, OSHITA *et al.* [37] propõem uma interface de controle de movimento de um personagem interativo usando manipulação com as mãos. A interface é inspirada com base no mecanismo de um boneco de marionete real (ver Figura 2.6) e pode ser aplicada em jogos de computador, comunicação via avatares, e criação de animação interativa, entre outros. Com essa interface é possivel realizar transformações de translação e rotação do personagem utilizando os movimentos dos dedos e das mãos. A Figura 2.7 ilustra o funcionamento dessa interface sendo utilizada pelo usuário. O sistema é dividido em três módulos: *processamento de dados de entrada*, que processa entradas a partir de um dispositivo que detecta o movimento das mãos; *interface de usuário*, que interpreta os dados de entrada processados e

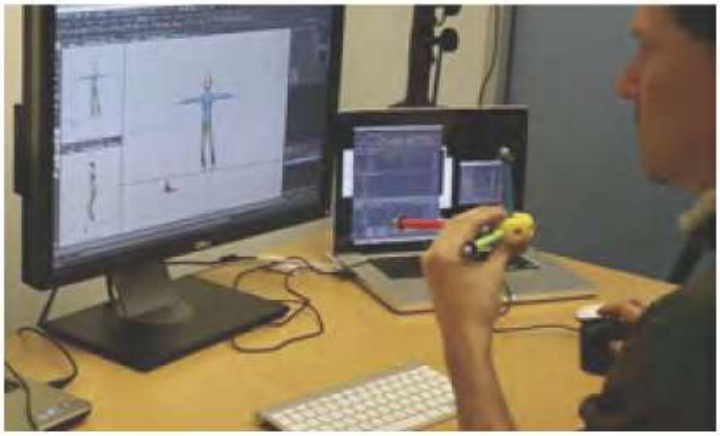


Figura 2.4: Animador profissional utilizando a interface para animar um personagem virtual. Figura retirada de [36].

Figura 2.5: Quadros resultantes para a animação do diálogo "Salário Mínimo". Figura extraída de [36].

determina os parâmetros de controle para o módulo de *controle de movimento*, que por sua vez, atualiza a pose do personagem com base nos parâmetros de controle invocados e do estado anterior desse personagem. Para detectar os movimentos das mãos, eles optaram por utilizar o dispositivo *Leap Motion*. No entanto, as mesmas restrições apresentadas na manipulação do boneco real, também verifica-se no ambiente virtual por causa das limitações desse dispositivo. Também, devido à falta de *feedback* apropriado quando ocorrem erros de rastreamento, o usuário não sabe se as duas mãos e todos os dedos foram reconhecidos durante o controle de movimento. Embora seja possível realizar vários passos curtos em ambientes pequenos, uma outra limitação da interface é que ela não permite que o boneco execute o movimento de caminhada em espaços maiores.

NINOMIYA *et al.* [38] optaram por utilizar uma simples *webcam* para desenvolver um sistema no qual o usuário pode manipular personagens de marionetes virtuais. Nesse sistema o usuário pode utilizar o movimento dos dedos para manipular os personagens. Esses movimentos são capturados em tempo real por diversos subsistemas, cada um utilizando uma câmera de vídeo digital conectada a um computador. Para controlar a marionete virtual, cada dedo é associado a uma determinada parte

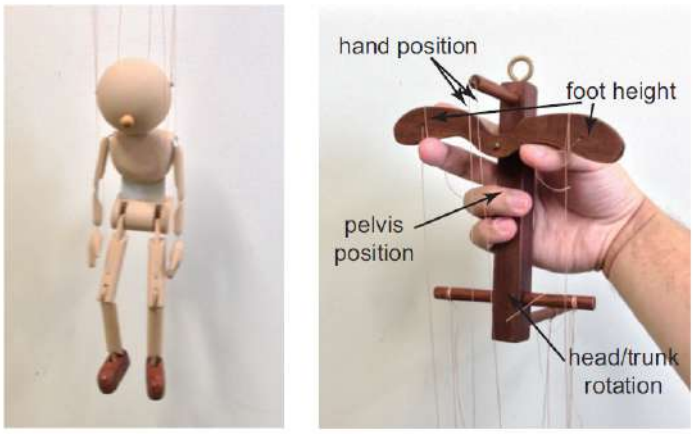


Figura 2.6: Mecanismo do boneco real que é suspenso por cerca de 10 cordas ligadas a diferentes partes do corpo e ao controlador. Figura extraída de [37].

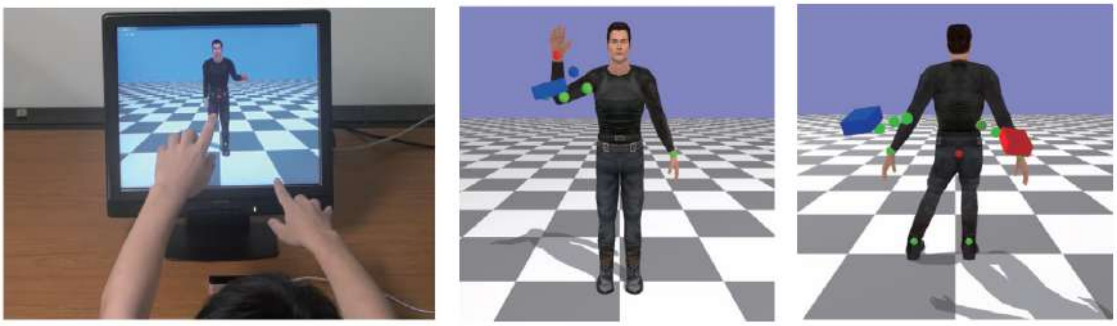


Figura 2.7: Interface implementada com o dispositivo *Leap Motion*. Os paralelepípedos representam as mãos do usuário (vermelho: a mão direita e azul: a mão esquerda). As esferas representam os dedos do utilizador (verde: dedo curvado, vermelho ou azul: dedo estendido). As esferas sobre o personagem são pontos controlados (vermelho: controlados pela mão direita ou os dedos, azul: controlados pela mão esquerda ou os dedos, verde: não controlados, mas o dedo correspondente é reconhecido). Figura retirada de [37].

da marionete, ligados por uma corda virtual, cujo comprimento é determinado pelo ângulo de flexão dos dedos. Assim, o usuário pode controlar o movimento e os gestos da marionete virtual quando movimenta os próprios dedos. No entanto, cada subsistema pode controlar apenas um personagem virtual na cena e o trabalho foi limitado a somente dois subsistemas. Eles criaram diversos tipos de personagens e de *backgrounds* para o sistema. Alguns resultados podem ser vistos na Figura 2.8, a qual exibe um mesmo personagem de marionete virtual em três cenários diferentes.

2.1.2 Marionetes de sombras

Um sistema de contação de histórias inspirado no tradicional teatro de sombras chinês, chamado de *ShadowStory*, foi projetado por LU *et al.* [39]. Esse sistema permite ao usuário criar os seus próprios personagens digitais articulados e utilizá-

Figura 2.8: Marionete virtual em cenários diferentes. Figura extraída de [38].

los para criar histórias personalizadas. Para esse fim, eles criaram uma interface interativa com operações simples na qual o usuário tem a possibilidade de criar os personagens, os acessórios e o cenário utilizando um *tablet* e uma caneta como dispositivo de entrada (ver Figura 2.9a). Para facilitar a interação nesse processo, as ferramentas virtuais são semelhantes àquelas utilizadas para criar os bonecos reais. Existe ainda, a possibilidade de selecionar os elementos da história diretamente em um banco de dados, o qual contém elementos digitalizados do teatro de sombras chinês, como mostra a Figura 2.9b. Concluída essa etapa, pode-se, ainda, exibir histórias ao vivo utilizando um projetor e uma tela de projeção. Diante desse cenário, os personagens ou acessórios podem ser controlados por movimentos corporais simples, pois cada um deles, independentemente, está associado a um par de sensores sem fio de movimento das mãos. Um dos sensores controla o deslocamento, que fica limitado a mover o personagem ou o acessório para a direita, para a esquerda, para cima ou para baixo. Já o outro sensor, faz com que o personagem curve o tronco para frente ou para trás, realizando movimentos para a esquerda ou para a direita. A combinação desses movimentos principais faz com que outros sejam gerados automaticamente, como os movimentos das articulações dos braços e das pernas do boneco. O sistema funciona de forma colaborativa, mas para criar uma história coerente, seriam necessários diversos usuários. Isso seria um problema na fase de criação dos elementos da história, pois o *tablet* suporta apenas um usuário da interface por vez.

(a) Interface. (b) Animação.

Figura 2.9: Interface para criar os objetos virtuais (a) e uma animação utilizando elementos do banco de dados (b). Figura extraída de [39].

ZHANG *et al.* [40] exibiram uma interface interativa para controlar marionetes do teatro de sombras chinês utilizando gestos corporais. Para manipular os bonecos, eles também utilizam os movimentos da mão do usuário, além dos movimentos do corpo todo. No entanto, eles empregam como dispositivo de captura de movimento apenas o *Kinect*. Os movimentos desse tipo de marionete estão limitados à duas dimensões. Além disso, o trabalho foi baseado em apenas dois modelos, um humano e um animal, como ilustra a Figura 2.10a. Cada um desses modelos foi associado aos movimentos de diferentes usuários. O modelo humano foi mapeado diretamente no esqueleto obtido com o *Kinect*, já o modelo animal necessitou de algumas adaptações. Movimentos simples como virar a cabeça e chutar, são executados e reconhecidos com facilidade. No entanto, movimentos mais complexos como uma cambalhota requer usuários experientes, com isso, eles optaram por utilizar algumas posturas especiais para representar tais movimentos. Eles também criaram um cenário 3D simulando um cenário real, que pode ser visto na Figura 2.10b.

(a) Modelos. (b) Cenário.

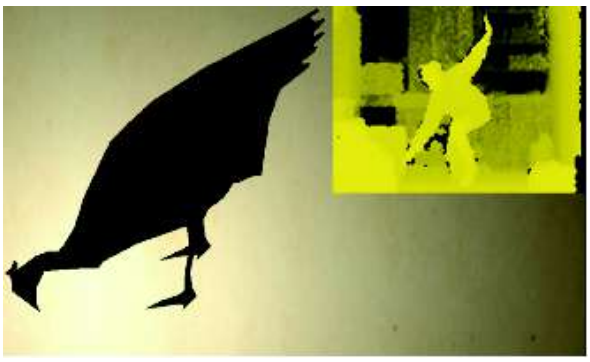
Figura 2.10: Marionetes do teatro de sombra chinês (a) e um cenário virtual desse mesmo teatro (b). Figura retirada de [40].

Os autores LEITE e ORVALHO [41] propõem em seu trabalho, um sistema

interativo que permite controlar os movimentos de um boneco de sombra utilizando os movimentos do próprio corpo do utilizador. O sistema funciona em tempo real e é baseado em sensores de profundidade. Os movimentos do corpo são capturados utilizando o *Kinect*, que retorna os dados associados ao esqueleto do usuário. De posse desses dados, pode-se mapeá-los no objeto virtual e gerar a animação em tempo real. Além de utilizar modelos cujo esqueleto são semelhantes ao humano, como o personagem 2D bípede ilustrado na Figura 2.11a; eles também utilizaram modelos que não o são, como mostram as Figuras 2.11b e 2.11c. Nesse último caso, foi necessário adaptar a correspondência entre os esqueletos do usuário e do boneco e isso também requer um esforço maior do usuário para compreender o sistema. Apenas as translações e rotações 2D são comportadas nesse sistema e, como ele não é automático, alguns recursos, como reconhecimento de gestos não são utilizados, o que acaba limitando os movimentos durante a animação.

(a) Humano.

(b) Dragão.



(c) Pássaro.

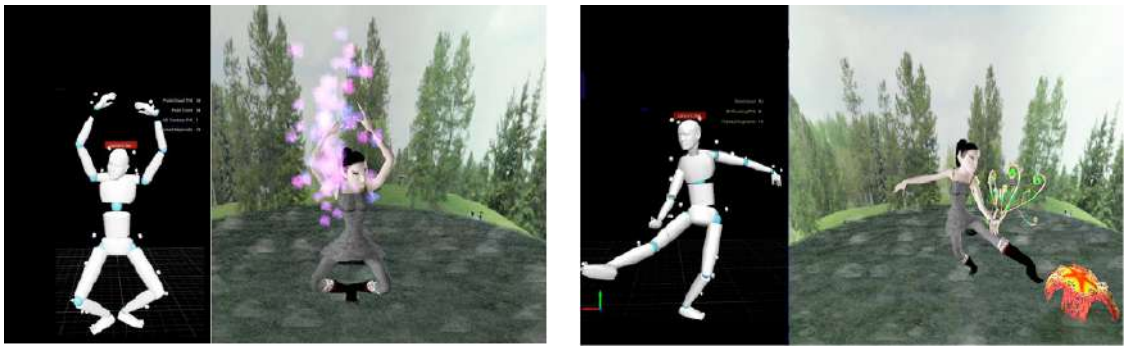
Figura 2.11: Bonecos de sombra controlados pelo usuário: Em (a) um modelo humano, em (b) e (c) modelos animais. Figura retirada de [41].

2.1.3 Teatro virtual interativo

WU *et al.* [42] propõem um sistema interativo para simular um ambiente de apresentação teatral. Nesse ambiente, atores reais podem controlar a animação de per-

sonagens virtuais em tempo real, com isso, preserva-se a interatividade do teatro tradicional. As imagens 3D são projetadas em três paredes de uma caverna digital, como exemplificado na Figura 2.12. O personagem virtual é animado quando se dá a captura de movimentos e reconhecimento de gestos dos atores reais, de forma natural e intuitiva. As informações do esqueleto, resultantes do rastreamento dos movimentos do ator, são mapeadas diretamente no esqueleto que representa o personagem virtual. Para isso, eles precisam estar em um mesmo sistema de coordenadas e devem ter medidas proporcionais. Para executar a animação do personagem virtual, são utilizadas 24 articulações do esqueleto e é possível realizar movimentos de rotação e translação do mesmo em 3D. O sistema é completamente automático e a sua configuração permite que um ator possa ouvir e ver, em tempo real, tanto outros atores quanto o público, através de microfones e câmeras, permitindo uma interação com todos aqueles que estão presentes na cena, seja assistindo ou atuando, ainda que a plateia não perceba a presença desse ator. Para que isso aconteça, também são instaladas câmeras dentro da caverna para que o ator possa acompanhar as reações do espectador e, assim, conduzir a cena de acordo com tais reações. Embora tenha citado a possibilidade de colaboração entre vários atores reais controlando diversos personagens, o sistema elaborado ficou restrito a apenas um usuário por vez. A Figura 2.13 ilustra um resultado desse sistema. Em cada uma das subfiguras (2.13a e 2.13b), à esquerda está o esqueleto resultante da captura de movimento do usuário e à direita, o ambiente virtual renderizado. Nessas subfiguras, pode-se verificar que o movimento real é um movimento espelhado quando aplicado no personagem virtual, o que pode se tornar um empecilho para alguns usuários.

Figura 2.12: Espectador assistindo a performance do ator em um ambiente de realidade virtual imersivo. Figura extraída de [42].



(a) (b)

Figura 2.13: Personagem em um ambiente virtual controlado por um ator real. Em cada uma das figuras, o lado direito ilustra os movimentos do ator real representado pelo esqueleto virtual e o lado direito estampa o espaço cênico em conjunto com o ator virtual. Em (a), o gesto com as mãos ativa os brilhos. Em (b) a elevação da mão e do pé esquerdos aciona o crescimento das plantas no lado esquerdo do cenário. Figura extraída de [42].

2.1.4 Esqueletos complexos

Um dos desafios de animar objetos esqueléticos é que, em alguns casos, eles tem estruturas complexas e podem apresentar dezenas ou até mesmo centenas de ossos e juntas. Diante de um número extenso de graus de liberdade de movimentos, representar e controlar cada um deles diretamente, ainda que com o auxílio de um *proxy* físico, pode tornar o trabalho do animador ainda mais entediante e demorado, inclusive durante as fases de produção e perfomance.

Pensando em solucionar problemas como esse, GLAUSER *et al.* [43] construíram um sistema de animação 3D que tem por objetivo animar personagens articulados complexos, utilizando um *proxy* físico. A interface natural integra software e hardware por intermédio de um controle modulado palpável, capaz de medir rotações 3D nas juntas, associado a algoritmos de *rigging*. Os dados de entrada do algoritmo são conjuntos de poses de um objeto articulado. O controle físico pode ser montado de forma que as suas juntas não estabeleçam uma correspondência biunívoca com as juntas do esqueleto virtual. Desse modo, apenas os graus de liberdade predominantes do esqueleto são controlados diretamente; os demais, são dominados por uma técnica de interpolação das poses, mediante uma abordagem que se assemelha à cinemática inversa, resultando, assim, em animações suaves. Para estabelecer a forma da estrutura física, primeiramente, constitui-se uma configuração dos módulos de modo que a mesma maximize o número de poses a ser construídas. As partes ligadas por juntas são unidas por dois discos e utilizam o conceito de junta universal aumentada; sendo assim, podem ser rotacionadas tridimensionalmente em qualquer direção. Para determinar as rotações são utilizados imãs e três sensores de efeito

Hall posicionados em cada junta, acompanhados por dois microprocessadores para capturar e enviar os dados adquiridos. Em seguida, as rotações detectadas devem ser mapeadas com os parâmetros que irão controlar o objeto virtual. Após a montagem, o controle é ligado ao computador, podendo, a partir daí, ter o seu esqueleto associado ao esqueleto virtual. Para tal, o algoritmo é dividido em etapas: na primeira, aloca-se todas as juntas disponíveis no dispositivo e na segunda, atribui-se divisores a nós com ramificação, com base na solução de um Problema Linear Inteiro (PLI) com algumas restrições. Em alguns casos, a solução pode não ser única, o que acaba por levá-los a escolher aquela com o menor erro rotacional médio dos divisores, o que faz com que a geometria da solução seja a mais próxima da geometria do esqueleto.

Sendo assim, é possível posicionar e animar diretamente objetos que possuem esqueletos complexos, ao manipular um simples controle articulado montável, com um número reduzido de articulações (ver Figura 2.14). O algoritmo também pode ser utilizado em animações de objetos não articulados, como por exemplo, para animar uma boca. O sistema promete simplificar o trabalho do animador e diminuir o tempo de preparo da pose de um quadro-chave, beneficiando a experiência do usuário.

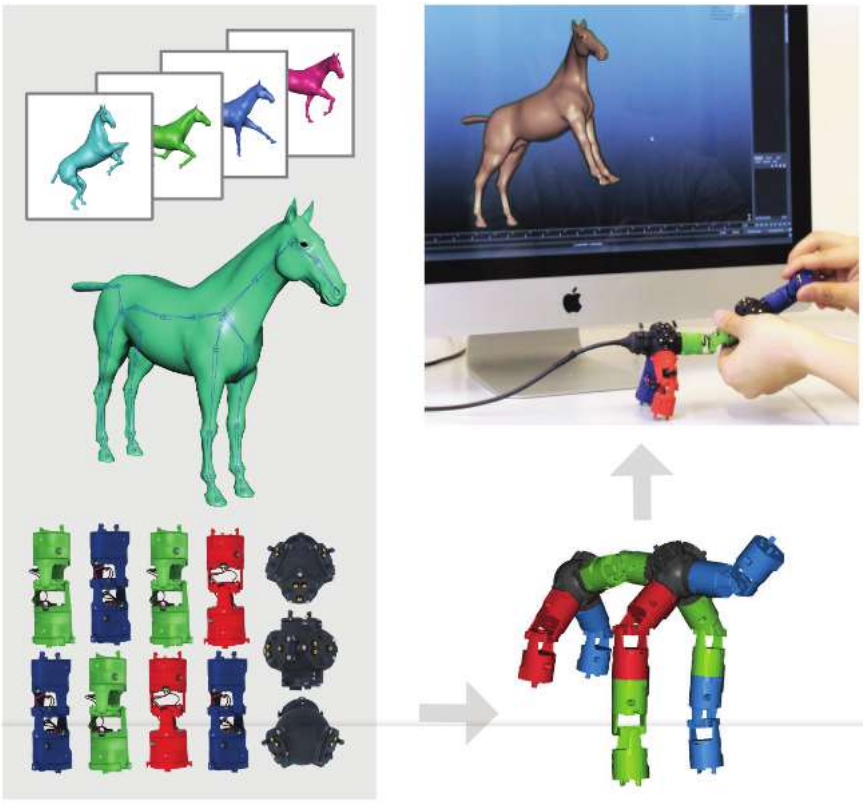


Figura 2.14: Sistema de animação 3D para personagens articulados complexos. A interface dispõe, como dispositivo de entrada, de um *proxy* físico articulado associado ao esqueleto virtual. Figura extraída de [43].

Embora tenha apresentado resultados satisfatórios frente a outros trabalhos se-

melhantes e com agilidade na composição das poses, muitas vezes a pose desejada não está contida no conjunto de poses iniciais e não pode ser representada através desse sistema. Outras vezes, há a predominância de um determinado movimento, quando o mesmo é prevalecente no conjunto de poses. Uma outra deficiência do sistema é que ele só reconhece as rotações locais, o que acarreta em um controle externo da posição e orientação global do objeto virtual.

2.1.5 *Extended Spatial Keyframing*

O método seminal de animação através de quadros-chaves espaciais (*Spatial Keyframing*) proposto por IGARASHI *et al.* serviu de inspiração para outros trabalhos. Um deles, elaborado pelos autores CHOI *et al.* [44] e denominado como *Extended Spatial Keyframing (ESK)*, apresenta uma extensão dessa técnica. Esses autores afirmam que o método proposto permite criar animações de alta qualidade com personagens complexos altamente articulados. O sistema admite que o personagem seja animado delicadamente pelo desempenho do usuário. Eles também utilizam animação em camadas para aumentar o nível de detalhe dos movimentos finais. Assim, como no método original, o mouse é o dispositivo adotado para fazer a comunicação do usuário com o sistema, o que acaba limitando os graus de liberdade ao controlar os personagens articulados.

2.1.6 Animações cíclicas

Os movimentos cíclicos são aqueles em que as ações se repetem, sempre retornando ao estágio inicial. Grande parte dos movimentos humanos são cíclicos, dentre os quais destacam-se os passos de uma caminhada ou de uma corrida, nadar, pedalar, remar, etc.. As animações cíclicas são tão úteis que existem motores de jogos com comandos apenas para integrar de forma suave diferentes ciclos de movimento.

Segundo WILLIAMS [3], na forma tradicional, uma passada em uma caminhada 'normal' realista, por exemplo, deveria ser construída utilizando ao menos cinco quadros-chaves, como ilustrados na Figura 2.15. Utilizando algumas estratégias, pode-se simplificar a quantidade de quadro-chaves necessários para gerar animações cíclicas de um mesmo movimento com estilos variados, ao aproveitar alguns quadros-chaves básicos. Ainda de acordo com WILLIAMS [3], uma forma mais simples de construir uma passada com diferentes estilos de movimento seria começar desenhando as duas posições básicas de contato, que estariam nos quadros-chaves extremos da animação e, a partir dessas posições, construir diversos ciclos de animação levando em consideração o modo de caminhar do personagem, por exemplo. A Figura 2.16 ilustra essa ideia. Nela é possível encontrar as mesmas poses extremas, tanto na imagem à esquerda, quanto na imagem à direita, todavia o que as di-

fere é exatamente a pose central, gerando com isso, diferentes estilos de movimento durante a passada.

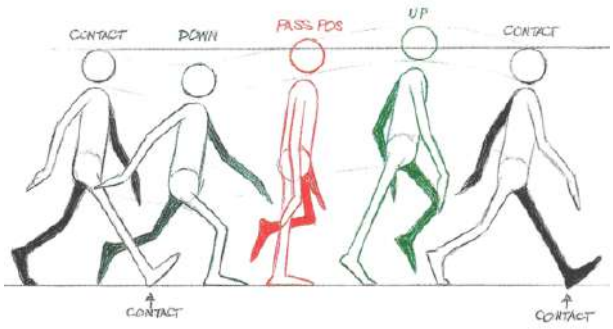


Figura 2.15: Poses-chaves em uma caminhada 'normal' realista. Figura extraída de [3].

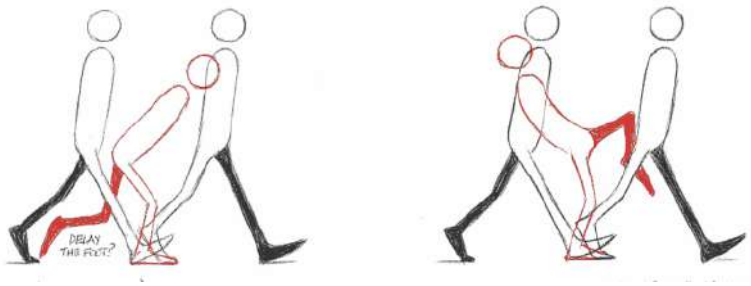


Figura 2.16: Estilos diferentes de caminhada utilizando as mesmas posições básicas de contato. Figura retirada de [3].

