Simulando Reações Flexíveis em Movimentos Capturados

Rubens F. Nunes[1], Creto A. Vidal[1], Joaquim B. Cavalcante-Neto[1], Victor B. Zordan[2]
[1]*Universidade Federal do Ceará* [2]*University of California, Riverside, USA*
{rubens, cvidal, joaquim}@lia.ufc.br, vbz@cs.ucr.edu

Abstract

Combining physically-based simulation and animation obtained by motion-captured data is a new technique used to adapt the motions captured in studio to the application's needs. In this paper, the focus is on adapting motion-captured sequences to flexible reactions of a virtual human to perturbations, without losing motion realism. The proposed technique follows some of the ideas presented in the literature, but uses a new way of computing feedforward controllers through an auxiliary dynamic simulation. This new computation is simple, general and efficient, and allows the computation of the feedforward controllers in run time. These claims are demonstrated through experiments described in the example section and in the accompanying videos.

1. Introdução

1.1. Descrição do problema

A animação realista de personagens virtuais é um dos fatores determinantes do sentimento de imersão em ambientes virtuais. O interesse por esse tipo de animação tem aumentado tanto nas indústrias de jogos e entretenimento como em outras diversas áreas, tais como biomecânica, robótica e reabilitação, e grandes avanços ocorreram durante as últimas décadas [10]. Mais especificamente, o cuidado em relação à forma como os personagens interagem