CICCONE *et al.* [45] apresentam um sistema simples de autoria para compor, representar e manipular animações cíclicas em tempo real. Por intermédio de uma interface natural, um ciclo de animação pode ser produzido sempre que o usuário executa loops de um determinado movimento. Tais loops produzem uma curva com trajetória aproximadamente periódica. Com o auxílio de um algoritmo de otimização para análise e extração automática de curvas, o sistema calcula a curva média do período e retorna um ciclo fechado do movimento suavizado. Os ciclos de movimento são editados com a ferramenta denominada *MoCurves*, mediante uma combinação de controles temporais e espaciais descritos por curvas de Bézier cúbicas. Além das transformações de translação, rotação, escala e tempo de cada ente animado, o usuário pode manipular, em uma única janela de exibição, curvas de ciclos de movimentos do personagem como um todo, ou de cada uma das suas partes independentemente, através de camadas, sincronizando-as utilizando uma função baseada no tempo. Para controlar a velocidade do movimento durante a animação, os autores desenharam pontos-chaves sobre a *Mocurve* que controla a translação. O usuário poderá editá-los diretamente, alterando as posições dos mesmos sobre a curva. Quanto mais próximos esses pontos estiverem um do outro, mais lento será

o movimento. Já quando eles estiverem mais afastados, o movimento será muito rápido. O contato com o solo, ou com outras regiões planas é determinado por *sketches*, onde um único traço assegura uma transformação espaço-temporal.

O sistema foi testado por animadores experientes e novatos, cujos desempenhos foram capturados por dispositivos distintos. Eles experimentaram desde os dispositivos mais usuais, como um mouse, um *tablet* e uma caneta digitalizadora, bem como alguns mais sofisticados tais quais o *Leap Motion*, os óculos de realidade virtual *HTC Vive* e um sistema de captura de movimentos do corpo inteiro, conforme ilustrado na Figura 2.17. Mas para que o sistema funcione com a eficiência prometida, os ciclos produzidos pelo usuário não devem ter variações significativas, tanto na forma, quanto no tempo, pois sendo assim o algoritmo pode não conseguir determinar um período ou extrair um loop satisfatório. Ademais, contatos diferentes dos planares não são editáveis por esse sistema. Apesar de permitir a manipulação de diversos atributos, o sistema não permite que o usuário execute movimentos mais elaborados como rolar, dar cambolhatas, etc., o que acaba limitando a criação dos animadores.

Figura 2.17: Exemplos de diferentes ciclos de animação produzidos pela performance do usuário, capturada por diferentes dispositos. Na figura da esquerda, utiliza-se o *Leap Motion* para gerar um ciclo de um pulo. Na figura central, com os óculos de realidade virtual *HTC Vive*, o usuário produz o ciclo de um pontapé. E a figura da direita exibe ciclos de soco, caminhada e samba criados com um sistema de captura de movimentos do corpo inteiro. Figura extraída de [45].

2.1.7 Interface multitoque

KIPP e NGUYEN [46] propõem a criação de uma interface bimanual intuitiva com tecnologia puramente multitoque. Tal interface propiciaria ao usuário, o gerenciamento dos muitos graus de liberdade dos movimentos de um braço humano 3D. Ela permitir-lhe-ia, ainda, o emprego de ambas as mãos, substituindo outros tipos de interface, como aquelas que oferecem o *mouse* como dispositivo de entrada. A Figura 2.18 ilustra uma aplicação dessa ideia. Nela, pode-se verificar que, além da visualização principal, há duas pequenas visualizações alternativas do boneco. Tais visualizações deveriam melhorar a percepção de profundidade pelo usuário, mas na prática não se obteve muito êxito com esse recurso. Os comandos foram divididos

entre a mão direita, que controla a posição do braço; a mão esquerda, que controla o formato e a orientação da mão virtual; e dois dedos de cada mão, para controlar os múltiplos graus de liberdade dos movimentos. O esqueleto do boneco 3D foi modelado por intermédio de uma estrutura de dados hierarquica em forma de árvore. A localização, orientação e formato das mãos e os movimentos do braço giratório seriam estabelecidos pelas transformações de translação e rotação 3D, obedecendo as mesmas limitações de movimentos da mão e do braço humano. Os movimentos foram gerados por meio de cinemática inversa e o formato das mãos é gerado a partir de modelos predefinidos, como mão espalmada, fechada, dentre outros. Para facilitar o manuseio, os comandos de toque e a visualização estão presentes em uma mesma tela, embora isso não tenha funcionado bem em telas menores, devido à problemas de oclusão. A interface não é colaborativa e ficou restrita tão somente a um utilizador.

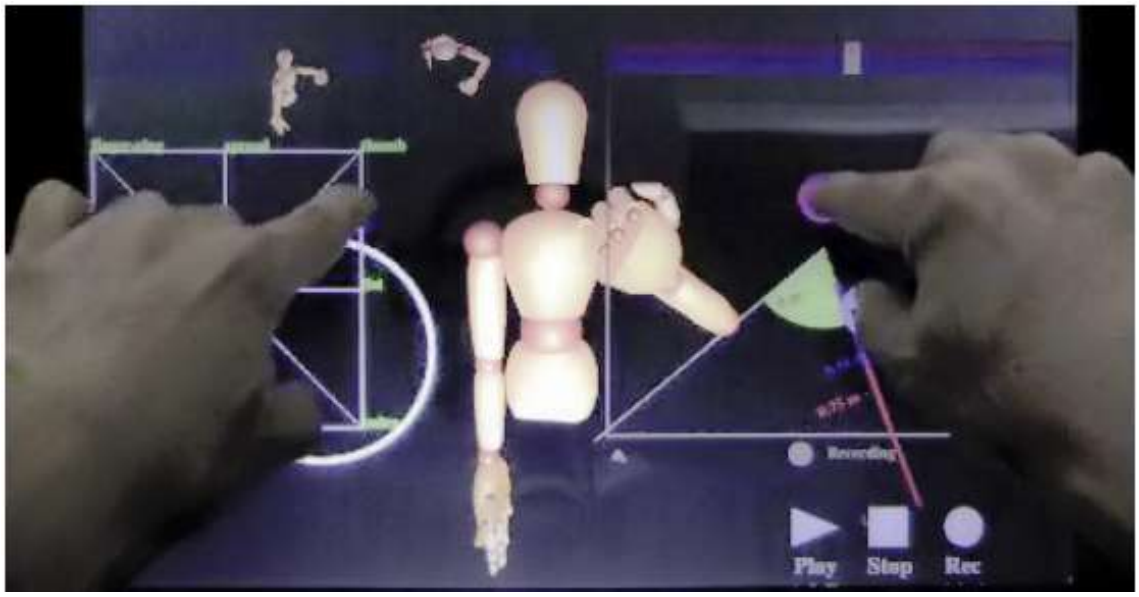


Figura 2.18: Interface multitoque bimanual que permite a animação de um modelo de braço humano 3D articulado. Figura retirada de [46].

2.2 Animação de objetos inarticulados

Nessa seção vamos exibir alguns trabalhos que propõem animar objetos virtuais inarticulados.

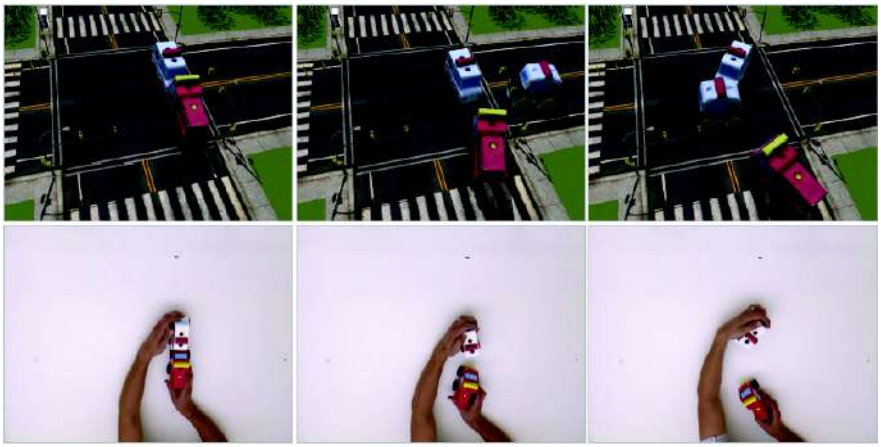
HELD *et al.* [47] apresentam um sistema que utiliza objetos físicos rígidos para produzir animações 3D através da reconstrução e do rastreamento dos movimentos desses objetos. Os objetos físicos – brinquedos, fantoches ou acessórios – são utilizados de duas maneiras: para construir modelos virtuais 3D e para serem manuseados pelo usuário, como ferramenta de animação. Para isso, na fase de pré-processamento,

é necessário que o usuário manipule o objeto na frente de um *Kinect* para construir os modelos virtuais 3D de cada um dos objetos físicos. Após essa fase, o animador deverá, novamente, manipular o objeto físico na frente do *Kinect* para que seja feita a captura dos movimentos dos mesmos. A Figura 2.19 ilustra um animador produzindo uma animação nesse sistema, utilizando brinquedos. Os modelos virtuais são utilizados para estimar a pose do objeto na cena. Isso ocorre, ao aplicar-lhes o algoritmo *Iteractive Closest Point* (ICP) e, assim, permitir o mapeamento do modelo sobre esses objetos. Os modelos 3D são, então, renderizados em tempo real e o animador poderá ajustar os movimentos do objeto durante a animação. O sistema também possibilita que, em qualquer momento da animação, o usuário faça ajustes de câmera, iluminação e *background* da animação renderizada.

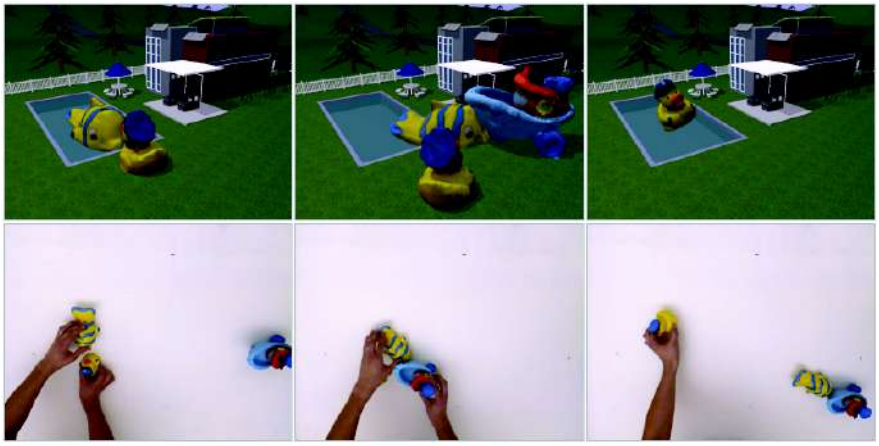
Figura 2.19: A configuração do sistema permite utilizar brinquedos e outros objetos físicos rígidos para realizar uma animação 3D. Figura retirada de [47].

E, quando a animação requer mais de dois bonecos movendo-se ao mesmo tempo, a interface possibilita criá-las em camadas das seguinte forma: realiza-se a captura dos movimentos dos bonecos separadamente para, em seguida, agrupá-las em uma animação única. A Figura 2.20 evidencia essa ideia apresentando duas animações em camadas: em 2.20a um acidente de trânsito envolvendo dois veículos e em 2.20b um intruso na piscina do pato. Algumas limitações estão, notoriamente, presentes no sistema. Ele não funciona com materiais deformáveis, como tecido; e nem com objetos articulados, pois ele não pode controlar os movimentos das juntas. O manipulador também não deverá mover os objetos rapidamente em grandes deslocamentos, pois o sistema poderá não acompanhar os movimentos dos mesmos e, também, deverá mantê-los dentro dos limites de alcance da câmera do *Kinect*.

Um outro sistema voltado para animações 3D é descrito por GUPTA *et al.* [48]. O sistema, chamado de *MotionMontage*, também adota o sensor *Kinect* para gravar,



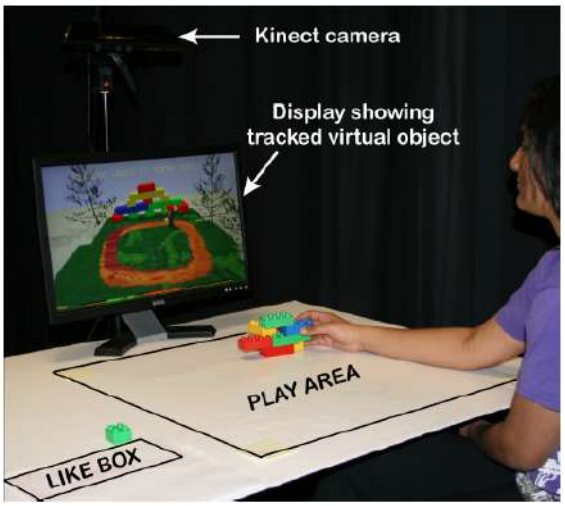
(a) Colisão de trânsito.



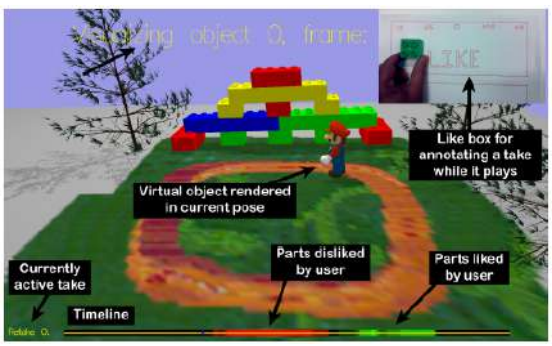
(b) Intruso na piscina.

Figura 2.20: Em cada uma das imagens, a parte superior ilustra os resultados da animação e a parte inferior a visão obtida com o *Kinect*. A animação em (a) ilustra uma colisão de trânsito entre dois veículos e em (b) o pato encontra um intruso em sua piscina. Figura extraída de [47].

em uma área demarcada, o movimento de um objeto rígido e mapeá-lo em um objeto virtual em tempo real. A configuração desse sistema é mostrada na Figura 2.21a. O objeto virtual é escolhido a partir de um banco de dados e o animador utiliza os movimentos de *proxies* físicos para animá-lo. O animador pode anotar as melhores partes de cada tomada, ao longo do tempo e, em seguida, poderá combiná-las, até chegar em uma animação 3D desejada. A Figura 2.21b apresenta o sistema na fase de anotação que funciona da seguinte maneira: o sistema localiza a posição de um bloco verde, que o usuário pode mover na área *Like*, e anota a tomada de acordo com o valor correspondente dessa posição. A Figura 2.21c mostra um resultado da animação. Além disso, o sistema permite múltiplos objetos movendo-se ao mesmo tempo, através de animação em camadas. Assim, como no trabalho visto anteriormente, esse sistema não realiza animações de personagens 3D articulados.



(a)



(b)

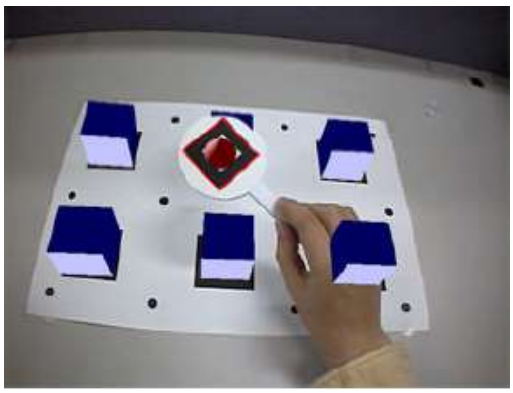
(c)

Figura 2.21: Animador utilizando *proxies* físicos no sistema *MontionMontage* para gravar múltiplas tomadas de uma animação 3D (a). Sistema na fase de anotação (b) e resultado da animação (c). Figura retirada de [48].

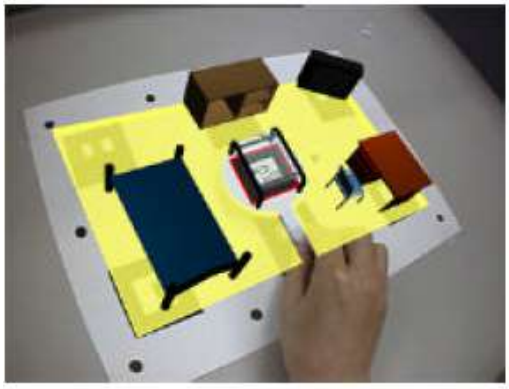
2.2.1 Interface de realidade aumentada

Os autores KATO *et al.* [49] apresentam um trabalho desenvolvido para interação com um objeto virtual e rastreamento do usuário através de uma interface de realidade aumentada sobre um tampo de mesa. Na fase de rastreamento do usuário, eles utilizaram múltiplos marcadores sobre o tampo da mesa, a fim de gerenciar um sistema de coordenadas global. Com esses marcadores e uma câmera acoplada em um *Head-mounted display* (HMD), é possível estimar a pose da câmera e a posição da cabeça do usuário em coordenadas globais. Assim, objetos virtuais 3D são vistos sobre a mesa real, pois eles também podem ser representados em coordenadas de mundo. O usuário pode tocar um objeto que está desenhado sobre o marcador e, mesmo com oclusão parcial, o sistema ainda funcionaria, como mostra a Figura 2.22a. Para que a interface tangível sobre a mesa fosse ainda mais explorada, foi desenvolvida um aplicação em design de interiores, como ilustra a Figura 2.22b. A ideia seria explorar os efeitos dessas interfaces na colaboração entre diversos usuários. Os sistemas de cada um deles não deveriam sofrer interferências e poderiam ser in-

tegrados, mas o trabalho ficou restrito a apenas um utilizador.



(a) Objetos virtuais sobre múltiplos marcadores.



(b) Um protótipo de uma aplicação em Design de interiores.

Figura 2.22: Interação com objetos virtuais sobre múltiplos marcadores em uma mesa, com oclusão gerada por um objeto físico em (a) e uma aplicação em Design de interiores em (b). Figura extraída de [49].

2.2.2 Interface semi-imersiva

DE ARAÙJO *et al.* [50] projetaram uma interface semi-imersiva que combina diferentes espaços de interação para que o usuário possa modelar cenas 3D. A interface possui um modelo bimanual assimétrico e permite executar o rastreamento das mãos e dos dedos do usuário em um certo espaço contínuo tridimensional, delimitado acima da superfície de uma mesa multitoque. Essa mesa, por sua vez, permitir-lhe-ia criar esboços 2D em sua superfície. Incorporando alguns gestos para manipular os esboços criados, o usuário poderá elaborar e editar em tempo real maquetes virtuais 3D sobre a superfície da mesa, em um ambiente estereoscópico. A configuração do sistema proporciona a união de dados captados por diferentes sensores.

A Figura 2.23 apresenta essa configuração com os seus respectivos dispositivos: um *Kinect*, dois controles tridimensionais de jogos *Gametrak*, um transmissor infravermelho estéreo, uma mesa multitoque com projetor 3D e um óculos NVIDIA 3D *Vision* receptor infravermelho. Esses dispositivos são utilizados para que o usuário possa realizar um pequeno conjunto de operações da seguinte maneira: a mesa multitoque possibilita que sejam realizados gestos 2D sobre a sua superfície; o *Gametrak* captura os gestos 3D executados no espaço acima da superfície da mesa; o *Kinect*, por intermédio do seu algoritmo de captura do esqueleto, rastreia a cabeça e localiza a mão dominante do usuário; o transmissor infravermelho, o projetor e o óculos 3D são utilizados para a visualização estereoscópica da cena 3D. A captura em tempo real dos gestos é feita diretamente através do rastreamento dos dedos, tanto na mesa quanto pelo *Gametrak*. Esse último, ligado aos dedos do usuário por cordas retráteis

e anéis, tem a função de rastrear a posição 3D do dedo indicador e do polegar de cada mão quando eles não estão mais em contacto com a superfície multitoque. Foi utilizado o filtro 1€ [51] para minimizar a instabilidade e atraso nos dados fornecidos pelo *Gametrak*. Os dados gerados pelo rastreamento dos dedos são utilizados para, incrementalmente, construir linhas e curvas de Bézier cúbicas, gerando, assim, traçados e gestos com trajetórias suaves. Um reconhecedor interativo de formas geométricas 2D baseado na lógica *fuzzy* foi utilizado para detectar círculos e elipses, aproximados por uma curva fechada por partes que utiliza quatro segmentos cúbicos de Bézier e, também, para detectar um gesto de apagar usado para excluir formas ou traços. Semelhante ao gesto de agarrar objetos físicos, a seleção dos objetos é realizada por intermédio de um gesto de pinça no espaço. Uma forma inteira é selecionada quando o dedo intersecta a sua *bounding box*. Quando há compartilhamento de elementos (vértices, arestas e/ou faces) por diversos objetos, os mesmos devem ser destacados previamente através da interseção com o dedo, para que não haja conflito de prioridade na seleção desses componentes. A transposição de gestos na superfície da mesa para o espaço que permite o rastreamento de gestos 3D pode ser feita acessando-se um operador disponível em um menu contextual posicionado sobre a própria mesa e operado pela mão não dominante. O menu também permite cortar parte da cena, utilizando planos de recorte tradicionais da *OpenGL* [52], para que possíveis faces ocluídas sejam exibidas.

A Figura 2.24a demonstra detalhes do sistema, com uma vista detalhada das cordas do *Gametrak* ligadas aos dedos e o menu com as teclas usadas para gestos de pinça e a Figura 2.24b mostra um resultado obtido ao utilizar esse sistema. As cordas ligadas ao dedo podem interferir a performance do sistema, prejudicando tanto a vizualização quanto a execução dos movimentos; o que difere de outros dispositivos de baixo custo que executam o rastreamento dos movimentos das mãos, estando elas livres de quaisquer acessórios, como é o caso do *Leap Motion*.

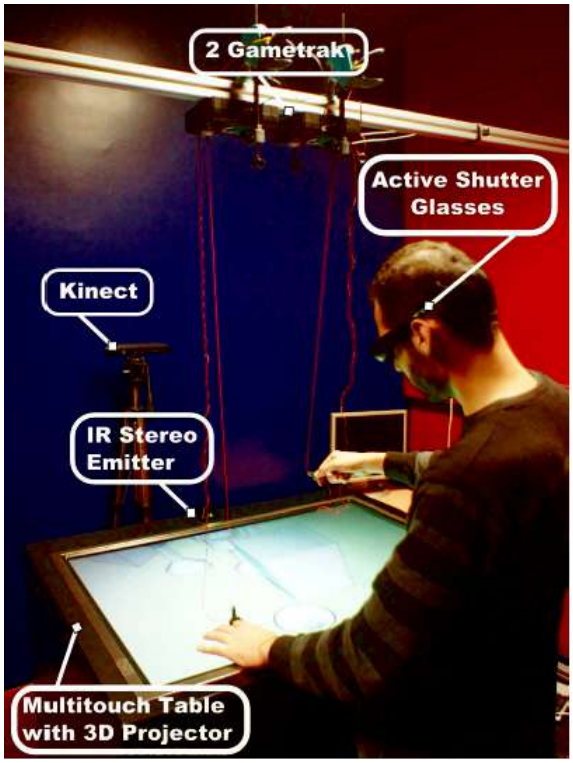
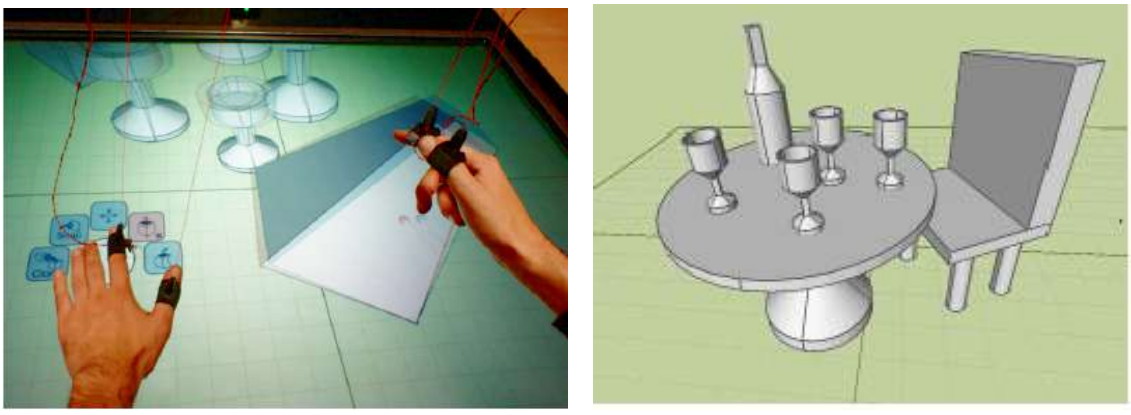


Figura 2.23: Configuração do ambiente semi-imersivo com a união de diversos dispositivos atuais. Figura retirada de [50].



(a) Manipulação. (b) Maquete 3D.

Figura 2.24: Em (a), podemos acompanhar a fase de criação, edição e manipulação na interface bimanual assimétrica para obter uma maquete virtual 3D, como a ilustrada na Figura (b). Figura retirada de [50].

Capítulo 3

Fundamentação teórica

Neste capítulo apresentaremos brevemente os conceitos fundamentais imprescindíveis ao desenvolvimento desse estudo. Em especial, estudaremos curvas e superfícies regulares, as quais serão adotadas na construção do nosso método. Esses conceitos serão referenciados, sempre que necessário, no Capítulo 5. Mais detalhes sobre os temas podem ser encontrados em [53–55].

3.1 Curvas

Nesta seção, estudaremos o conceito de curvas diferenciáveis em $\mathbb{R}^n$.

Definição 1. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função. Diz-se que f é* diferenciável *se ela possui derivadas contínuas de todas as ordens.*

Definição 2. *Seja $I = (a, b) \subset \mathbb{R}$. Uma* curva diferenciável parametrizada *C é uma aplicação $\alpha : I \subset \mathbb{R} \to \mathbb{R}^n$, dada por*

$$\alpha(t) = (x_1(t), x_2(t), ..., x_n(t)), \tag{3.1}$$

com $x_1, x_2, ..., x_n$ diferenciáveis.

A imagem $\alpha(I)$ é denominada *traço da curva C* e a variável t, o *parâmetro*. Adotamos a notação $x_i'(t)$ para representar a *derivada primeira* de $x_i(t)$ em um ponto t, com $i = 1, 2, ..., n$.

Exemplo 1. A curva parametrizada

$$\alpha(t) = (r\cos(t), r\,\mathrm{sen}(t)), \quad t \in [0, 2\pi] \tag{3.2}$$

é uma *curva circular* no plano em torno da origem (0,0), com raio $r > 0$ e possui como traço a circunferência $x^2 + y^2 = r^2$, centrada na origem, conforme a Figura 3.1.

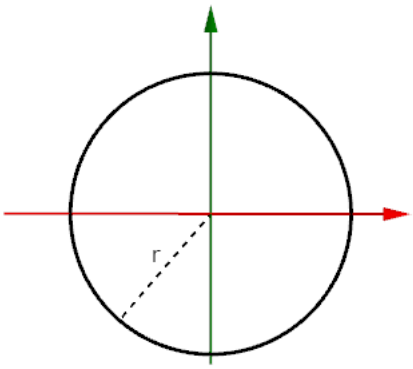


Figura 3.1: Circunferência de raio r centrada na origem.

Definição 3. *Seja $\alpha : \mathbb{R} \to \mathbb{R}^n$ uma curva diferenciável parametrizada. Então, o vetor*

$$\alpha'(t) = (x_1'(t), x_2'(t), ..., x_n'(t)) \tag{3.3}$$

é dito vetor tangente *(ou vetor velocidade) da curva α em t.*

Definição 4. *Uma curva diferenciável parametrizada $\alpha : I \to \mathbb{R}^n$ é chamada* regular *se $\alpha'(t) \neq 0$ para todo $t \in I$. Nesse caso, o ponto $\alpha(t)$ é dito ser* regular*. Caso $\alpha'(t) = 0$, diz-se que $\alpha(t)$ é um* ponto singular*.*

Exemplo 2. No exemplo 1, o vetor tangente à curva dada pela equação 3.2 em t é igual a:

$$\alpha'(t) = (-r\,\mathrm{sen}(t), r\cos(t)) \tag{3.4}$$

Observe que, $\alpha'(t) \neq 0$ para todo $t \in [0, 2\pi]$, portanto, essa curva também é regular.

3.2 Superfícies

Nesta seção faremos uma descrição sucinta de superfícies no espaço euclidiano. Para maiores detalhes ver [53]

Definição 5. *Um subconjunto $S \subset \mathbb{R}^3$ é uma* superfície regular *se, para cada ponto $p \in S$, existe uma vizinhaça aberta V de p em $\mathbb{R}^3$ e, se existe um subconjunto aberto $U \subset \mathbb{R}^2$ sobre $V \cap S \subset \mathbb{R}^3$ e uma aplicação suave $\mathbf{x} : U \to \mathbb{R}^3$ tal que:*

(i) $\mathbf{x}$ *é diferenciável,*

(ii) $\mathbf{x}$ *é um homeomorfismo, isto é, $\mathbf{x}$ tem uma inversa $\mathbf{x}^{-1} : V \cap S \to U$ contínua*

(iii) *a matriz Jacobiana $D_q\mathbf{x}$ tem posto 2 para cada ponto $q \in U$.*

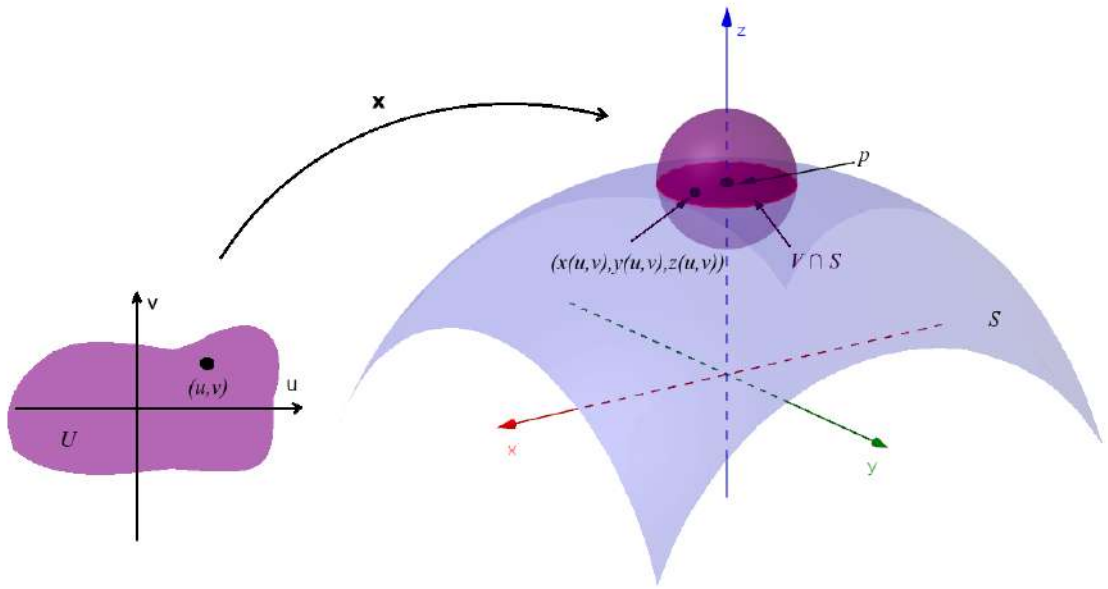


Figura 3.2: Superfície regular

A aplicação $\mathbf{x}$ é chamada uma *parametrização* em uma vizinhança do ponto p e, podemos escrevê-la como uma terna de funções da seguinte forma:

$$\mathbf{x}(u,v) = (x(u,v), y(u,v), z(u,v)), \quad (u,v) \in U. \tag{3.5}$$

Definição 6. *Se $S \subset \mathbb{R}^3$ é uma superfície que tem uma parametrização $\mathbf{x}(u,v)$, então uma* curva de superfície *tem parametrização da seguinte forma:*

$$\alpha(t) = \mathbf{x}(u(t), v(t)), \quad t \in I \tag{3.6}$$

Definição 7. *Sejam $S \subset \mathbb{R}^3$ é uma superfície que tem uma parametrização $\mathbf{x}(u,v)$ e (u_0, v_0) um ponto fixo em $U \subset \mathbb{R}^2$. As* curvas coordenadas *são curvas de superfície nas quais um dos parâmetros, u ou v, é constante e são dadas por:*

$$\begin{aligned} \alpha_{u_0}(t) &= \mathbf{x}(u_0, v(t)), \\ \alpha_{v_0}(t) &= \mathbf{x}(u(t), v_0), \end{aligned} \tag{3.7}$$

com $t \in I$.

Exemplo 3. Seja $U = \{(u,v);\ 0 \leq u \leq \pi,\ 0 \leq v < 2\pi\}$. $\mathbf{x} : U \to \mathbb{R}^3$ dada por

$$\mathbf{x}(u,v) = (r\,\mathrm{sen}u\cos v, r\,\mathrm{sen}u\,\mathrm{sen}v, r\cos u) \tag{3.8}$$

é uma parametrização da *esfera* centrada na origem em $\mathbb{R}^3$, com raio $r > 0$. As

curvas coordenadas são dadas pelos paralelos e pelos meridianos da esfera (ver a Figura 3.3) e possuem formato circular.

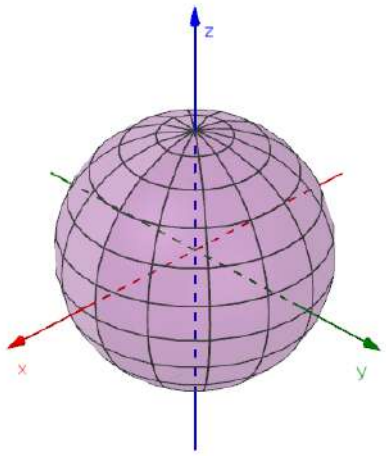


Figura 3.3: Esfera.

Definição 8. *Seja um ponto $p \in U$. As derivadas parciais de $\mathbf{x}(u,v)$ em relação a u e a v equivalem aos* vetores tangentes *no ponto p e são, respectivamente, dadas por:*

$$\mathbf{x}_u(u,v) = \left(\frac{\partial x}{\partial u}(u,v), \frac{\partial y}{\partial u}(u,v), \frac{\partial z}{\partial u}(u,v)\right) \tag{3.9}$$

$$\mathbf{x}_v(u,v) = \left(\frac{\partial x}{\partial v}(u,v), \frac{\partial y}{\partial v}(u,v), \frac{\partial z}{\partial v}(u,v)\right) \tag{3.10}$$

Os vetores tangentes são os mesmos vetores colunas da matriz Jacobiana mencionada na definição 5.

Como $\mathbf{x}$ é uma aplicação diferenciável, as funções $x(u,v)$, $y(u,v)$ e $z(u,v)$ possuem derivadas parciais contínuas de todas as ordens em U.

Definição 9. *Seja $\mathbf{x} : U \to \mathbb{R}^3$ uma parametrização da superfície regular S. Um* vetor normal *à S em um ponto $p \in \mathbf{x}(U)$ é dado por:*

$$n(p) = (\mathbf{x}_u \times \mathbf{x}_v)(p) \tag{3.11}$$

Definição 10. *Um* vetor normal unitário *é determinado por:*

$$N(p) = \frac{n(p)}{||n(p)||}, \quad p \in \mathbf{x}(U) \tag{3.12}$$

3.2.1 Superfícies cilindrícas

As superfícies cilíndricas são aquelas que podem ser geradas pela translação de uma curva C na direção de um vetor em $\mathbb{R}^3$.

Definição 11. *Seja C uma curva diretriz com uma parametrização $\alpha(u) = (x_1(u), x_2(u), x_3(u))$, $u \in I$ e um vetor diretor $\vec{v} = (a, b, c)$. Então, uma* parametrização da superfície cilíndrica *S tem como equações:*

$$\begin{aligned} x(u,v) &= x_1(u) + av, \\ y(u,v) &= x_2(u) + bv, \\ z(u,v) &= x_3(u) + cv, \end{aligned} \tag{3.13}$$

com $v \in I$.

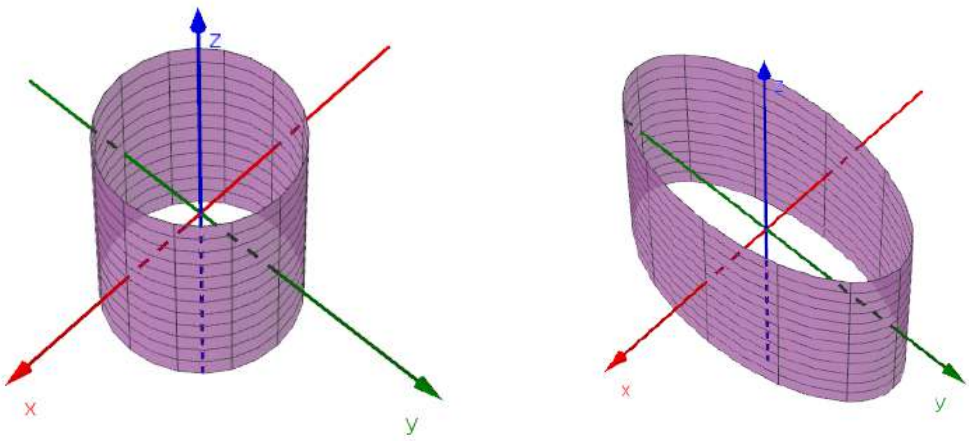
Quando a diretriz tem o traço conhecido, como um círculo, uma elipse, uma parábola ou uma hipérbole, a superfície cilíndrica pode ser circular, elíptica, parabólica ou hiperbólica, conforme a diretriz empregadas. Na Figura 3.4 podemos observar duas dessas superfícies, à esquerda temos um cilindro circular (Figura 3.4a) e à direita, um cilíndro elíptico (Figura 3.4b).

Exemplo 4. Um *cilindro circular reto* que possui uma curva diretriz como a vista no Exemplo 1 e vetor diretor $\vec{v} = (0, 0, 1)$, pode ser descrito com as seguintes equações paramétricas:

$$\begin{cases} x(u,v) = r\cos(u) \\ y(u,v) = r\,\mathrm{sen}(u) \\ z(u,v) = v \end{cases} \tag{3.14}$$

com os parâmetros $u, v, r \in \mathbb{R}$, tal que $0 \leq u < 2\pi$ e r igual ao raio do círculo.

Nesse caso, o eixo do cilindro está ao longo do eixo Oz e as curvas coordenadas são circunferências paralelas ao plano xy e segmentos de retas paralelas ao eixo, como podemos observar na Figura 3.4a.



(a) Cilindro circular reto. (b) Cilindro elíptico reto.

Figura 3.4: Superfícies cilindrícas.

3.2.2 Superfícies de Revolução

As superfícies de revolução são geradas mediante a rotação de uma curva plana em torno de uma reta fixa em $\mathbb{R}^3$.

Definição 12. *Seja* Π *um plano em* $\mathbb{R}^3$*. Se tomarmos uma reta* l *e uma curva plana* C*, ambas em* Π*, ao rotacionar* C *em* $\mathbb{R}^3$ *em torno de* l*, o conjunto de pontos resultantes* S *é chamado de* superfície de revolução *gerada por* C*. A curva* C*, por sua vez, é denominada* curva geratriz *e a reta* l *é intitulada de* eixo de revolução*.*

Por conveniência, adota-se o plano xz para ser o plano Π e o eixo Oz como eixo de revolução.

Definição 13. *Seja uma curva plana* C *dada por* $\alpha(v) = (\varphi(v), 0, \psi(v))$ *e seja* u *o ângulo de rotação dessa curva em torno do eixo* Oz*. A aplicação*

$$\begin{aligned} x(u,v) &= \varphi(v)\cos(u), \\ y(u,v) &= \varphi(v)\,\mathrm{sen}(u), \\ z(u,v) &= \psi(v) \end{aligned} \tag{3.15}$$

do conjunto aberto $U = \{(u,v) \in \mathbb{R}^2;\ 0 \leq u < 2\pi,\ a < v < b\}$ *em* S *é uma* parametrização padrão da superfície de revolução S*.*

3.3 Sistemas de coordenadas

O sistema de coordenadas cartesianas é um dos mais utilizados para descrever pontos no espaço. Mas, além dele, existem outros sistemas de coordenadas que podem ser empregados com a mesma finalidade. Podemos tomar, por exemplo, coordenadas esféricas e cilíndricas, as quais serão descritas, respectivamente, nas subseções 3.3.1 e 3.3.2.

Genericamente, considera-se um ponto fixo O na origem do sistema de coordenadas, um plano de referência e um eixo de referência contido no plano, ambos passando por esse mesmo ponto O. Com isso, pode-se verificar a relação entre o sistema de coordenadas cartesianas e os demais sistemas de coordenadas.

3.3.1 Relação entre coordenadas cartesianas e coordenadas esféricas

Aqui, os pontos são descritos através de suas posições em esferas.

Para determinar a relação entre as coordenadas cartesianas (x, y, z) e esféricas (r, θ, ϕ), considera-se o ponto fixo O como sendo a origem do sistema de coordenadas

cartesianas; o plano de referência será determinado pelo plano Oxz e o eixo de referência como sendo o eixo Oz.

Para todo ponto $P \neq O$, r é a distância de P a O, θ é o ângulo do raio OP com o eixo de referência Oz, e ϕ o ângulo entre o plano de referência Oxz e o plano que contém o eixo Oz e o ponto P, conforme ilustrado na Figura 3.5. O ângulo ϕ coincide com o ângulo formado pela projeção ortogonal de OP sobre o plano Oxy e o eixo Ox.

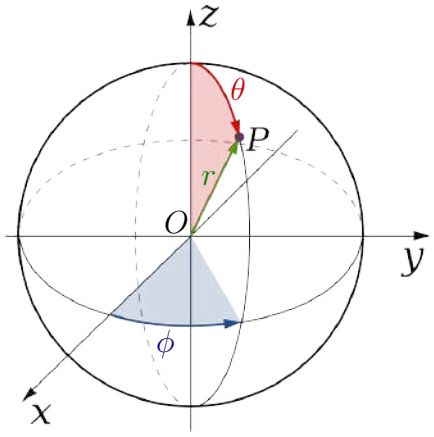


Figura 3.5: Relação entre as coordenadas cartesianas e esféricas de um ponto P.

Com isso, a relação entre as coordenadas cartesianas (x, y, z) e esféricas (r, θ, ϕ) fica estabelecida das seguintes maneiras:

$$\begin{cases} r = \sqrt{x^2 + y^2 + z^2}, \\ \theta = \arccos\left(\dfrac{z}{\sqrt{x^2 + y^2 + z^2}}\right), \\ \phi = \arctan\left(\dfrac{y}{x}\right) \end{cases} \qquad x, y, z \in \mathbb{R}. \tag{3.16}$$

e

$$\begin{cases} x = r\,\mathrm{sen}\theta\cos\phi \\ y = r\,\mathrm{sen}\theta\,\mathrm{sen}\phi, \quad r > 0,\ 0 \leq \theta \leq \pi \ \text{ e } \ 0 \leq \phi < 2\pi. \\ z = r\cos\theta \end{cases} \tag{3.17}$$

O arco tangente deve ser definida considerando o quadrante correto no qual (x, y) encontra-se.

3.3.2 Relação entre coordenadas cartesianas e coordenadas cilíndricas

Nesse caso, a posição dos pontos é delimitada ao longo de cilindros. Denota-se por O a origem do sistema de coordenadas, adota-se o eixo de referência como Oz e

o plano de referência como Oxz. Assim, para cada ponto P, toma-se o cilindro circular reto de raio r e eixo Oz. Considera-se o plano Oxy, perpendicular a Oz por O; z, a distância de P ao plano Oxy e ϕ, o ângulo do plano passando por Oz e P e o plano Oxz.

As coordenadas cartesianas (x, y, z) e cilíndricas (r, ϕ, z) em P, relacionam-se da seguinte forma:

$$\begin{cases} r = \sqrt{x^2 + y^2}, \\ \phi = \arctan\left(\dfrac{y}{x}\right), \\ z = z \end{cases} \qquad x, y, z \in \mathbb{R}. \tag{3.18}$$

e

$$\begin{cases} x = r\cos\phi \\ y = r\,\mathrm{sen}\phi, \quad r > 0,\ 0 \leq \phi < 2\pi \ \text{ e } \ z \in \mathbb{R}. \\ z = z \end{cases} \tag{3.19}$$

A tangente inversa deve ser definida considerando o quadrante correto onde encontram-se as coordenadas (x, y).

Capítulo 4

Quadros-chaves Espaciais (*Spatial Keyframing*)

Nesse capítulo faremos uma exposição do sistema que mais nos inspirou no decorrer da construção desse trabalho, o *Spatial Keyframing*. Abordaremos em mais detalhes os seus fundamentos para desvendar como se processa o seu funcionamento.

Como visto brevemente no Capítulo 1, o sistema de animação *Spatial Keyframing* proposto por IGARASHI *et al.* [11] possibilita criar animações de personagens articulados 3D utilizando quadros-chaves espaciais. Tais animações são baseadas na performance do usuário, que utiliza um dispositivo 2D, como um mouse, para controlar as animações. O *Spatial Keyframing* é composto por dois subsistemas: um para criar os quadros-chaves espaciais e outro para construir animações interpolando esses mesmos quadros-chaves, com isso, concebe-se novos quadros em tempo real, através da performance do usuário. Nas seções seguintes, estudaremos em mais detalhes cada um desses subsistemas.

4.1 Construção de quadros-chaves espaciais

Nessa fase, deve-se executar duas etapas determinantes para a composição de um quadro-chave:

1. *Pose:* é o conjunto composto pela posição, escala e orientação de cada uma das juntas do personagem, com relação à junta pai;

2. *Marcador:* é a posição no espaço tridimensional, à qual cada pose é associada.

Utilizando a interface criada pelos autores para executar cada uma das etapas acima, antes de mais nada, precisa-se importar um personagem articulado 3D. Com isso, o sistema permitirá, inicialmente, gerar as diferentes poses do personagem.

Estas, por sua vez, devem ser especificadas manualmente pelo animador. Utilizando um mouse, ele poderá rotacionar as juntas e/ou ossos do personagem, até que se consiga um resultado considerável. Ademais, pode-se, ainda, modificar a posição espacial do personagem em questão, realizando um movimento de translação ao arrastá-lo com o mouse.

Após concluir a primeira etapa, segue-se com a criação do marcador. Nessa etapa, o animador deve mover o botão controlador, representado por uma esfera magenta na interface e posicioná-lo em um local arbitrário do espaço tridimensional, gerando assim, um marcador que, por sua vez, é designado por uma esfera amarela.

A materialização do marcador, sinaliza a criação de um quadro-chave espacial. Cada um deles consiste da união desses dois elementos descritos acima: a pose do personagem e as coordenadas 3D que correspondem à posição do marcador.

A Figura 4.1 apresenta uma animação de um personagem caminhando, elaborada com quatro quadros-chaves espaciais. Na parte superior da figura, podemos verificar cada uma das poses do personagem, quando a posição do botão controlador (esfera magenta) coincide com a posição do marcador (esfera amarela), já que todos esses elementos compartilham de um mesmo plano. A parte inferior da figura exibe partes de uma animação resultante da movimentação do botão controlador.

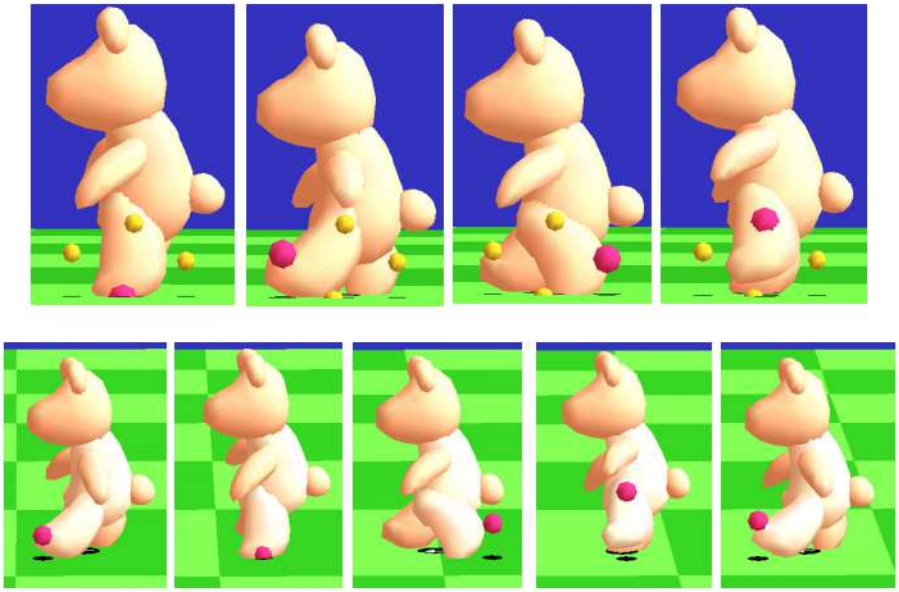


Figura 4.1: Uma animação construída com quatro quadros-chaves espaciais. Na parte superior, cada quadro-chave é associado com um marcador, representado por uma esfera amarela. Novas poses (parte inferior), geradas durante a performance do usuário ao manipular o botão controlador (esfera magenta). Figura retirada de [11].

Para gerar novas poses e seus respectivos marcadores associados, o animador poderá repetir essas etapas para tantos quadros-chaves quantos forem necessários

para produzir a animação, gerando assim, um conjunto de quadros-chaves espaciais. Logo, esse conjunto será n-dimensional, respeitando o mínimo de três quadro-chaves com, ao menos, uma das poses distinta das demais. Esse número mínimo de quadros-chaves é exigido pelo sistema para que a animação não seja estática, ou seja, para que ela apresente algum movimento.

4.2 Animação através da perfomance do usuário

Após realizar todas as etapas descrita na seção 4.1 deste capítulo e o conjunto de quadros-chaves espaciais já estiver sido configurado, pode-se produzir as animações. Para tal, o animador deverá manipular o botão controlador, arrastando-o com o auxílio de um mouse. Através da performance do usuário, o sistema sintetiza uma sequência de animação interpolando, ininterruptamente, as poses mais próximas. Nessa fase não é possível gerar novos quadros-chaves espaciais diretamente, mas apenas sintetizar novas poses via computador.

Diante disso, fica evidente que o resultado da animação depende, conjuntamente, dos movimentos realizados pelo animador e dos quadro-chaves espaciais pré-configurados.

A Figura 4.2 mostra um outro exemplo que, dessa vez, fora produzido com seis quadros-chaves espaciais (parte superior). Ao manipular o botão controlador (esfera magenta), novas poses são geradas por intermédio de uma interpolação dos quadros-chaves (parte inferior). Quanto mais próximo o botão controlador estiver de um marcador (esfera amarela), mais parecida com a pose do respectivo quadro-chave a nova pose será.

A interface também permite gravar a animação. O resultado pode ser assistido quantas vezes o usuário desejar e de diversas direções, apenas modificando a posição da câmera.

4.3 Algoritmo

Os autores apresentam o algoritmo, não como o mais eficiente para combinar os quadros chaves e gerar novas poses por interpolação; mas, tão somente, para que torne-se possível implementar tais ideias de uma forma mais fácil. O algoritmo pode ser sintetizado da seguinte maneira:

- **Objetivo:** criar animações obtendo novas poses ao interpolar quadros-chaves espaciais em tempo real;
- **Entrada:** coordenada (x, y, z) do cursor do mouse e um conjunto de quadros-chaves espaciais;

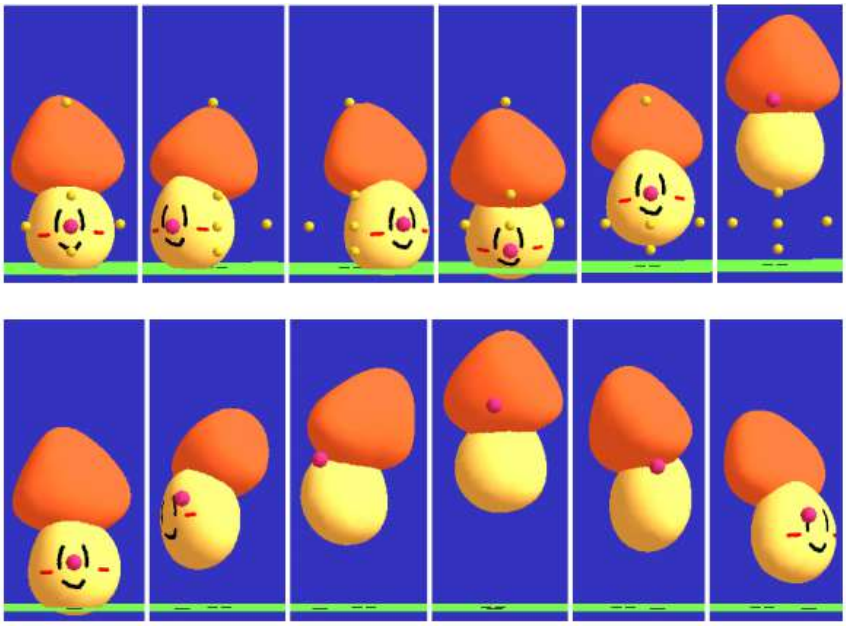


Figura 4.2: Sequência de animação (parte inferior) com novas poses geradas mediante interpolação de poses dos seis quadros-chaves espaciais (parte superior). Figura retirada de [11].

- **Saída:** nova pose do personagem.

O modelo articulado 3D constitui-se de múltiplas partes rígidas vinculadas por uma estrutura hierárquica. A pose de uma junta é definida com relação à junta pai por sua posição, orientação e escala. A pose do personagem é definida como um conjunto de transformações locais, formado pelas poses relativas de cada junta.

O sistema permite transladar todo o esqueleto apenas pela raiz. Assim, o mesmo é composto por matrizes de rotação 3×3 para todas as juntas e um vetor de translação para a raiz. Cada entrada das matrizes é interpolada por uma função de base radial (RBF), a qual será detalhada na subseção seguinte. Ademais, a matriz resultante deve ser ortonormalizada, pois não se pode garantir que a matriz interpolada obtida é uma matriz de rotação ortogonal.

4.3.1 Interpolação

O método de interpolação adotado foi o método proposto por TURK e O'BRIEN [56]. Nesse método, cada elemento de cada uma das matrizes é visto como uma função real no espaço de controle. Tal função é dada por:

$$f(\mathbf{x}) = \sum_{j=1}^{n} d_j \Phi(\mathbf{x} - c_j) + P(\mathbf{x}) \tag{4.1}$$

onde $\Phi(\mathbf{x})$ é uma função de base radial, c_j é a posição do marcador, d_j é o peso e $P(\mathbf{x})$ é um polinômio de primeiro grau.

A função base adotada foi $\Phi(\mathbf{x}) = |x|$ e foi escolhida de modo experimental. Embora ela não seja tão suave, ela apresenta resultados bastantes satisfatórios ao interpolar utilizando a posição do ponteiro do mouse.

O sistema deve ser solucionado para cada peso d_j, quando $f(x)$ representa uma certa pose em seu marcador associado. Supondo que $h_j = f(c_j)$, pode-se representar tal restrição como

$$h_i = \sum_{j=1}^{n} d_j \Phi(c_i - c_j) + P(c_i) \tag{4.2}$$

Como cada peso d_j e os coeficientes de $P(x)$ são lineares, então a equação acima pode ser formulada como um sistema linear, da seguinte forma:

$$\begin{bmatrix} \Phi_{11} & \cdots & \Phi_{1n} & 1 & c_1^x & c_1^y & c_1^z \\ \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ \Phi_{n1} & \cdots & \Phi_{nn} & 1 & c_n^x & c_n^y & c_n^z \\ 1 & \cdots & 1 & 0 & 0 & 0 & 0 \\ c_1^x & \cdots & c_n^x & 0 & 0 & 0 & 0 \\ c_1^y & \cdots & c_n^y & 0 & 0 & 0 & 0 \\ c_1^z & \cdots & c_n^z & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ \vdots \\ d_{1n} \\ p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

onde $c_i = (c_i^x, c_i^y, c_i^z)$, $\Phi_{ij} = \Phi(c_i - c_j)$ e $P(\mathbf{x}) = p_0 + p_1 x + p_2 y + p_3 z$.

Ao solucionar o sistema linear acima, obtém-se o valor da função de interpolação $f(x)$. É necessário solucioná-lo para cada elemento da matriz de rotação 3×3. Observando que a matriz dos coeficientes aumentada é a mesma para todas as nove entradas, torna-se necessário calcular a inversa da matriz aumentada somente uma vez para cada junta. No caso da raiz, utiliza-se esse mesmo método para calcular em cada entrada do vetor de translação.

Os autores atestam que utilizar esse sistema seja mais vantajoso do que um sistema baseado em quadros-chaves temporal, pois ele é rápido, divertido e eficiente ao que se propõe. No entanto, ele ainda não é tão eficaz para realizar animações 3D. Como, as animações são projetadas em uma tela plana, uma das desvantagens é que perde-se a noção de profundidade do controlador e, não se sabe ao certo quais marcadores têm mais influência na interpolação. Embora o sistema permita a edição da terceira dimensão através da sombra do marcador, foi utilizado um dispositivo de entrada bidimensional, no caso um mouse, para controlar as animações. Visto que o mouse é um dispositivo 2D, isso acaba gerando uma desvantagem, como por exemplo a perda de alguns graus de liberdade. Ademais, isso acaba por reduzir o espaço dos marcadores. O ideal seria utilizar dispositivos de entrada 3D para facilitar e aprimorar a produção de animações de personagens articulados 3D.

Por outro lado, utilizar o sistema em 3D requer um cuidado e uma atenção maior

em relação à profundidade, pois o controlador e os marcadores estão projetados em uma tela 2D. Além de perder a noção de profundidade do controlador, é difícil identificar quais marcadores estão sendo interpolados em 3D.

A ideia aqui é, justamente, utilizar um dispositivo de entrada 3D *hands free* para que, por meio de captura de gestos e de movimentos das mãos, possa elaborar animações em tempo real. A velocidade e o resultado final da animação seriam controlados pela própria performance do usuário ao realizar determinados gestos e movimentos. Além disso, com os marcadores dispersos não necessariamente sobre um mesmo plano no espaço tridimensional, pode-se configurar diversos estilos de animação; o personagem pode, por exemplo, caminhar normalmente como um jovem, ou como um idoso, em uma mesma animação. No Capítulo 5 faremos uma exposição mais detalhada da ideia, ao descrever o método proposto.

Capítulo 5

Método Proposto

Neste capítulo apresentaremos as ideias que foram desenvolvidas no decurso deste trabalho. Descreveremos cada uma das etapas presentes na concepção do método nas seções seguintes.

A nossa proposta é impulsionar a criação de animações de personagens articulados 3D utilizando quadros-chaves espaciais, tomando por base o trabalho proposto por IGARASHI *et al.* [11]. Da mesma forma que o método seminal, o nosso algoritmo tem como dados de entrada um conjunto de quadros-chaves espaciais, compostos por poses associadas aos seus respectivos marcadores posicionados no espaço tridimensional, e um controlador; os quais serão descritos, respectivamente, nas Seções 5.1 e 5.2. Como resultado, o algoritmo fornece novas poses para gerar animações baseadas na performance do usuário. Isso tornar-se-á possível, à medida que o sistema interpola as poses-chaves e fornece novas poses em tempo real.

Em nosso sistema, para manipular o controlador, o usuário deverá executar gestos e movimentos naturais com a mão, que serão capturados por um sensor de profundidade. A posição da mão do usuário será mapeada na posição 3D do controlador no espaço virtual, em tempo real, ao passo que ele movimentá-la. Além do resultado final, o movimento também estabelecerá a velocidade da animação.

Geralmente, o movimento executado com a mão segue, naturalmente, o traço aproximado de uma curva conhecida, a qual poderá ser associada à curvas contidas em superfícies, preferencialmente curvas coordenadas, o que facilitará ainda mais essa associação e a tornará mais evidente. Nós acrescentamos como dado de entrada do algoritmo, superfícies para apoiar os marcadores, pois acreditamos que elas facilitam a localização em 3D de diversos objetos na cena, como o controlador e os próprios marcadores. Dessa forma, as superfícies foram pensadas de modo que contivessem curvas coordenadas simples, que pudessem ser facilmente associadas ao traçado do movimento da mão do usuário e que, de alguma forma, também estivessem naturalmente associadas com o movimento ou ação a ser animada, por exemplo, curvas circulares para animações cíclicas. Tais superfícies serão específicadas na

Seção 5.3.

Para manipular objetos oriundos de sistemas de coordenadas diversos na cena, em alguns momentos, precisamos realizar transformações de coordenadas e, assim, estabelecer uma relação simples entre eles. Na Seção 5.4 vamos expor mais detalhes dessas transformações.

Propomos ainda, um método para criar animações mais sofisticadas, com diversos ciclos de movimentos de estilos diferentes em uma mesma animação, ou que mistura mais de uma ação, em tempo real, sem a necessidade de interromper o sistema para configurar novos quadros-chaves. Nesse caso, é possível utilizar uma ou mais superfícies, cada uma associada a um conjunto de marcadores para realizar uma determinada ação ou estilo de movimento. Além disso, vamos realizar transições suaves entre esses movimentos com estilos ou ações diferentes, através de interpolações globais e locais. Elucidaremos esse fato na Seção 5.5.

Como recursos adicionais, nós criamos um método para suavizar o traçado da curva resultante do movimento da mão do usuário e, para garantir um traço ainda mais preciso e próximo dos marcadores, nós projetamos a posição do controlador sobre a superfície para assegurar resultados mais próximos das poses originais.

5.1 Quadro-chaves espaciais

Na Seção 4.1 do Capítulo 4 vimos que, precisamos de no mínimo três quadros-chaves espaciais para construir uma animação e que, cada quadro-chave é composto por dois elementos básicos: uma pose e um marcador associado à tal pose. De posse desses conhecimentos, o que é preciso, antes de mais nada, é fornecer esses dois elementos para compor cada quadro-chave espacial adotado para conceber as animações, os quais serão descritos, respectivamente, nas Subseções 5.1.1 e 5.1.2.

5.1.1 Pose

O esqueleto humano é formado por centenas de ossos. Esses, por sua vez, são ligados por articulações ou juntas. Sabemos que os movimentos dos ossos, são limitados. Em algumas articulações, como as dos cotovelos e joelhos, os movimentos do antebraço e da perna, respectivamente, são executados com apenas um grau de liberdade; já em relação à cabeça temos três graus de liberdade. Modelar esses movimentos de maneira realista torna-se uma tarefa um tanto quanto complexa e, consequentemente, um dos grandes desafios da animação computacional. Tais limitações tornam-se ainda mais evidentes, em se tratando de animação de esqueletos humanóides através de quadros-chaves. Para limitar esses movimentos e torná-los o mais natural possível, pode-se dispor de técnicas como cinemática direta ou inversa.

Se por um lado, essas técnicas tornam os movimentos mais realistas, por outro, eles também ficam bastante restritos à personagens com aspecto humano ou animal. Em nosso projeto, nós não restringimos as rotações nas juntas, para que o animador tenha liberdade de criar animações não-realistas.

Para gerar uma pose, o que precisamos, preliminarmente, é definir o modelo que será adotado. Dentre alguns tipos de modelos humanos existentes, há o modelo geométrico e o modelo articulado. Ainda que, na computação gráfica, modelos geométricos em malhas poligonais sejam muito populares para representar objetos, optamos por utilizar um simples modelo hierárquico articulado 3D, em conformidade ao método adotado. Tal modelo é constituído por uma aproximação bastante rudimentar de um esqueleto humano, como o ilustrado na Figura 5.1. Nele, as partes rígidas, que representam os ossos, são conectadas por articulações ou juntas, formando uma estrutura hierárquica que permite a movimentação relativa entre seus componentes. Não obstante, pode-se utilizar qualquer modelo articulado 3D nesses moldes, sem prejuízos ao método.

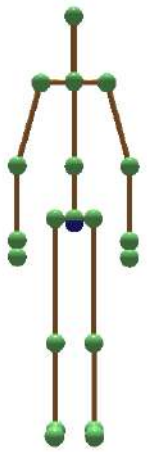


Figura 5.1: Modelo humano hierárquico articulado. Os ossos são representados por cilindros e as juntas por esferas.

Com o propósito de reduzir o esforço do animador, deve-se gerar um número mínimo de quadros-chaves. Assim, o ideal é que cada pose-chave contenha movimentos extremos da ação ou do movimento a ser realizado em uma animação. Por exemplo, ao animar uma passada de um humano caminhando, devemos considerar poses tais como *i*) a perna direita está fazendo contato com o solo; *ii*) a perna que estava na parte de trás está passando a perna da frente e *iii*) a perna direita está na parte de trás e o contato com o solo é feito com a perna esquerda, conforme ilustrado na Figura 5.2.

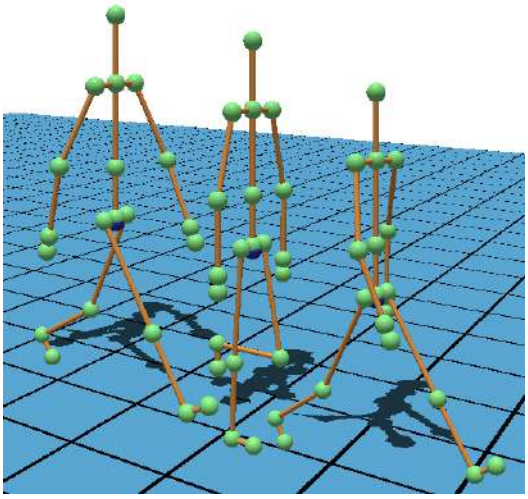


Figura 5.2: Poses-chaves com movimentos extremos de uma passada. A pose à esquerda mostra a perna direita fazendo contato com o solo. Na pose central, a perna esquerda está passando a perna direita e na pose à direita a perna direita está na parte de trás e o contato com o solo é feito com a perna esquerda.

5.1.2 Marcadores

Nesse sistema, cada marcador é associado a uma pose-chave. Ele é descrito por coordenadas no espaço tridimensional e visualizado na tela em forma de esferas representadas pela cor verde-escura, ou vermelha quando está selecionado (ver Figura 5.3). Essa disposição espacial, na maioria das vezes, é determinante para que o usuário possa alcançar os resultados almejados durante o curso da animação, além de facilitar a particularização de diversos estilos de movimentos e ações em uma mesma animação.

5.2 Controlador

O controlador é a ferramenta através da qual o usuário pode interagir com o ambiente virtual e, assim, construir uma sequência de animação. Ele também pode ser visualizado na tela em forma de esfera, representada nesse sistema pela cor amarela (ver Figura 5.3). A posição do controlador é um dos dados de entrada do algoritmo e é descrita por coordenadas no espaço tridimensional. Ela é usada para interpolar as diferentes poses.

Após configurar os quadros-chaves espaciais, é necessário mover o controlador no espaço, a cada nova posição, o sistema realiza uma interpolação entre os quadros-chaves em tempo real, como a vista na Subseção 4.3.1, para que se possa gerar novos quadros da sequência. Quanto mais próximo o controlador estiver de um marcador, mais parecido com a pose ao qual esse é associado a nova pose será.

A posição do controlador, originalmente, poderia ser alterada por dispositivos

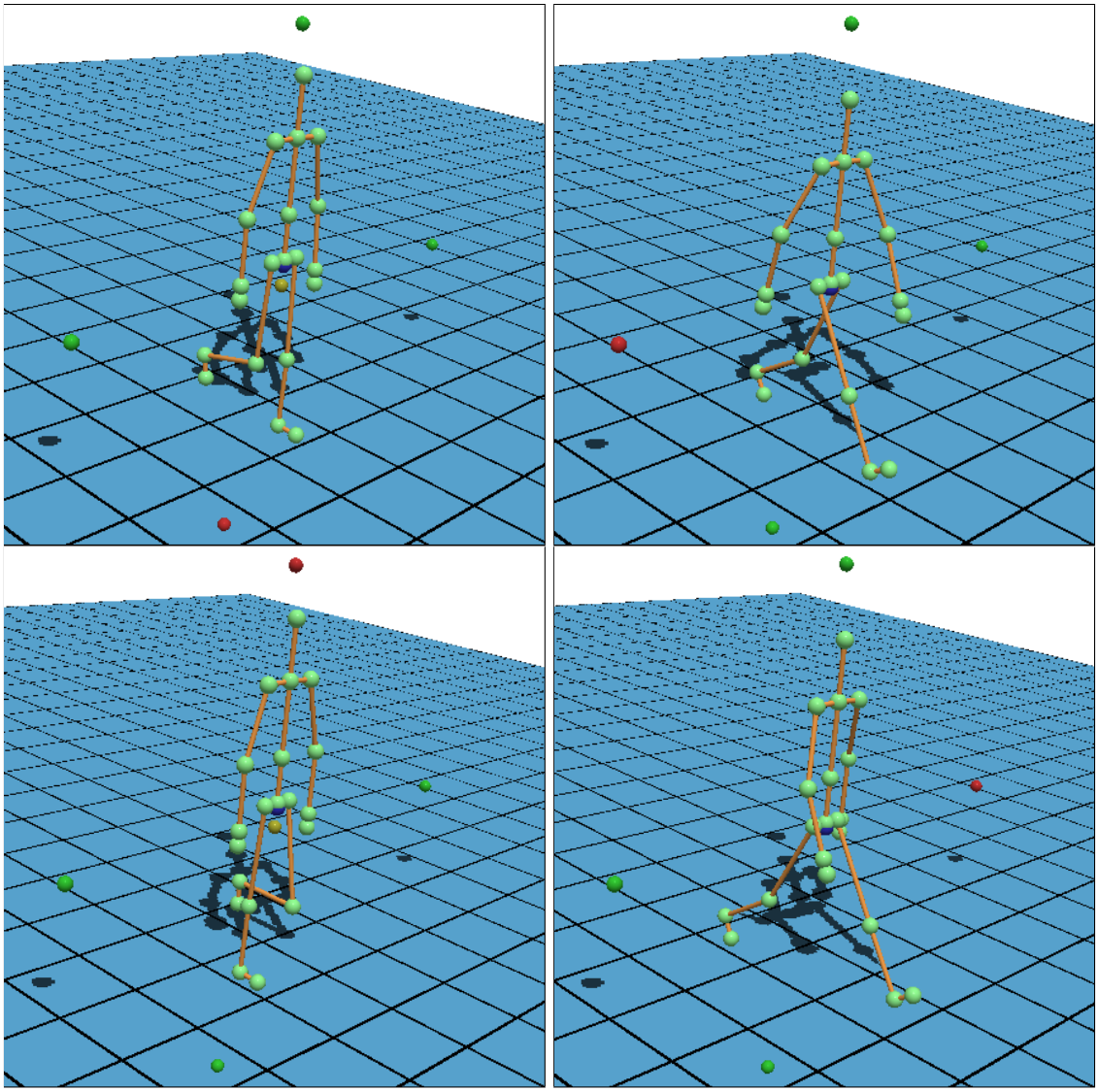


Figura 5.3: Quadros-chaves espaciais de um personagem articulado configurados para gerar uma animação de uma caminhada simples. As esferas verde-escuras representam os marcadores, a esfera vermelha identifica o marcador cujo quadro-chave está sendo retratado e a esfera amarela simboliza a posição do controlador.

simples e usuais em 2D como um mouse ou uma tela sensível ao toque. Visto que esses dispositivos não são ideais para manipular diretamente objetos no espaço tridimensional, propomos uma extensão dessa ideia para garantir uma total interação com o ambiente 3D, ao utilizar equipamentos mais atuais e sofisticados, como os sensores de profundidade. A escolha do dispositivo, é uma das etapas cruciais para favorecer o desempenho adequado do método. Existem diversos dispositivos apropriados para tal fim, como os apresentados na Subseção 1.1.2.

Nesse caso, a posição do controlador corresponde às coordenadas 3D da posição da mão do usuário capturadas pelo dispositivo, quando o mesmo realiza um determinado gesto. Diante disso, o sistema mapeia tal posição 3D na posição do controlador, em tempo real, até que o gesto seja finalizado. O gesto será detalhado na Subseção

6.1.1. A performance do usuário determinará, além do resultado final, a velocidade da animação, à medida que o mesmo movimenta a mão sobre a área destinada para tal ação.

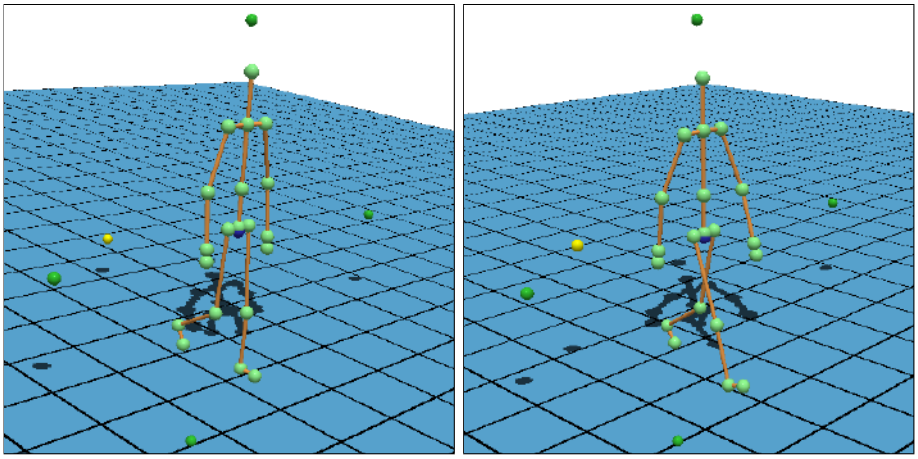
É importante salientar que, o traço capturado pelo sensor pode ser, muitas vezes, bastante ruidoso. Com isso, nós disponibilizamos como atributo em nosso sistema, uma opção para suavizar o traçado da curva produzida durante a performance do usuário, através de splines de aproximação.

De posse dos dados iniciais vistos até aqui – pose, marcador e controlador – já é possível gerar quadros-chaves espaciais e construir sequências de animações através da performance do usuário. No entanto, ainda que o sensor 3D transmita informações bastante precisas em tempo real e garanta a integridade das informações relativas à posição do dedo do usuário no espaço tridimensional, permitindo movimentos rápidos e lentos; posicionar a mão no espaço físico de modo que se possa localizar e interagir de forma intuitiva com os demais objetos na cena sem perder a noção de profundidade, não é uma tarefa trivial quando estamos visualizando, tão somente, a projeção da cena em uma tela 2D.

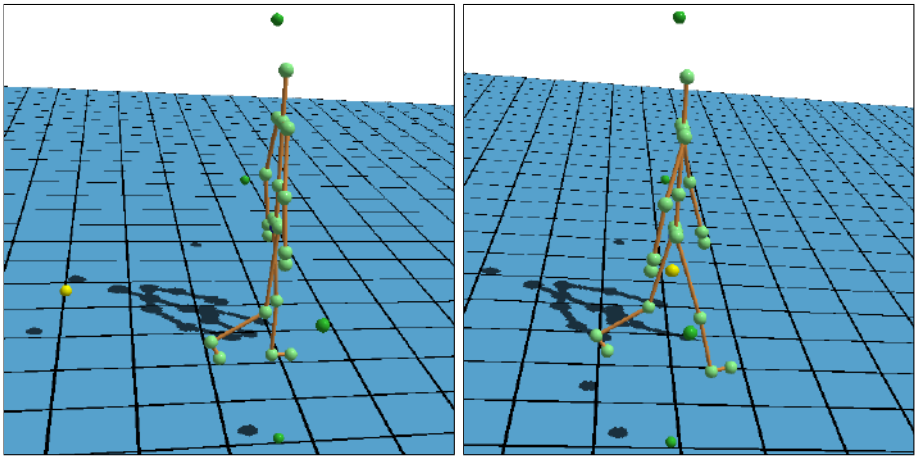
Em alguns casos, o animador pode estar interessado em posicionar o controlador em locais específicos no espaço, por exemplo, próximo de algum dos marcadores, como ilustrado na Figura 5.4. Observe na Figura 5.4a que, visualmente, os controladores representados pelas esferas amarelas, parecem bastante próximos de um dos marcadores, representados por esferas verde-escuras, e, apesar disso, geram quadros intermediários com uma diferença acentuada. Porém, quando mudamos o ponto de vista através de uma rotação, podemos observar que a posição do controlador não corresponde ao imaginado (ver Figura 5.4b), ou seja, na figura da esquerda ele está muito mais afastado dos marcadores do que na figura da direita.

Situações como essa podem fazer com que não se obtenha os resultados esperados através da performance do usuário em determinadas animações. Para solucionar problemas desse tipo, resolvemos vincular a disposição dos marcadores sobre superfícies no espaço 3D. O ideal é que a localização dos objetos no ambiente virtual seja rápida, simples e eficiente, para não prejudicar o resultado da animação durante a performance do usuário. A ideia de adotar uma superfície para apoiar os marcadores tem, justamente, o intuito de facilitar e agilizar a localização espacial dos mesmos na cena.

Mas como podemos escolher uma superfície de forma que possamos associá-la a um gesto manual natural? Antes de pensar na superfície, vamos iniciar a nossa construção pensando em curvas que podem ser associadas a esses gestos e que, de alguma forma, possam estar contidas na superfície. Na seção seguinte, descreveremos como determinamos essas superfícies.



(a) Quadros intermediários com posições diferentes do controlador.



(b) Os mesmos quadros acima, respectivamente, em diferentes pontos de vista.

Figura 5.4: Quadros intermediários gerados ao mover o controlador (esfera amarela) no espaço. Nas figuras superiores o controlador aparenta estar bem próximo de um dos marcadores (esferas verde-escuras). Nas figuras inferiores, ao mudar o ponto de vista da cena, observa-se que há uma grande diferença em relação à profundidade.

5.3 Superfícies

Partindo da ideia de associar gestos e movimentos com curvas sobre a superfície adotada para apoiar os marcadores, podemos, inicialmente, preconceber os movimentos que gostaríamos de executar e, por fim, encontrar as equações paramétricas de uma superfície que possa satisfazer as nossas reais necessidades.

Em uma configuração ideal, tal superfície deveria ser limitada, parametrizável e orientável, sendo ela simples e fácil de manipular computacionalmente. Para escolhermos superfícies adequadas para serem empregadas na construção do nosso método, analisamos algumas superfícies – regulares, de revolução e cilindrícas – cujas definições podem ser encontradas na Seção 3.2 do Capítulo 3.

A princípio, optamos por utilizar superfícies regulares em $\mathbb{R}^3$, pois elas não possuem arestas, autointerseções ou pontos singulares. Elas também possuem os vetores

tangentes linearmente independentes, com isso, pode-se facilmente calcular o vetor normal à superfície em cada ponto amostrado, o que é imprescindível para que possamos visualizar tal superfície tridimensional projetada em uma tela 2D, utilizando bibliotecas gráficas como a *OpenGL* [52].

Para facilitar ainda mais essa escolha e torná-la simples e imediata, podemos eleger superfícies que possuam uma parametrização padrão conhecida e que contenham curvas coordenadas semelhantes à forma do traçado descrito pelos gestos realizados para controlar a animação. Nas subseções seguintes, descreveremos algumas dessas superfícies:

5.3.1 Esferícas, cilíndricas e de revolução

Um movimento bastante espontâneo e simples de ser realizado com as mãos é o movimento circular. Ao realizar tal movimento, a ideia que ele preconiza seria de uma animação cíclica, como os passos em uma caminhada, por exemplo. Nesse caso, o ideal seria adotar uma superfície que fosse composta por curvas circulares. Para facilitar e controlar ainda mais a disposição espacial dos marcadores sobre a mesma, o ideal seria que, exatamente, curvas coordenadas dessa superfície tivessem o formato circular. Existem diversas superfícies nessas condições, como a esfera e o cilindro circular (ver os respectivos Exemplos 3 e 4 do Capítulo 3), que são superfícies simples e conhecidamente descritas na forma paramétrica.

A vantagem de escolher superfícies como cilindros circulares e até mesmo esferas para apoiar os marcadores é que, além delas serem simples e popularmente descrita na forma paramétrica, elas ainda possuem curvas coordenadas em forma de circunferência no espaço, embora no cilindro, apenas quando há interseção com planos perpendiculares ao eixo. Então, podemos facilmente associá-las com gestos circulares. Outras superfícies que poderiam ser empregadas para tal fim, seriam as superfícies de revolução, descritas na Subseção 3.2.2, pois elas também possuem curvas coordenadas circulares.

Porém, não necessariamente, precisamos ficar restritos aos movimentos circulares, até porque, executá-lo com perfeição não é uma tarefa das mais simples. Na prática, gestos circulares formam uma curva cuja trajetória se assemelha ao formato de uma elipse. Com isso, tornar-se-ia necessário relacioná-los a superfícies que permitissem dispor os marcadores no espaço de forma que os movimentos elípticos pudessem ser ajustados a curvas sobre tais superfícies.

Existem algumas superfícies simples que são capazes de satisfazer as condições expostas acima, uma delas é o *cilindro elíptico*, como vimos na Subseção 3.2.1. Assim como o cilindro circular, o cilindro elíptico com eixo ao longo do eixo z

também possui uma parametrização simples e extremamente conhecida, mas o que os diferenciam são, justamente, as curvas coordenadas quando o parâmetro v é constante (ver Figura 3.4). Estes com curvas em formato de elipses (ver Figura 3.4b) e aqueles com curvas circulares (ver Figura 3.4a), obtidas quando realiza-se a interseção da respectiva superfície com planos paralelos ao plano xy, ao longo do eixo z.

As superfícies apresentadas até aqui, nitidamente satisfazem a condição inicial de utilizar superfícies cujas curvas coordenadas possam ser associadas a gestos naturais, executados pelo usuário, para realizar determinadas animações com movimentos cíclicos. No entanto, queremos elaborar animações mais complexas, envolvendo diversos estilos de movimento de uma mesma ação, ou ainda combinar diversos tipos de ações, por exemplo corrida e salto, em uma mesma animação. Nesse caso, talvez não seja apropriado utilizar tais superfícies, pois a relação entre elas e os movimentos ou os gestos executados pelo usuário pode não ser tão intuitiva e direta. Para resolver isso, podemos construir uma superfície, cujo modelo possa ser adaptável ao gesto e ao estilo de movimento, ou ação a serem animados.

Conhecendo as curvas desejadas e fazendo com que elas coincidam com curvas de contorno na superfície que será utilizada para apoiar os marcadores, podemos estabelecer outras superfícies regulares, mesmo que tais superfícies não tenham uma descrição simples e conhecida na forma paramétrica, como as que vimos até então. Não obstante, tomando como premissa as parametrizações padrões de superfícies como as cilíndricas e de revolução, vistas no Capítulo 3 e fazendo alguns ajustes, estabelecemos novas superfícies para apoiar marcadores no espaço 3D, como no caso que veremos na subseção a seguir:

5.3.2 Superfície Lins

Tomando por base o cilindro circular, podemos compor uma outra superfície, mesmo que ela não possua curvas coordenadas em formato de circuferência. Mais especificamente, se desejarmos que algumas curvas coordenadas dessa nova superfície tenham o formato de elipses, as quais teriam um dos eixos acrescidos ou reduzidos de um determinado valor, para cada valor de v, obteríamos, assim, uma outra superfície regular que não é cilíndrica, mas que também possui curvas coordenadas em formato de elipses, quando o parâmetro v é constante.

Como visto na definição 13, ao tomar uma curva geratriz no plano xz, pode-se gerar uma superfície de revolução ao rotacionar tal curva em torno do eixo Oz. E, mesmo que a superfície a ser gerada não seja uma superfície de revolução, ainda podemos admitir essa ideia e eleger uma curva "geratriz" para a nossa superfície.

Nesse caso, a curva não será rotacionada em torno do eixo Oz, mas, para um determinado valor de u, ela deverá conter os vértices do semieixo das elipses, ou seja, para cada v no domínio, os vértices do eixo variante das elipses estariam apoiados sobre tal curva. A Figura 5.5 mostra uma curva plana regular utilizada para esse fim. Essa curva é descrita pela seguinte equação:

$$f(v) = e^{\,\mathrm{sen}(\frac{10-v}{3})} - e^{\,\mathrm{sen}(\frac{10}{3})}; \quad v \in \mathbb{R} \tag{5.1}$$

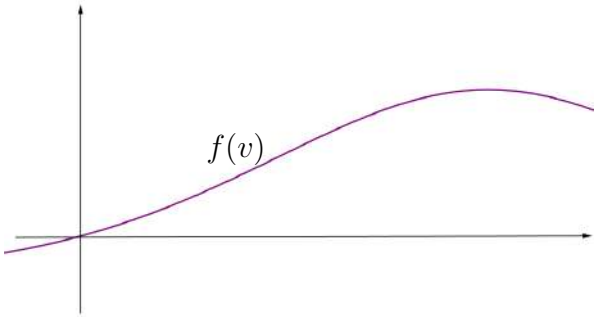


Figura 5.5: Curva plana regular dada pela equação $f(v) = e^{\,\mathrm{sen}(\frac{10-v}{3})} - e^{\,\mathrm{sen}(\frac{10}{3})}$.

Já que tomamos por referência o cilindro circular descrito parametricamente como na Equação 3.14, tendo como curva diretriz um círculo de raio r como o do Exemplo 1 e eixo ao longo do eixo z; para descrever as elipses, devemos eleger um dos eixos – x ou y – para ser o eixo que será modificado. Se quisermos variar o eixo da elipse na direção do eixo y, basta adicionar o valor correspondente a $f(v)$ ao raio r nessa coordenada. Então, podemos compor as equações paramétricas dessa nova superfície da seguinte forma:

$$\begin{cases} x(u,v) = r\cos(u) \\ y(u,v) = (r + (e^{\,\mathrm{sen}(\frac{10-v}{3})} - e^{\,\mathrm{sen}(\frac{10}{3})}))\,\mathrm{sen}(u) \\ z(u,v) = v \end{cases} \tag{5.2}$$

com $r, u, v \in \mathbb{R}$, tal que $0 \leq u < 2\pi$. Aqui, r representa o comprimento de um dos vértices das elipses.

A Figura 5.6 exibe a superfície sob diferentes pontos de vista.

Observe que, quando $v = 0$ teremos

$$y(u,0) = (r + (e^{\,\mathrm{sen}\left(\frac{10-0}{3}\right)} - e^{\,\mathrm{sen}\left(\frac{10}{3}\right)}))\,\mathrm{sen}(u) = r\,\mathrm{sen}(u).$$

Isso nos dá

$$\begin{cases} x(u,0) = r\cos(u) \\ y(u,0) = r\,\mathrm{sen}(u) \\ z(u,0) = 0 \end{cases} \tag{5.3}$$

Assim, obtemos uma curva coordenada em formato de circunferência e po-

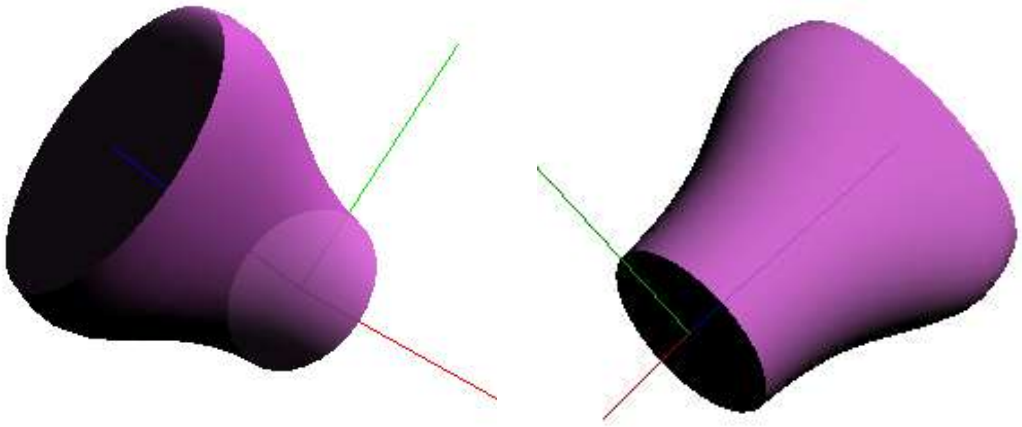


Figura 5.6: Diferentes visualizações da superfície Lins.

deríamos adaptar mais de um tipo de gesto – circular e elíptico – sobre a mesma superfície. O que poderia nos dar a liberdade de associá-los a diferentes estilos de movimentos ou ações em uma mesma animação.

5.3.3 União de superfícies

Além de utilizar individualmente as superfícies descritas acima, ainda há a possibilidade de criar animações utilizando mais de uma delas ao mesmo tempo. Para gerar uma única animação que tenha ações diferentes, como ciclos de corrida e salto independentes, podemos posicionar os conjuntos de marcadores de cada ação específica em superfícies distintas, que estejam interligadas suavemente. Nesse caso, poderíamos utilizar um cilindro para posicionar os marcadores com os quadro-chaves que possibilitam que se execute o movimento de corrida; já os marcadores do salto poderiam ser associados a uma superfície como a descrita na Subseção 5.3.2, pois, a amplitude do gesto associado à superfície, ilustraria de forma natural e suave a diferença entre os movimentos.

Unir as superfícies de forma que se realize um encaixe perfeito também é desejável. O ideal é que não se tenha falhas ou buracos após a junção, pois assim, a transição entre elas, ao mover o controlador por exemplo, pode ser feita de forma contínua e suave. As superfícies vistas até aqui podem facilmente serem conectadas sem que ocorra falhas ou buracos. Como observado na Equação 5.3, além das elipses, a superfície Lins ainda tem uma curva circular, então ela pode ser conectada a um cilindro circular com um mesmo raio r, além do cilindro elíptico.

5.4 Transformação de coordenadas

Em nosso método, estamos lidando com diversas entidades – posição dos marcadores no espaço, superfícies, posição capturada pelo sensor de profundidade, além da própria representação do esqueleto – as quais devem ser visualizadas em uma mesma tela 2D. Cada uma delas, pode ter o seu próprio referencial em um sistema de coordenadas 3D, diante disso, é necessário determinar uma relação comum entre elas.

O sistema de coordenadas do sensor *Leap Motion* equipara-se ao sistema utilizado pela biblioteca *OpenGL*. Na Figura 5.7 podemos visualizar a relação entre o sistema de coordenadas cartesianas e esféricas adotado por GUNA *et al.* [25], o qual também aderimos em nosso sistema. Podemos observar, também, que a origem do sistema de coordenadas encontra-se sobre o centro do sensor. Com o objetivo de facilitar alguns cálculos, deslocamos essa origem para o centro do campo de visão do dispositivo.

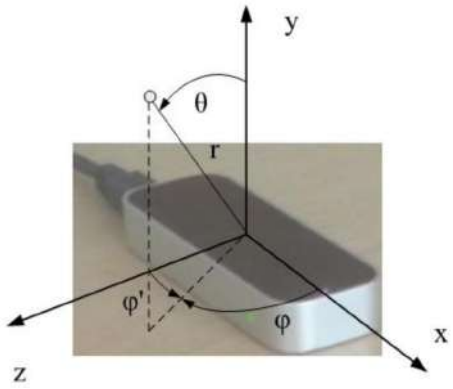


Figura 5.7: Relação entre as coordenadas cartesianas e esféricas do sensor *Leap Motion*. Figura retirada de [25].

5.4.1 Coordenadas de superfícies

Em alguns momentos, estamos interessados em realizar determinadas transformações dos objetos presentes na cena, como projetar e deslocar os marcadores sobre a superfície, ou projetar a posição do controlador também sobre a superfície, à medida que o mesmo desloca-se na cena. Isso pode garantir mais um auxílio visual e, também, um traço mais preciso, já que o movimento do usuário pode gerar traços bem ruídosos. Como consequência, pode-se produzir resultados mais fidedignos com as poses originais, tornando o resultado mais estável.

É importante destacar também, que para intensificar a acuidade visual diante do ambiente 3D, além de projetar a posição do controlador sobre a superfície, também exibimos uma curva coordenada dessa superfície na posição da projeção.

Quando estamos lidando com superfícies cilíndricas ou esféricas, dado um ponto

no espaço tridimensional podemos facilmente encontrar o seu equivalente em coordenadas de superfície, utilizando as relações entre coordenadas esféricas, cilíndricas e cartesianas, vistas nas Seção 3.3. Já para a superfície Lins encontramos uma solução diferente, como veremos a seguir.

Esfera

Para transformar um ponto com coordenadas cartesianas (x_0, y_0, z_0) em um ponto sobre uma esfera de raio R, com coordenadas (x_1, y_1, z_1), primeiro deve-se encontrar o ponto correspondente em coordenadas esféricas (r, θ, ϕ), utilizando a equação 3.16, para determinar os ângulos θ e ϕ. Depois, determinar (x_1, y_1, z_1) ao aplicar os valores de R, θ e ϕ correspondentes na equação 3.17.

Cilindro

Já no caso do cilindro circular reto de raio R e eixo ao longo do eixo z, uma das opções é encontrar o ângulo ϕ, o que pode ser feito ao encontrar o ponto correspondente em coordenadas cilíndricas (r, ϕ, z) através da equação 3.18. E, depois, aplicar os valores correspondentes na equação 3.19 para encontrar o ponto de coordenadas (x_1, y_1, z_1) sobre o cilindro.

Uma outra opção seria, dado um ponto com coordenadas cartesianas (x_0, y_0, z_0), encontrar o ponto correspondente com coordenadas (x_1, y_1) na curva de interseção da superfície com um plano paralelo ao plano xy, passando por z. No caso das superfícies esféricas e cilindrícas como as descritas nos parágrafos acima, teríamos curvas em forma de circunferências.

Superfície Lins

Podemos aplicar essa mesma ideia para determinar um ponto sobre a superfície Lins. Se tivermos um ponto P_0 com coordenadas (x_0, y_0, z_0) no sistema de coordenadas cartesianas, para determinarmos o ponto correspondente P_1 com coordenadas (x_1, y_1, z_1) sobre essa superfície, basta lembrarmos que, as curvas coordenadas quando o parâmetro v é constante possuem a forma de elipses paralelas ao plano xy, como vimos na Subseção 5.3.2. Daí, para cada valor de $z_0 = z_1$, o nosso esforço resume-se a encontrar as coordenadas correspondentes x_1 e y_1 de um ponto P_1 em uma elipse no plano xy. A Figura 5.8 ilustra esse fato. Diante disso, devemos determinar o ponto P_1 de interseção da elipse com a reta que passa pelo ponto P_0 e a origem O, no mesmo quadrante em que P_0 se encontra. Se considerarmos a equação da reta como

$$y = \frac{y_0}{x_0} x$$

e a equação da elipse sendo

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1,$$

então podemos utilizar a seguinte relação para determinar as coordenadas de P_1:

$$\begin{cases} x_1 = \dfrac{ab}{\sqrt{(ay_0)^2 + (bx_0)^2}} x_0, \\ y_1 = \dfrac{ab}{\sqrt{(ay_0)^2 + (bx_0)^2}} y_0, \\ z_1 = z_0. \end{cases} \tag{5.4}$$

Nesse caso, basta tomarmos $a \in \mathbb{R}$ e $b = a + \left(e^{\,\mathrm{sen}\left(\frac{10-z_0}{3}\right)} - e^{\,\mathrm{sen}\left(\frac{10}{3}\right)}\right)$ como vértices da elipse.

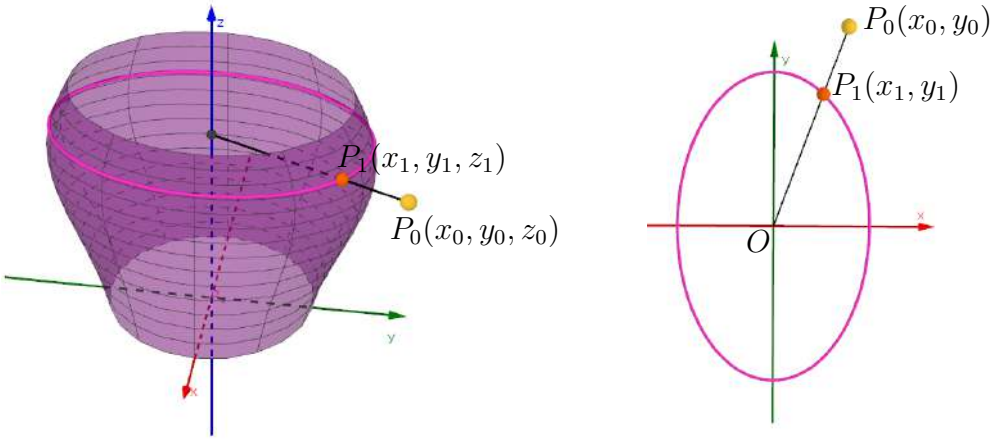


Figura 5.8: Ponto sobre a superfície Lins.

5.5 Interpolação

Até então, o nosso sistema está hábil para realizar, de forma satisfatória, animações simples como por exemplo, ciclos de caminhada com apenas um estilo de movimento. A interporlação para gerar animações simples foi criada utilizando o mesmo conceito proposto por IGARASHI *et al.*[11], visto na Seção 4.3.1. Ela utiliza uma RBF e funciona adequadamente nesse tipo de animação. No entanto, gostaríamos de executar animações mais elaboradas em tempo real, com diversos tipos de ações ou estilos de movimento em uma mesma animação, sem que houvesse a necessidade de executar etapas posteriores para juntar essas ações de forma suave, em uma mesma animação.

Para gerar essas animações complexas de forma mais robusta e com transição suave entre as ações, podemos utilizar as seguintes soluções:

- *Interpolação global* - utilizar todos os marcadores de todos os subconjuntos de marcadores presentes na cena para obter uma nova pose;

- *Interpolação local* - apenas os marcadores presentes em um subconjunto de marcadores que correspondem a uma ação ou estilo de movimento específico e que estão dispostos sobre uma mesma região ou superfície serão aplicados na interpolação.

- *Interpolação especial* - realizada quando estiver em uma região de transição entre ações diferentes. A posição do controlador é projetada em cada região adjacente, gerando novas poses em cada posição projetada. Essas poses são interpoladas linearmente usando as distâncias de projeção como pesos, resultando em uma nova pose da sequência.

A posição do controlador irá definir qual é a interpolação a ser adotada em cada momento. Para determinar se será utilizada uma interpolação global, local, ou especial, é necessário determinar sobre qual região encontra-se se o ponto correspondente à posição do controlador. Se o mesmo está na região de uma superfície que contém um conjunto de marcadores, a interpolação poderá ser global ou local. Porém se ele está na região de transição, será feita uma interpolação especial.

Quando a animação possui estilos diferente de um mesmo movimento, por exemplo andar triste, alegre ou normal, basta dispor subconjuntos de quadros-chaves de estilos diferentes espaçados sobre uma mesma superfície. A Figura 5.9 ilustra esse caso. Assim, é possível utilizar apenas uma interpolação global e a semelhança entre diversas poses presentes nos subconjuntos garante a suavidade do movimento.

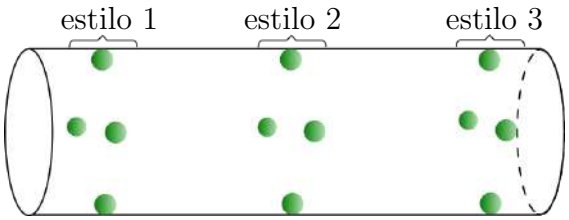


Figura 5.9: Superfície apoiando subconjuntos de marcadores de poses-chaves configuradas para realizar uma animação de um movimento com três estilos diferentes.

Já para animações com mais de um tipo de ação, como por exemplo, corrida e salto, é possível realizar configurações adicionais que permitam tanto utilizar interpolações locais, quanto uma interpolação especial. Sendo assim, podemos restringir o subconjunto dos marcadores de cada ação, sobre uma determinada superfície, ou região de uma mesma superfície, como ilustra a Figura 5.10, e realizar interpolações locais em cada subconjunto de marcadores que corresponde a uma ação específica.

Quando o controlador encontra-se na região de transição dessas ações, o mesmo é projetado em cada um dos planos que separam as regiões adjacentes, produzindo dois novos marcadores, cada um associado a uma pose resultante de uma respectiva interpolação local, em cada região adjacente. Por fim, realiza-se uma interpolação linear utilizando a posição do controlador e dos dois novos marcadores.

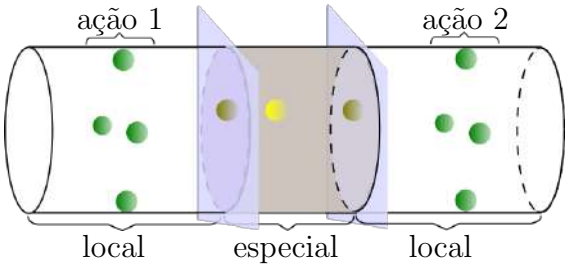


Figura 5.10: Superfície dividida em três regiões de interpolação. Cada região de interpolação local, contém subconjuntos de marcadores (verde) de poses-chaves configuradas para realizar uma determinada ação. Na região de interpolação especial, a posição do controlador (amarelo) é projetada nas regiões adjacentes.

No entanto, não necessariamente essas são formas exclusivas, as superfícies e interpolações podem ser ajustadas como desejado dentre as opções disponíveis no sistema, ficando o animador, livre para escolher a configuração que mais o agradar. Por exemplo, quando uma animação possui diversos estilos de animação, poderia realizar interpolações locais e especiais, ao invés de global. E quando há diversas ações na mesma animação, poderia empregar apenas uma interpolação global, com os marcadores sobre uma mesma superfície.

5.5.1 Marcadores especiais

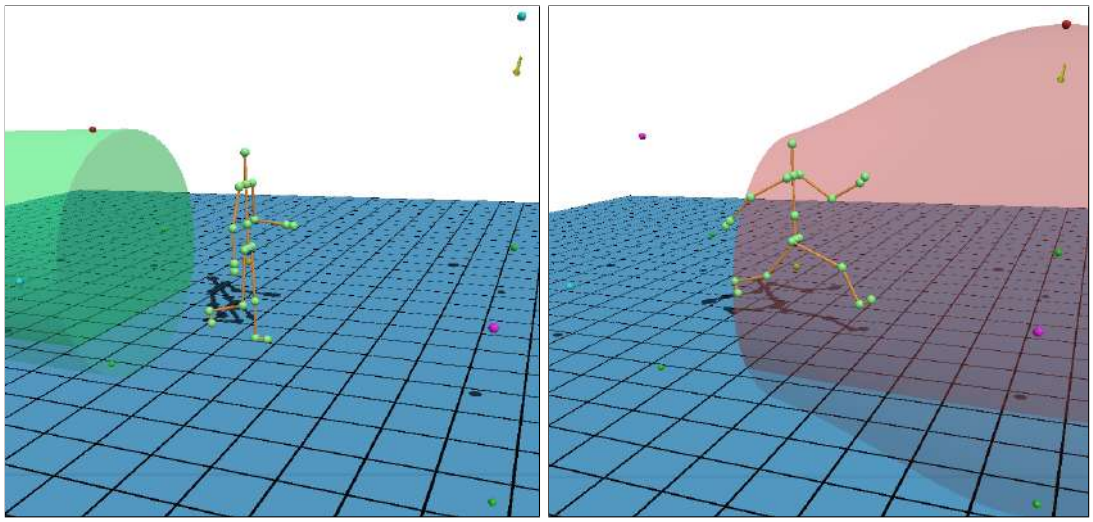
Em alguns casos, apenas realizar uma interpolação especial na região de transição não é suficiente para que a transição entre as diferentes ações, ao realizar animações compostas, seja feita de forma suave. Com isso, marcadores especiais foram criados devido à necessidade de solucionar problemas como esse.

Em cada conjunto de marcadores associados a um estilo de animação, apoiados sobre uma determinada superfície, elegemos um marcador de entrada na animação e um marcador de saída. É importante salientar, que as poses de entrada e saída devem ter alguma relação de proximidade, para que o movimento transcorra de forma suave. Ou seja, a pose de saída de uma região deve ter um movimento correlato com a pose de entrada na região adjacente.

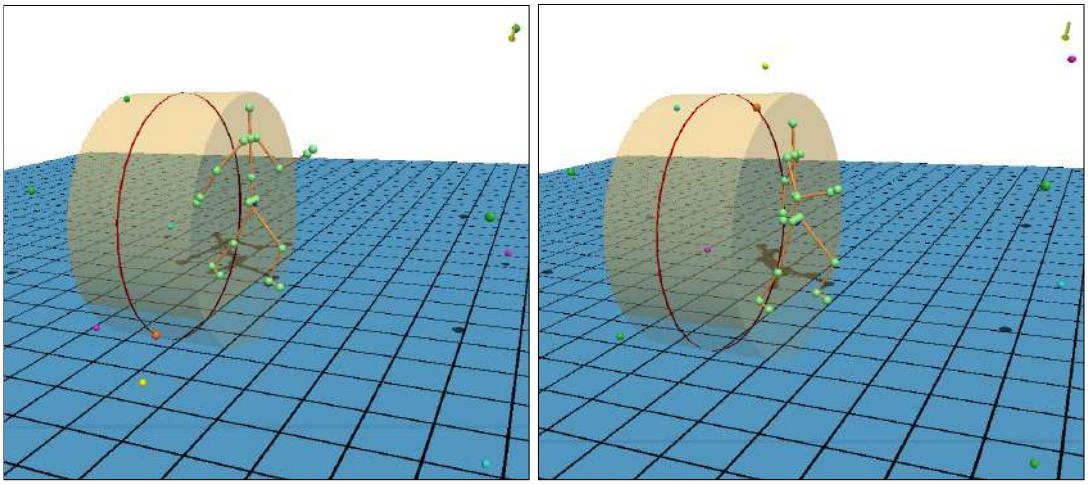
Dessa forma, em alguns momentos deve-se realizar rotações da posição dos marcadores sobre a superfície, para que a posição do marcador que representa o quadro-chave que se deseja iniciar o ciclo de movimento fique mais próxima da posição do controlador. Esse alinhamento acontece sempre que o controlador está posicionado

na região limitada da superfície que não contém marcadores associados, ou seja, quando ele sai do espaço definido para uma superfície que contém marcadores e entra no espaço de uma superfície de transição. Ao mover o controlador dentro desse espaço, os marcadores associados às superfícies vizinhas rotacionam, até que se saia do espaço de transição e entre no espaço de uma superfície que tenha marcadores associados.

A Figura 5.11 ilustra essa ideia. Nela, podemos observar a configuração inicial dos marcadores de cada ação sobre uma determinada superfície (5.11a) e o resultado de algumas rotações quando o controlador está na região de transição (5.11b). Os marcadores de entrada são simbolizados por esferas de cor turquesa e os de saída da animação pela cor rosa. Na Figura 5.11b à esquerda, o controlador está saindo da região da superfície à esquerda e entrando na região da superfície à direita, com isso, os respectivos marcadores de saída e entrada dessas regiões, estão alinhados com o controlador. Na Figura 5.11b à direita ocorre o oposto, o controlador está saindo da região à direita e entrando na região à esquerda.



(a) Configuração inicial.



(b) Marcadores rotacionados.

Figura 5.11: Configuração de marcadores de entrada (esferas turquesas) e de saída (esferas rosas) de cada região correspondente a uma ação. Em (a) temos a configuração inicial e em (b) após a rotação dos marcadores.

Capítulo 6

Resultados

Nesse capítulo apresentaremos alguns detalhes da concepção do método, a interface gráfica desenvolvida, os resultados através de algumas animações e uma reflexão relativa às vantagens e as limitações do sistema proposto.

Nós implementamos as nossas ideias utilizando a linguagem de programação C++, as bibliotecas Opengl [52] e Tucano [57], em um computador com um processador Intel® Core™ i7, CPU 930 @ 2.80GHz × 8, com 16 GiB de memória RAM e placa gráfica NVidia GeForce GTX 460/PCIe/SSE2.

Para validar o nosso método nós desenvolvemos uma interface, que além de um recurso gráfico, possui um recurso gestual. Mais detalhes da interface serão vistos na Seção 6.1. O gesto e o dispositivo escolhido serão explorados na Subseção 6.1.1 e a parte gráfica será especificada na Subseção 6.1.2.

Por fim, testamos o nosso sistema gerando animações desde as mais simples, como um ciclo de caminhada, até algumas mais elaboradas, com movimentos e ações diferentes em uma mesma animação. As animações serão exibidas na Seção 6.2.

6.1 Interface

Como vimos no Capítulo 5, o nosso método propõe uma forma de criar animações usando a técnica de quadros-chaves espaciais, através da performance do usuário, o qual terá os gestos e movimentos das mãos capturados por um sensor de profundidade.

Com isso, desenvolvemos uma interface que dispõe de um recurso gestual, para captura de movimentos da mão, cujos detalhes serão exibidos na Subseção 6.1.1; além de um recurso gráfico, dotado de botões que podem ser gerenciados através do mouse e conferem determinadas características adicionais ao sistema, como gravar uma animação que está sendo processada em tempo real, para revê-la posterior-

mente, ou suavizar o traçado da curva, ou ainda projetar a posição do controlador sobre a superfície, quando se desejar executar traçados mais precisos para aproximar-se das poses originais. Exibiremos mais detalhes desses recursos na Subseção 6.1.2.

6.1.1 Recurso gestual

Vimos na Seção 5.2 que a posição do controlador corresponde às coordenadas 3D da mão do usuário. Quando ele executa um determinado gesto e a movimenta, as coordenadas são capturadas por um sensor de profundidade.

A princípio, optamos por utilizar o *Kinect* em sua versão original, já que, como especificado no trabalho [58], a nossa abordagem inicial era um tanto quanto diferente do método aqui apresentado. Por fim, com a intenção de capturar gestos e movimentos das mãos com confiança, adotamos o dispositivo *Leap Motion*, pois ele confere precisão e robustez adequada para um melhor desempenho do nosso sistema. Ainda que, como afirmado por SHARP *et al.* [59], o *Kinect* em sua nova versão para *XBox One* tenha se mostrado mais eficiente que outros dispositivos disponíveis para o rastreamento de mãos, como o próprio *Leap Motion*, não chegamos a realizar nenhum teste com tal dispositivo na atual versão do trabalho, pois os fundamentos que aqui desenvolvemos podem, facilmente, ser adaptados ao uso com qualquer um dos dispositivos que tenham a mesma funcionalidade tal qual o dispositivo que adotamos.

Depois de escolher o dispositivo, fez-se necessário definir um gesto que permitisse interagir com o sistema de animação. A nossa escolha foi implementar o gesto de apontar o dedo indicador, como ilustrado na Figura 6.1. Para realizá-lo, deve-se considerar as fases necessárias para que um gesto possa ser executado, como a preparação, o início e o final do gesto. Primeiramente, na fase de preparação, deve-se posicionar a mão no espaço destinado ao campo de visão do dispositivo, para que o mesmo a identifique automaticamente. Então, determinamos que o início do gesto ocorre quando somente o dedo indicador estiver apontando, o que pode ser facilmente verificado através do SDK do *Leap*, o qual indica quando um dedo está ou não estendido. Já o final do gesto, pode ocorrer em decorrência das seguintes circunstâncias: *i*) quando o dedo indicador não estiver mais extendido; *ii*) quando algum dos outros dedos, além do indicador, também estiver extendido; ou *iii*) quando a mão sair do campo de visão do sensor.

A captura da posição do dedo é iniciada após o usuário realizar o gesto, na área destinada para captura de gestos e movimentos, dentro do campo de visão do dispositivo e persiste até que o mesmo seja finalizado. O sensor captura a posição 3D do dedo indicador do usuário e transmite tal informação ao sistema em tempo real. Diante disso, o sistema mapeia os dados recebidos na posição do controlador.

Figura 6.1: Gesto de apontar o dedo indicador no campo de visão do sensor.

6.1.2 Recurso gráfico

A nossa interface, no entanto, não está restrita ao uso de gestos, pois ela também é composta por uma parte gráfica, a qual dispõe de botões que podem ser acessados através do mouse, conforme ilustrado na Figura 6.2. Os botões encontram-se do lado esquerdo da figura e conferem características adicionais ao nosso sistema.

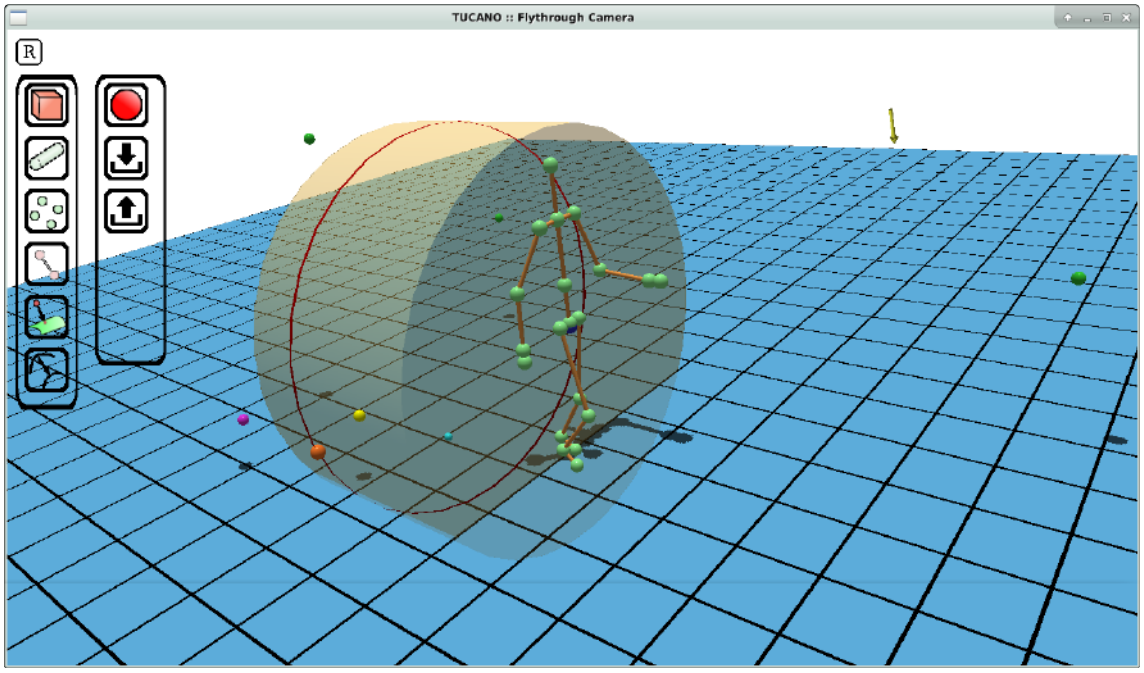


Figura 6.2: Interface desenvolvida para experimentar o método proposto e apresentar os resultados obtidos. Na parte esquerda da figura, podemos observar os botões que permitem executar comandos com recursos adicionais.

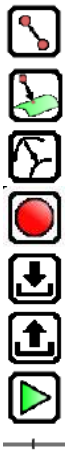
Cada um dos botões presentes na interface será caracterizado a seguir:

 - *Reload*;

 - Exibir/ocultar a área de interação;

 - Exibir/ocultar a superfície;

 - Exibir/ocultar os marcadores;

 - Exibir/ocultar o controlador, bem como a sua projeção sobre a superfície;

- Fixar o controlador sobre a superfície;

- Suavizar o traçado da curva produzida pela performance do usuário;

- Gravar uma animação resultante da performance do usuário;

- Gerar um arquivo de texto com a animação gravada;

- Ler um arquivo de texto com uma animação gravada;

- Executar uma animação gravada;

- Controlar a velocidade ao executar uma animação gravada.

Esses dois últimos botões ficam acessíveis apenas quando o sistema ler um arquivo de texto com uma performance gravada.

Outras configurações, tais como posição da luz, rotação e translação da câmera, dentre outras, podem ser alteradas através do mouse.

Para darmos início aos nossos testes, precisamos configurar as poses-chaves, o que pode ser feito posicionando ossos e/ou juntas no espaço tridimensional, em locais definidos pelo usuário, com o auxílio de um mouse.

Utilizar softwares como *3ds Max, Soft Image* e *Maya* da *Autodesk* [60] ou *Blender* [61], que permitem a construção e edição de poses-chaves de personagens articulados 3D, para auxiliar na construção das poses-chaves com as restrições necessárias para o adequado funcionamento do método também é uma opção viável. No nosso sistema, permitimos a importação do *Blender* a partir de um *plugin* especialmente desenvolvido.

Com o sistema minimamente configurado pelos dispositivos físicos necessários e os quadros-chaves espaciais, pôde-se realizar alguns testes, cujos resultados serão vistos na Seção 6.2.

6.2 Animações

Ao longo dessa seção, exibiremos as configurações e os resultados das animações que foram produzidas utilizando o nosso sistema.

6.2.1 Caminhada

Iniciamos os nossos testes construindo uma animações de uma caminhada simples. Em se tratando de animação de esqueletos humanos, a caminhada é um movimento primário e simples de ser executado. Nesse caso, empregamos um cilindro circular reto para apoiar os marcadores, por se tratar de um movimento cíclico e utilizamos os quadro-chaves retratados na Figura 6.3. Nela podemos observar as poses-chaves

e seus respectivos marcadores associados, representados pelas esferas vermelhas em cada quadro.

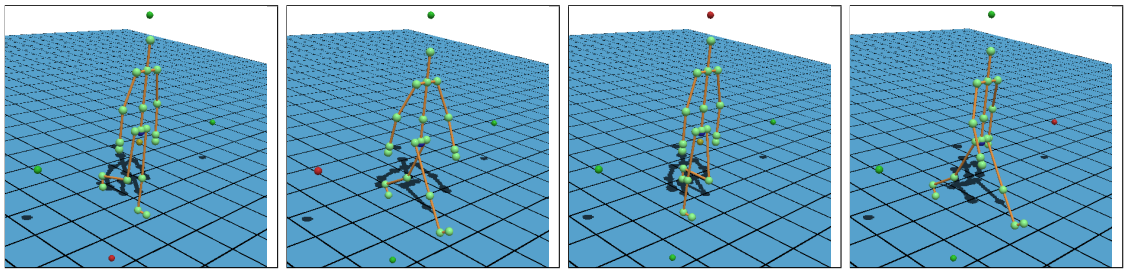


Figura 6.3: Quadros-chaves utilizados para gerar a animação de uma caminhada simples.

A Figura 6.4 exibe alguns resultados com novas poses geradas através desse sistema de animação. À direita, podemos observar os marcadores apoiados sobre um cilindro, bem como o controlador (esfera amarela) e a sua projeção sobre a superfície (esfera vermelha). Para intensificar a acuidade visual diante do ambiente 3D, também exibimos uma curva coordenada da superfície na posição da projeção do controlador.

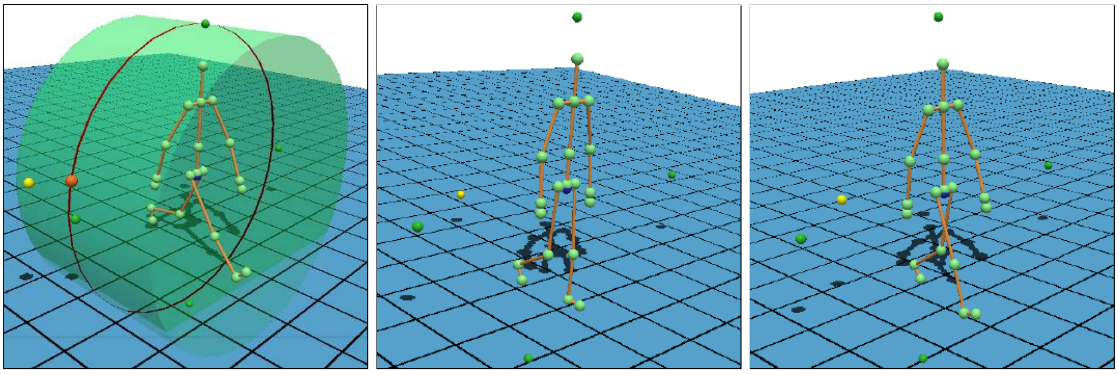
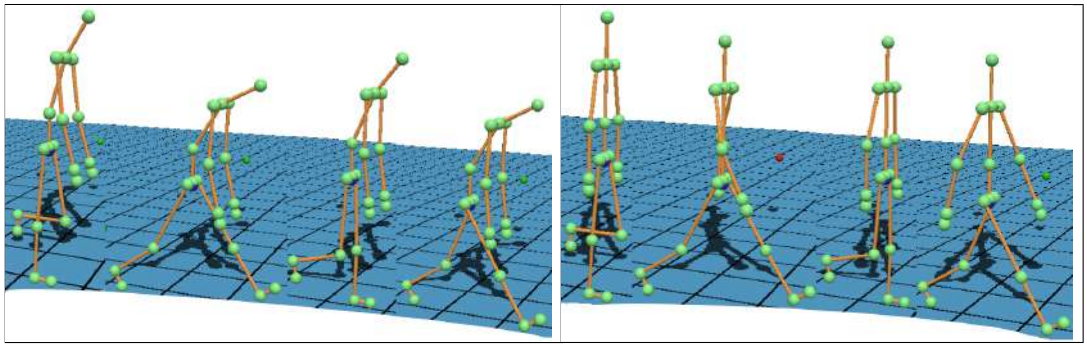


Figura 6.4: Novas poses geradas através da interpolação das poses-chaves de um ciclo de caminhada.

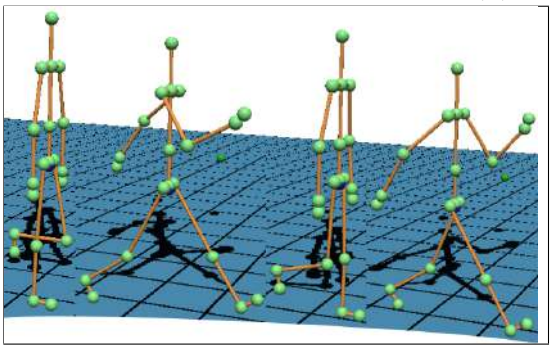
6.2.2 Caminhada com estilos diferentes

Dando continuidade aos nossos testes, nós elaboramos uma animação de uma ação com mais de um estilo de movimento. Nesse exemplo foi construída uma animação de uma caminhada com os seguintes estilos: triste, normal e alegre. Os subconjuntos de poses-chaves referente a cada um desses estilos podem ser visualizados na Figura 6.5. Foi utilizado um cilindro circular reto para apoiar os marcadores. Como vimos na Seção 5.5, nesse caso é possível utilizar tanto uma interpolação global ou subdividir a superfície em regiões de interpolação local ou especial.

A Figura 6.6 exibe diversas poses novas geradas pelo sistema, com os diferentes estilos, em uma sequência da esquerda para a direita e de cima para baixo.



(a) Triste. (b) Normal.



(c) Feliz.

Figura 6.5: Quadros-chaves utilizados para gerar a animação de uma caminhada com três estilos diferentes de movimento: (a) triste, (b) normal e (c) feliz.

6.2.3 Corrida e Salto

Por conseguinte, nós experimentamos misturar dois movimentos com ações diferentes em uma mesma animação. Para tal, nós adotamos os movimentos de corrida e salto ao realizar o nosso teste. As poses-chaves para cada uma das ações podem ser vistas na Figura 6.8, em 6.8a as poses da corrida e em 6.8b, as poses correspondentes ao salto. Cada subconjunto de quadros-chaves referentes a uma das ações foi associado com uma superfície diferente, com uma região de integração entre elas, conforme ilustrado na Figura 6.7. No caso da corrida, utilizamos um cilindro, pois assim como a caminhada, também é um movimento cíclico (ver Figura 6.7 à esquerda). Já para o salto, utilizamos a superfície Lins, iniciando com o mesmo raio do cilindro, para que não houvesse falhas quando as unisse (ver Figura 6.7 à direita).

Nas superfícies, utilizamos a interpolação local e na região de integração, uma interpolação especial e como feedback visual, temos um cilindro de cor marrom, para representar a região de transição (ver Figura 6.7 central). Nesse caso, temos um diferencial de eleger poses de entrada e saída de cada uma das superfícies para garantir a suavidade do movimento, conforme pode ser visto na figura 5.11.

Algumas poses resultantes podem ser visualizadas na Figura 6.9. A sequência está ordenada da esquerda para a direita e de cima para baixo.

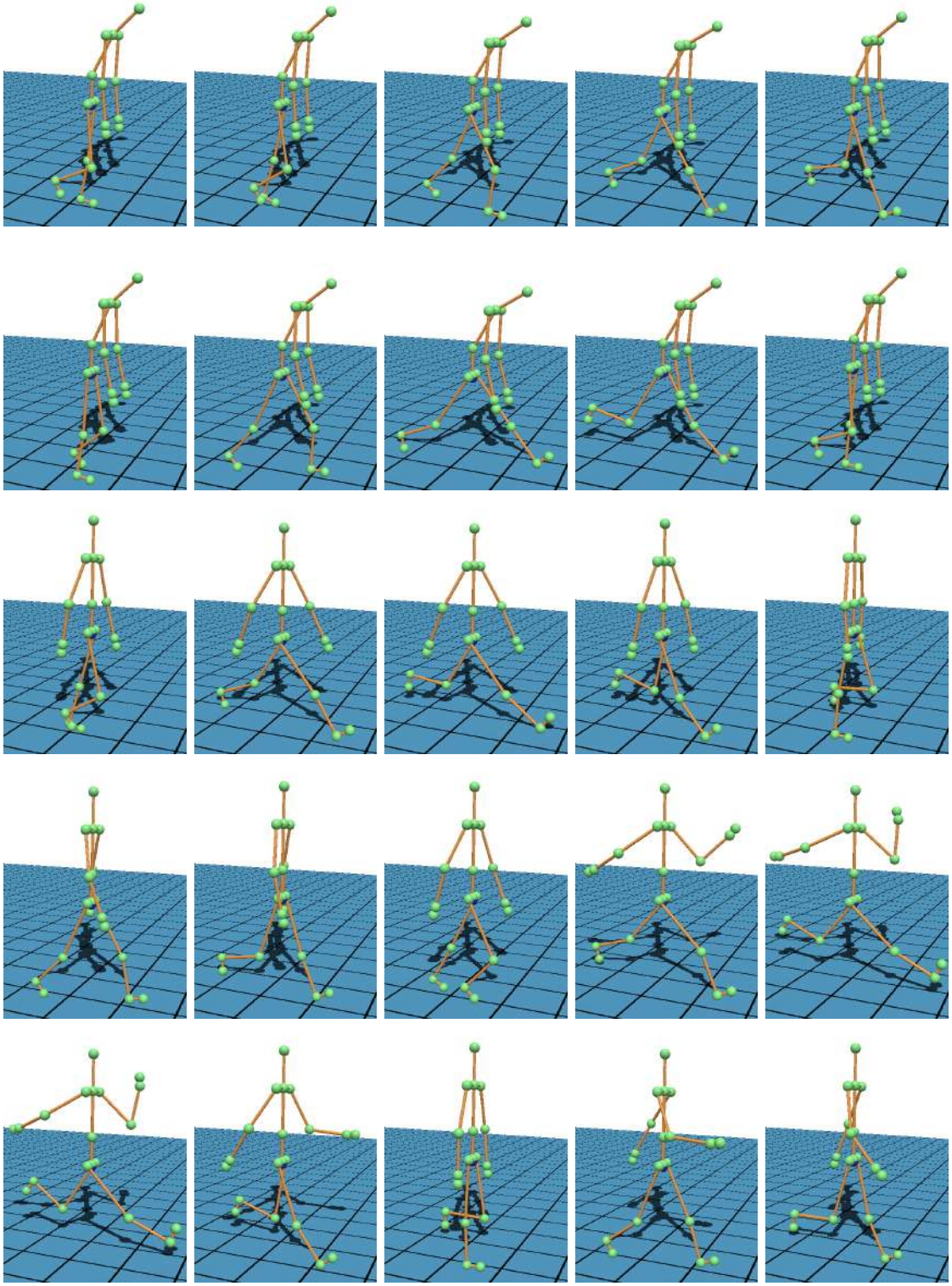


Figura 6.6: Novas poses geradas através da interpolação das poses-chaves dos diferentes estilos de caminhada. A sequência está ordenada da esquerda para a direita e de cima para baixo.

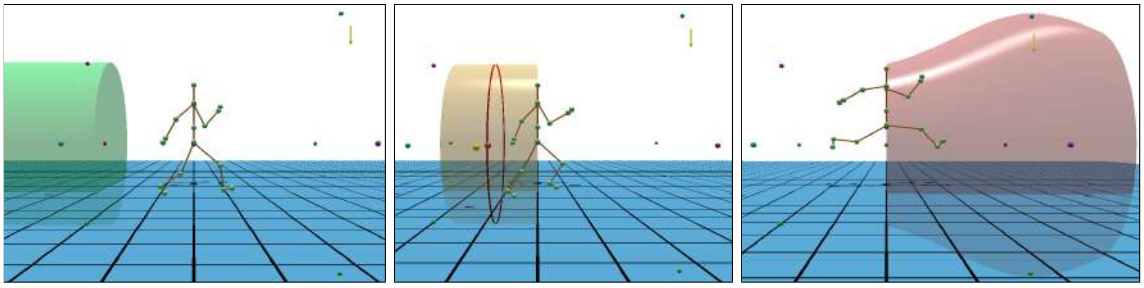
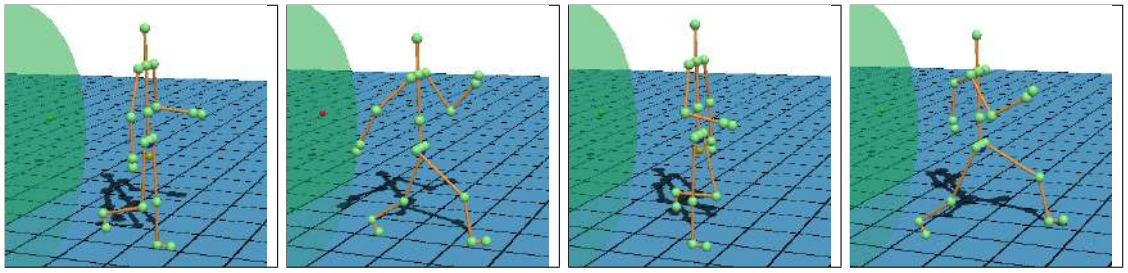
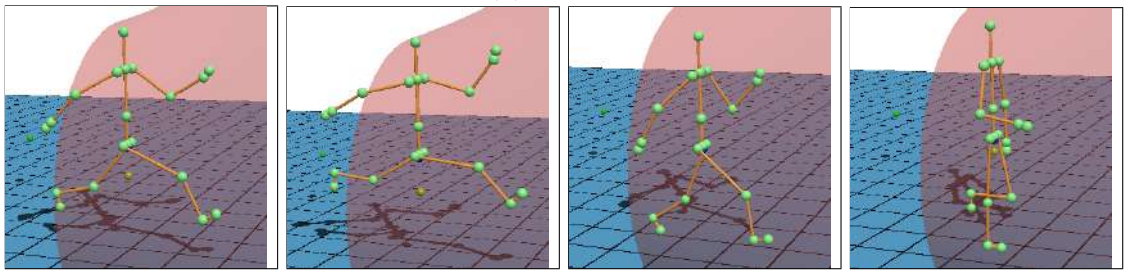


Figura 6.7: Animação de duas diferentes ações (corrida e salto) configuradas em distintos subconjuntos de marcadores (à esquerda e à direita). Na parte central, temos uma região de transição que depende de uma interpolação especial; nas demais, apenas os marcadores dos subconjuntos de marcadores de cada ação estão sendo utilizados na interpolação.



(a) Corrida.



(b) Salto.

Figura 6.8: Quadros-chaves utilizados para gerar uma animação composta de uma corrida e um salto.

6.2.4 Luta

Nessa animação, testamos uma animação com diversas ações distintas com marcadores posicionados sobre uma mesma superfície. Nesse caso, a interpolação é global. A Figura 6.10 apresenta as poses-chaves relativas as ações de soco, chute e defesa em um combate.

A Figura 6.11 exibe o resultado da animação de um combate.

6.2.5 Capoeira

A capoeira é uma arte marcial com origem em crioulos brasileiros. Para finalizar os nossos testes, nós produzimos uma animação com alguns golpes de capoeira.

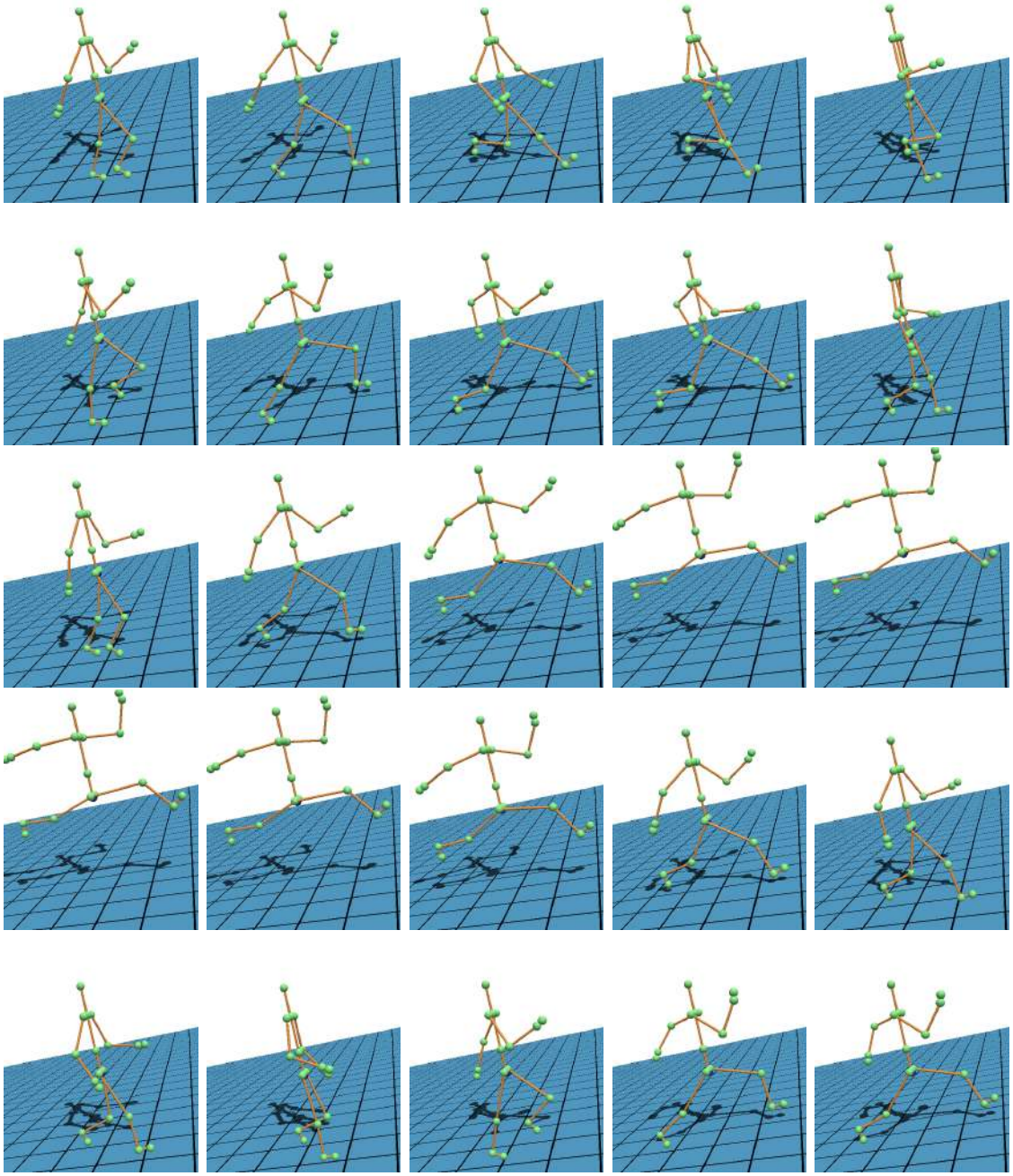


Figura 6.9: Sequência de novas poses ordenadas da esquerda para a direita e de cima para baixo, geradas através da interpolação das poses-chaves dos subconjuntos de marcadores de corrida e salto.

Nós apresentamos na Figura 6.13 os resultados da animação para os golpes ginga e martelo. O golpe básico, denominado ginga, consiste de movimentos repetitivos com a perna direita para a frente e a mão para trás, alternando o lado do corpo, com a perna esquerda para frente e a outra mão para trás. A partir desse movimento básico, lança-se os demais golpes. O martelo é um golpe com o peito do pé e consiste de levantar a perna até a altura da cabeça.

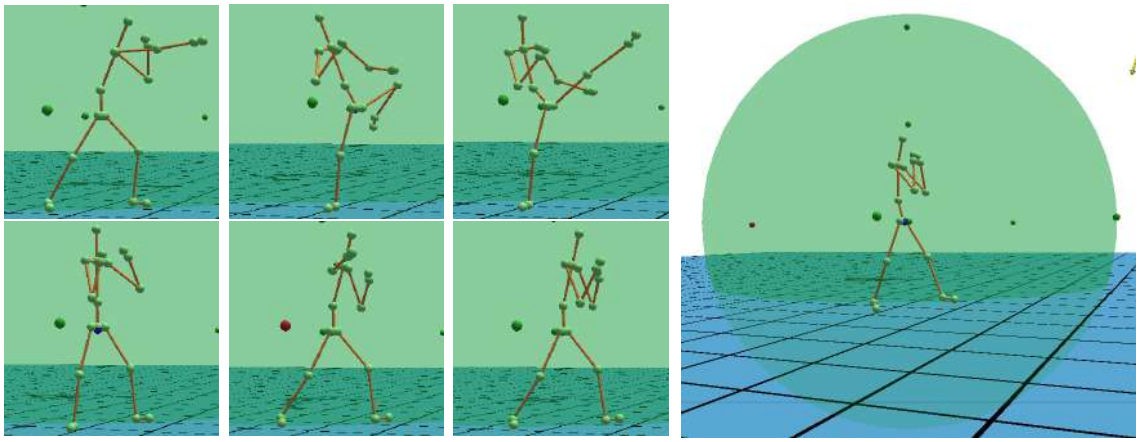


Figura 6.10: Poses-chaves dos marcadores de ações relativas a soco, defesa e chute sobre uma superfície esférica.

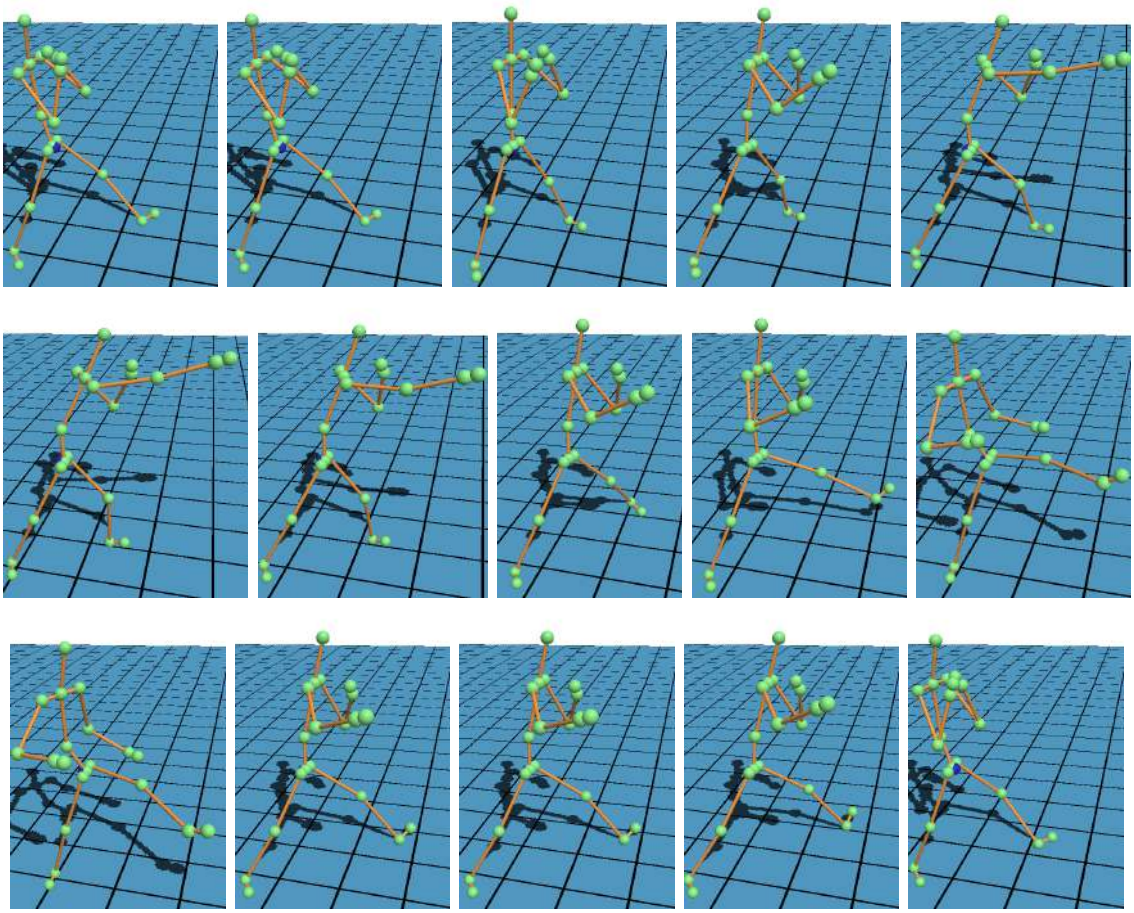


Figura 6.11: Sequência de novas poses ordenadas da esquerda para a direita e de cima para baixo, geradas através da interpolação das poses-chaves dos marcadores de soco, defesa e chute.

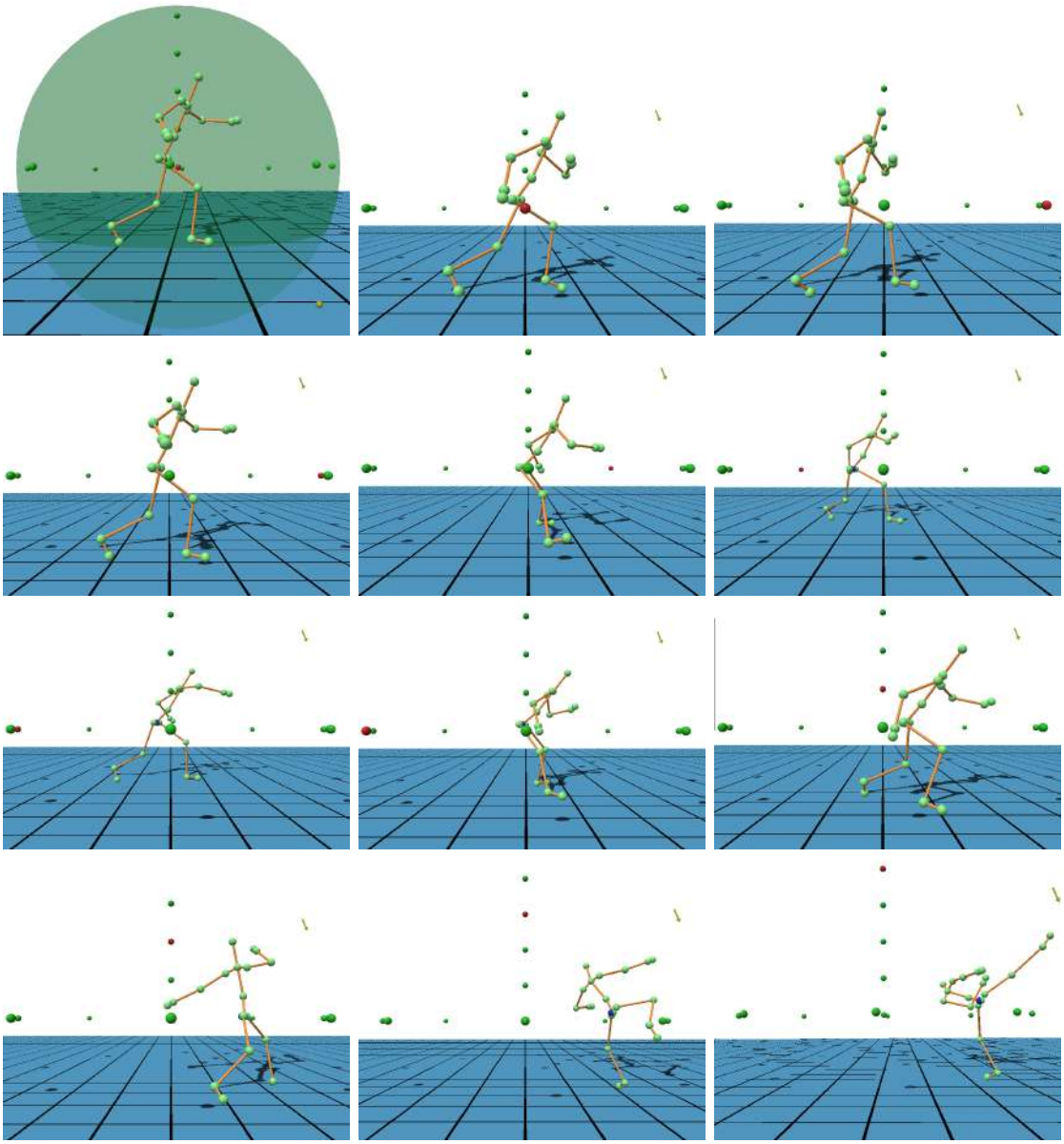


Figura 6.12: Poses-chaves dos golpes ginga e martelo sobre uma superfície esférica.

6.3 Reflexões

O nosso sistema foi inteiramente desenvolvido, na medida do possível, de forma simples e minimalista. Ele requer tão somente um sensor de profundidade acessível capaz de capturar gestos e movimentos das mãos em tempo real, além dos dispositivos usuais, como um computador com tela 2D e um mouse. As ideias foram desenvolvidas tão somente sobre um modelo articulado 3D hierárquico como o apresentado na Subseção 5.1.1. Posteriormente, pode-se ajustar um personagem ao modelo, como no método proposto por BARAN e POPOVIĆ [62], ou algum outro método que tenha a mesma finalidade. No entanto, isso não faz parte do objetivo principal desse trabalho e o modelo adotado foi suficiente para realizar os testes. Uma outra possibilidade poderia ser utilizar personagens com modelos mais complexos, altamente

articulados.

A interface desenvolvida apresenta um recurso gestual, além do visual e o espaço de interação limita-se ao campo de visão do dispositivo. O usuário precisa realizar um determinado gesto e movimentar a mão no campo de visão de um sensor de profundidade para garantir que o sistema utilize os dados capturados para produzir as animações. Geralmente, realizar um gesto com as mãos e fazer com que o sistema utilizado faça a captura do mesmo com precisão não é uma tarefa trivial. Uma dificuldade encontrada para determinar um gesto é, justamente, saber quando ele está iniciando, quando ele está sendo executado e quando ele já finalizou. É preciso cautela para descrever tais ações quando estamos lidando com captura de gestos para interfaces de animações em tempo real. Uma boa escolha do dispositivo também deve ser levada em consideração. Nós optamos por utilizar um dispositivo 3D *hands-free* para captura de gestos e movimentos da mão, o que dá uma liberdade maior de movimentação.

Em se tratando do mundo real, geralmente cada pessoa dá 2 passos em 1 segundo. Ao utilizar métodos tradicionais baseados em quadro-chaves, o animador precisa estar atento para ajustar o tempo às poses-chaves e construir as poses intermediárias necessárias para que se faça uma animação realista. Em nossa abordagem, não nos preocupamos em ajustar as poses-chaves ao tempo, pois a própria performance do usuário determina a velocidade da animação. Movimentos curtos ou amplos também terão influência sobre o resultado final da animação.

Como é característico na animação por quadros-chaves, o resultado final sempre depende de uma boa escolha das poses-chaves. Em alguns momentos, percebemos movimentos exagerados, como ocorre em algumas poses da Figura 6.6. Nela, podemos observar que a abertura da perna ocorre de forma bem ampla, levando a uma dobra que não condiz com os movimentos de uma caminhada realista. Isso pode ser um problema quando se quer produzir animações com resultados fiéis aos movimentos humanos.

O sistema funciona de forma bastante satisfatória quando realizamos animações simples, com um ou diversos estilos de movimento. Bem como, quando realizamos animações complexas, com ações diferentes, utilizando uma região de interpolação especial.

Ao gerar animações de uma mesma ação com diversos estilos de movimento, que contém poses-chaves semelhantes nos diversos subconjuntos de quadros-chaves, como ocorre na Figura 6.5, além de facilitar o trabalho do animador, pois reduz consideravelmente o seu esforço ao produzir as poses-chaves, ainda torna a animação mais suave sem a necessidade de configurações adicionais.

Já nas animações complexas, com ações diferentes, além da interpolação especial, o recurso de designar marcadores de entrada e de saída em cada uma das regiões

de interpolação local, fez com que a integração das ações durante a performance do usuário ocorresse de forma contínua e suave.

O sistema não é muito confiável quando configura-se poses-chaves com movimentos extremos de ações muito diferentes em uma mesma superfície, quando é necessário realizar interpolações globais. Isso pode ser observado em animações do tipo visto nas Subseções 6.2.4 e 6.2.5. Na imagem da quarta linha e primeira coluna da Figura 6.13, podemos observar uma pose nova muito diferente das poses-chaves originais.

Uma outra limitação do nosso sistema seria em relação ao limite do campo de visão do dispositivo adotado. Como ele não dispõe de um recurso visual ou de contato físico, algumas vezes o usuário não consegue permanecer com a mão dentro do campo de visão do sensor. Se a mão do usuário não estiver no campo de alcance do sensor, o sistema não irá responder de forma adequada, pois a posição do controlador é um dado de entrada fundamental para realizar as animações, sem ela, o sistema fica inerte. Ademais, é necessário um curto treinamento inicial para conseguir posicionar a mão dentro do espaço virtual de forma adequada e com uma certa naturalidade.

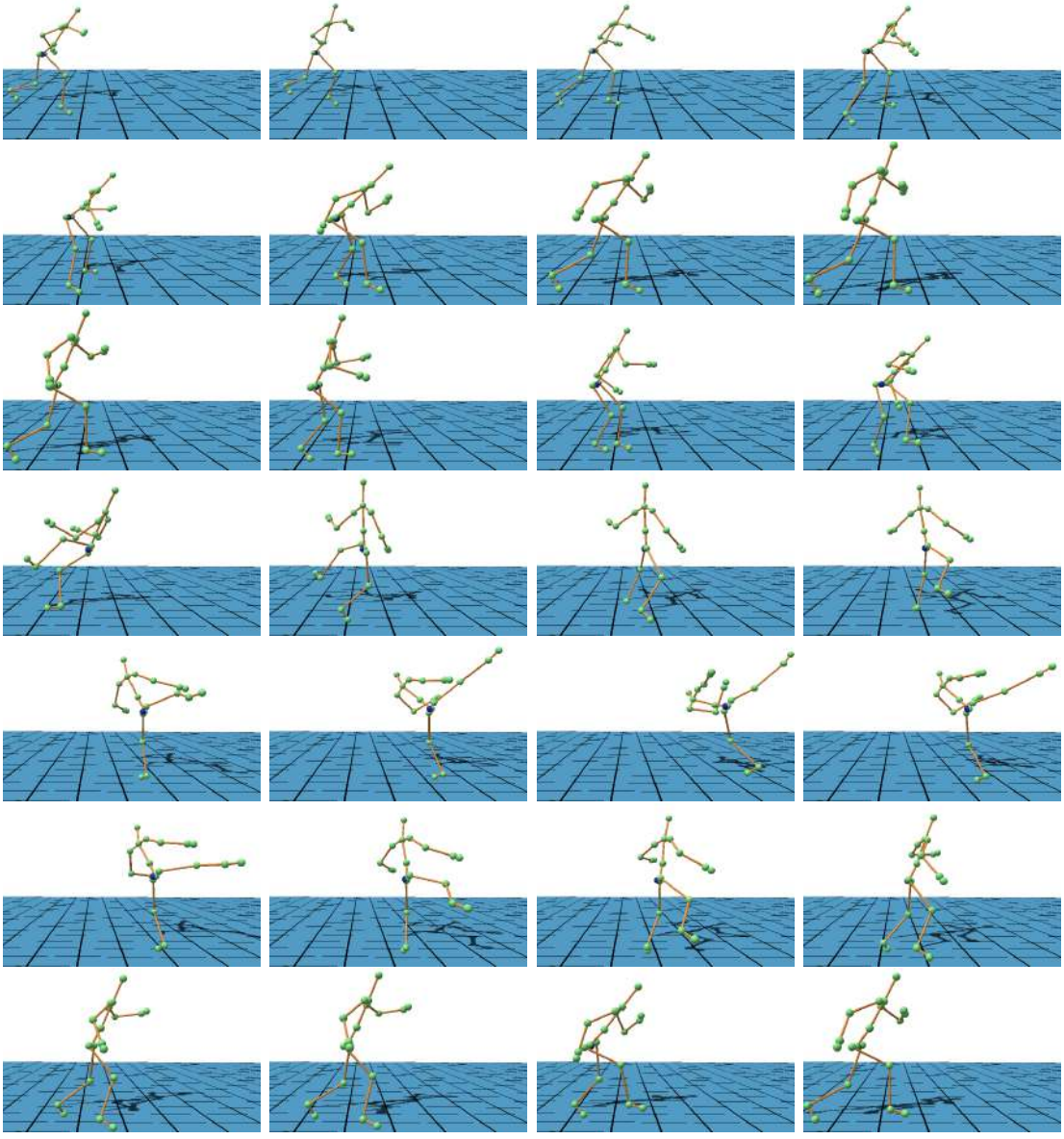


Figura 6.13: Sequência de novas poses ordenadas da esquerda para a direita e de cima para baixo, geradas através da interpolação das poses-chaves configuradas para produzir os golpes de capoeira ginga e martelo.

Capítulo 7

Conclusões

Neste trabalho, apresentamos o resultado da nossa pesquisa de doutorado que resultou em uma interface natural 3D para produzir animações de personagens articulados de forma interativa, ao utilizar um dispositivo para capturar a posição da mão do usuário e empregá-la no sistema.

Nós acreditamos que as interações 3D tornar-se-ão uma realidade comum em um futuro próximo. Por isso, a busca de novos paradigmas que comportam sistemas de interações natural é de grande importância. O nosso sistema funciona com sensores de profundidade capazes de detectar gestos e movimentos das mãos, independentemente do dispositivo utilizado. Nós apresentamos uma ideia simples de interação através de gestos das mãos por meio de um sistema minimalista, fácil de ser configurado, que utiliza tão somente, um computador com um mouse e uma tela 2D e um sensor de profundidade capaz de reconhecer gestos e movimentos da mão.

Em nosso método, nós apresentamos uma extensão da técnica de quadros-chaves espaciais que permite uma transição suave entre diversas ações ou estilos de movimento em uma mesma animação. Como incremento, para melhorar o alcance visual no espaço 3D, nós utilizamos superfícies para apoiar os marcadores, pois acreditamos que assim, facilita-se a localização dos mesmos no espaço virtual tridimensional.

Nós desenvolvemos uma interface para realizar alguns testes e poder validar o nosso método. Para isso, apresentamos os resultados de algumas animações com diferentes configurações e recursos do sistema: caminhada, corrida e salto, luta e capoeira.

Embora apresente resultados satisfatórios, o sistema ainda pode ser ampliado de diversas formas. Como trabalhos futuros, gostaríamos de permitir mais de um controle, para criar animações combinadas, como por exemplo andar e acenar a mão ao mesmo tempo. Gostaríamos também de criar mais animações e usar esqueletos mais complexos, diferentes dos bípedes.

Finalmente, gostaríamos de utilizar algum tipo de suporte físico, como uma haste colocada no eixo correspondente ao eixo principal das superfícies, para melhorar

ainda mais a sensação de posição do usuário em relação ao dispositivo de captura, sem impor limitações severas de movimento, o que pode aumentar a precisão do movimento da mão.

Referências Bibliográficas

[1] WRIGHT, J. *Animation Writing and Development: From Script Development to Pitch*. Focal Press visual effects and animation series. Focal Press, 2005. ISBN: 9780240805498.

[2] DENSLOW, P. K. "What is animation and who needs to know? an essay on definitions", *A Reader in Animation Studies. John Libbey, Sydney*, 1997.

[3] WILLIAMS, R. *The Animator's Survival Kit–Revised Edition: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators*. Faber & Faber, Inc., 2009. ISBN: 0571238343, 9780571238347.

[4] WATT, A. *3D Computer Graphics*. 2nd ed. Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc., 1993. ISBN: 0201631865.

[5] THALMANN, N. M., THALMANN, D. "Computer Animation". In: *Computer Animation: Theory and Practice*, pp. 13–17, Tokyo, Springer Japan, 1990. ISBN: 978-4-431-68105-2. doi: 10.1007/978-4-431-68105-2_3. Disponível em: <http://dx.doi.org/10.1007/978-4-431-68105-2_3>.

[6] PARENT, R. *Computer Animation: Algorithms and Techniques*. 3 ed. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 2012. ISBN: 9780124158429, 9780124159730.

[7] GEIJTENBEEK, T., PRONOST, N. "Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review". In: *Computer Graphics Forum*, v. 31, pp. 2492–2515. Wiley Online Library, 2012.

[8] POPOVIĆ, Z., WITKIN, A. "Physically Based Motion Transformation". In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pp. 11–20, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN: 0-201-48560-5. doi: 10.1145/311535.311536. Disponível em: <http://dx.doi.org/10.1145/311535.311536>.

[9] LASSETER, J. "Principles of Traditional Animation Applied to 3D Computer Animation". In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pp. 35–44, New York, NY, USA, 1987. ACM. ISBN: 0-89791-227-6. doi: 10.1145/37401.37407. Disponível em: <http://doi.acm.org/10.1145/37401.37407>.

[10] STURMAN, D. "Interactive Keyframe Animation of 3-D Articulated Models". In: *Proceedings of Graphics Interface '84*, GI '84, pp. 35–40, 1984. Disponível em: <http://graphicsinterface.org/wp-content/uploads/gi1984-6.pdf>.

[11] IGARASHI, T., MOSCOVICH, T., HUGHES, J. F. "Spatial Keyframing for Performance-driven Animation". In: *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pp. 107–115, New York, NY, USA, 2005. ACM. ISBN: 1-59593-198-8. doi: 10.1145/1073368.1073383. Disponível em: <http://doi.acm.org/10.1145/1073368.1073383>.

[12] SUTHERLAND, I. E. "Sketchpad: A Man-machine Graphical Communication System". In: *Proceedings of the May 21-23, 1963, Spring Joint Computer Conference*, AFIPS '63 (Spring), pp. 329–346, New York, NY, USA, 1963. ACM. doi: 10.1145/1461551.1461591. Disponível em: <http://doi.acm.org/10.1145/1461551.1461591>.

[13] SUTHERLAND, I. E. "The Ultimate Display". In: *Proceedings of the IFIP Congress*, pp. 506–508, 1965.

[14] SUTHERLAND, I. E. "A Head-Mounted Three-Dimensional Display". In: *AFIPS Conference Proceedings (1968) 33, I*, pp. 757–764, 1968.

[15] ZHANG, Z. *Microsoft Kinect Sensor and Its Effect*. Relatório técnico, April 2012. Disponível em: <https://www.microsoft.com/en-us/research/publication/microsoft-kinect-sensor-and-its-effect/>.

[16] Microsoft Kinect. "Kinect for Xbox One website". http://www.xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one, . Acessado em 15/08/2016.

[17] Microsoft Kinect. "Kinect Sensor website". https://msdn.microsoft.com/en-us/library/hh438998.aspx, . Acessado em 05/07/2015.

[18] CRUZ, L., LUCIO, D., VELHO, L. "Kinect and RGBD Images: Challenges and Applications". In: *Proceedings of the 2012 25th SIBGRAPI Conference*

on Graphics, Patterns and Images Tutorials, SIBGRAPI-T '12, pp. 36–49, Washington, DC, USA, 2012. IEEE Computer Society. ISBN: 978-0-7695-4830-2. doi: 10.1109/SIBGRAPI-T.2012.13. Disponível em: <http://dx.doi.org/10.1109/SIBGRAPI-T.2012.13>.

[19] SMISEK, J., JANCOSEK, M., PAJDLA, T. "3D with Kinect". In: *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, Nov 2011. doi: 10.1109/ICCVW.2011.6130380.

[20] Xtion PRO LIVE. "Xtion PRO LIVE website". https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/. Acessado em 05/11/2016.

[21] Creative Senz3D. "Creative Senz3D website". http://us.creative.com/p/web-cameras/creative-senz3d. Acessado em 26/04/2017.

[22] Intel RealSense Cameras. "Intel RealSense Cameras website". http://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html. Acessado em 26/04/2017.

[23] Structure Sensor. "Structure Sensor website". http://structure.io/. Acessado em 05/11/2016.

[24] Leap Motion. "Leap Motion website". https://www.leapmotion.com/. Acessado em 10/08/2016.

[25] GUNA, J., JAKUS, G., POGAČNIK, M., et al. "An Analysis of the Precision and Reliability of the Leap Motion Sensor and Its Suitability for Static and Dynamic Tracking", *Sensors*, v. 14, n. 2, pp. 3702, 2014. ISSN: 1424-8220. doi: 10.3390/s140203702. Disponível em: <http://www.mdpi.com/1424-8220/14/2/3702>.

[26] Myo. "Myo website". https://www.myo.com/. Acessado em 05/11/2016.

[27] Ring ZERO. "Ring ZERO website". http://ringzero.logbar.jp/. Acessado em 05/11/2016.

[28] BOWMAN, D. A., KRUIJFF, E., LAVIOLA, J. J., et al. "An Introduction to 3D User Interface Design", *Presence: Teleoper. Virtual Environ.*, v. 10, n. 1, pp. 96–108, fev. 2001. ISSN: 1054-7460. doi: 10.1162/105474601750182342. Disponível em: <http://dx.doi.org/10.1162/105474601750182342>.

[29] BOWMAN, D. A., KRUIJFF, E., LAVIOLA, J. J., et al. *3D User Interfaces: Theory and Practice*. Redwood City, CA, USA, Addison Wesley Longman Publishing Co., Inc., 2004. ISBN: 0201758679.

[30] BARNES, C., JACOBS, D. E., SANDERS, J., et al. "Video Puppetry: A Performative Interface for Cutout Animation". In: *ACM SIGGRAPH Asia 2008 Papers*, SIGGRAPH Asia '08, pp. 124:1–124:9, New York, NY, USA, 2008. ACM. ISBN: 978-1-4503-1831-0. doi: 10.1145/1457515.1409077. Disponível em: <http://doi.acm.org/10.1145/1457515.1409077>.

[31] GUPTA, S., JANG, S., RAMANI, K. "PuppetX: a framework for gestural interactions with user constructed playthings". In: *AVI'14*, pp. 73–80, 2014.

[32] LUO, Z., CHEN, I.-M., YEO, S. H., et al. "Building Hand Motion-Based Character Animation: The Case of Puppetry." In: Sourin, A., Sourina, O. (Eds.), *CW*, pp. 46–52. IEEE Computer Society, 2010. ISBN: 978-0-7695-4215-7. Disponível em: <http://dblp.uni-trier.de/db/conf/cw/cw2010.html#Luo10HMB>.

[33] LUO, Z., LIN, C.-C., CHEN, I.-M., et al. "Puppet Playing: An Interactive Character Animation System with Hand Motion Control". In: Gavrilova, M., Tan, C., Sourin, A., et al. (Eds.), *Transactions on Computational Science XII*, v. 6670, *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 19–35, 2011. ISBN: 978-3-642-22335-8. doi: 10.1007/978-3-642-22336-5_2.

[34] TSOLI, A., MAHMOOD, N., BLACK, M. J. "Breathing Life into Shape: Capturing, Modeling and Animating 3D Human Breathing", *ACM Transactions on Graphics, (Proc. SIGGRAPH)*, v. 33, n. 4, pp. 52:1–52:11, jul. 2014. Disponível em: <http://dl.acm.org/citation.cfm?doid=2601097.2601225>.

[35] XIA, S., WANG, C., CHAI, J., et al. "Realtime Style Transfer for Unlabeled Heterogeneous Human Motion", *ACM Trans. Graph.*, v. 34, n. 4, pp. 119:1–119:10, jul. 2015. ISSN: 0730-0301. doi: 10.1145/2766999. Disponível em: <http://doi.acm.org/10.1145/2766999>.

[36] SHIRATORI, T., MAHLER, M., TREZEVANT, W., et al. "Expressing animated performances through puppeteering." In: Lécuyer, A., Steinicke, F., Billinghurst, M. (Eds.), *3DUI*, pp. 59–66. IEEE, 2013. ISBN: 978-1-4673-6097-5. Disponível em: <http://dblp.uni-trier.de/db/conf/3dui/3dui2013.html#ShiratoriMTH13>.

[37] OSHITA, M., SENJU, Y., MORISHIGE, S. "Character motion control interface with hand manipulation inspired by puppet mechanism." In: Wang, C.

C. L., Magnenat-Thalmann, N., Pan, Z. (Eds.), *VRCAI*, pp. 131–138. ACM, 2013. ISBN: 978-1-4503-2590-5. Disponível em: <http://dblp.uni-trier.de/db/conf/vrcai/vrcai2013.html#OshitaSM13>.

[38] NINOMIYA, D., MIYAZAKI, K., NAKATSU, R. "Networked virtual marionette theater". In: *Technologies for E-Learning and Digital Entertainment*, Springer, pp. 619–627, 2008.

[39] LU, F., TIAN, F., JIANG, Y., et al. "ShadowStory: creative and collaborative digital storytelling inspired by cultural heritage." In: Tan, D. S., Amershi, S., Begole, B., et al. (Eds.), *CHI*, pp. 1919–1928. ACM, 2011. ISBN: 978-1-4503-0228-9. Disponível em: <http://dblp.uni-trier.de/db/conf/chi/chi2011.html#LvTJCLLZDW11>.

[40] ZHANG, H., SONG, Y., CHEN, Z., et al. "Chinese Shadow Puppetry with an Interactive Interface Using the Kinect Sensor". In: *Proceedings of the 12th International Conference on Computer Vision - Volume Part I*, ECCV'12, pp. 352–361, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN: 978-3-642-33862-5. doi: 10.1007/978-3-642-33863-2_35. Disponível em: <http://dx.doi.org/10.1007/978-3-642-33863-2_35>.

[41] LEITE, L., ORVALHO, V. "Shape Your Body: Control a Virtual Silhouette Using Body Motion". In: *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pp. 1913–1918, New York, NY, USA, 2012. ACM. ISBN: 978-1-4503-1016-1. doi: 10.1145/2212776.2223728. Disponível em: <http://doi.acm.org/10.1145/2212776.2223728>.

[42] WU, Q., BOULANGER, P., KAZAKEVICH, M., et al. "A Real-time Performance System for Virtual Theater". In: *Proceedings of the 2010 ACM Workshop on Surreal Media and Virtual Cloning*, SMVC '10, pp. 3–8, New York, NY, USA, 2010. ACM. ISBN: 978-1-4503-0175-6. doi: 10.1145/1878083.1878087. Disponível em: <http://doi.acm.org/10.1145/1878083.1878087>.

[43] GLAUSER, O., MA, W.-C., PANOZZO, D., et al. "Rig Animation with a Tangible and Modular Input Device", *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 2016.

[44] CHOI, B., YOU, M., NOH, J. "Extended spatial keyframing for complex character animation", *Computer Animation and Virtual Worlds*, jan 2008. Disponível em: <http://koasas.kaist.ac.kr/bitstream/10203/24306/1/cavw08_esk.pdf>.

[45] CICCONE, L., GUAY, M., NITTI, M., et al. “Authoring Motion Cycles”. In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA ’17, pp. 8:1–8:9, New York, NY, USA, 2017. ACM. ISBN: 978-1-4503-5091-4. doi: 10.1145/3099564.3099570. Disponível em: <http://doi.acm.org/10.1145/3099564.3099570>.

[46] KIPP, M., NGUYEN, Q. “Multitouch Puppetry: Creating Coordinated 3D Motion for an Articulated Arm”. In: *ACM International Conference on Interactive Tabletops and Surfaces*, ITS ’10, pp. 147–156, New York, NY, USA, 2010. ACM. ISBN: 978-1-4503-0399-6. doi: 10.1145/1936652.1936682. Disponível em: <http://doi.acm.org/10.1145/1936652.1936682>.

[47] HELD, R., GUPTA, A., CURLESS, B., et al. “3D Puppetry: A Kinect-based Interface for 3D Animation”. In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST ’12, pp. 423–434, New York, NY, USA, 2012. ACM. ISBN: 978-1-4503-1580-7. doi: 10.1145/2380116.2380170. Disponível em: <http://doi.acm.org/10.1145/2380116.2380170>.

[48] GUPTA, A., AGRAWALA, M., CURLESS, B., et al. “MotionMontage: A System to Annotate and Combine Motion Takes for 3D Animations”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’14, pp. 2017–2026, New York, NY, USA, 2014. ACM. ISBN: 978-1-4503-2473-1. doi: 10.1145/2556288.2557218. Disponível em: <http://doi.acm.org/10.1145/2556288.2557218>.

[49] KATO, H., BILLINGHURST, M., POUPYREV, I., et al. “Virtual Object Manipulation on a Table-Top AR Environment, ISAR”, *Presence*, pp. 111–119, 2000.

[50] DE ARAÙJO, B. R., CASIEZ, G., JORGE, J. A. “Mockup Builder: Direct 3D Modeling on and Above the Surface in a Continuous Interaction Space”. In: *Proceedings of Graphics Interface 2012*, GI ’12, pp. 173–180, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society. ISBN: 978-1-4503-1420-6. Disponível em: <http://dl.acm.org/citation.cfm?id=2305276.2305305>.

[51] CASIEZ, G., ROUSSEL, N., VOGEL, D. “1€ Filter: A Simple Speed-based Low-pass Filter for Noisy Input in Interactive Systems”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’12, pp. 2527–2530, New York, NY, USA, 2012. ACM. ISBN: 978-

1-4503-1015-4. doi: 10.1145/2207676.2208639. Disponível em: <http://doi.acm.org/10.1145/2207676.2208639>.

[52] OpenGL. "OpenGL website". http://www.opengl.org/. Acessado em 25/07/2016.

[53] DO CARMO, M. P. *Differential geometry of curves and surfaces.* Prentice Hall, 1976. ISBN: 978-0-13-212589-5.

[54] PRESSLEY, A. *Elementary differential geometry.* Springer undergraduate mathematics series. London, Dordrecht, Heidelberg, Springer, 2010. ISBN: 978-1-84882-890-2.

[55] GRAY, A. *Modern Differential Geometry of Curves and Surfaces with Mathematica.* 1st ed. Boca Raton, FL, USA, CRC Press, Inc., 1996. ISBN: 0849371643.

[56] TURK, G., O'BRIEN, J. F. "Modelling with Implicit Surfaces That Interpolate", *ACM Trans. Graph.*, v. 21, n. 4, pp. 855–873, out. 2002. ISSN: 0730-0301. doi: 10.1145/571647.571650. Disponível em: <http://doi.acm.org/10.1145/571647.571650>.

[57] Tucano. "Tucano: A library for rapid prototyping with modern OpenGL and GLSL website". http://www.lcg.ufrj.br/tucano/. Acessado em 18/12/2015.

[58] SOUZA, A. L. E. L., MARROQUIM, R., VELHO, L. "Sketches on natural interactions with virtual scenes". In: *Workshop of Works in Progress (WIP) in SIBGRAPI (XXVIII Conference on Graphics, Patterns and Images)*, 2015.

[59] SHARP, T., KESKIN, C., ROBERTSON, D., et al. "Accurate, Robust, and Flexible Real-time Hand Tracking". CHI, April 2015. Disponível em: <http://research.microsoft.com/apps/pubs/default.aspx?id=238453>.

[60] Autodesk. "Autodesk website". http://www.autodesk.com/. Acessado em 08/02/2017.

[61] Blender. "Blender website". https://www.blender.org/. Acessado em 08/02/2017.

[62] BARAN, I., POPOVIĆ, J. "Automatic Rigging and Animation of 3D Characters". In: *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. doi: 10.1145/1275808.1276467. Disponível em: <http://doi.acm.org/10.1145/1275808.1276467>.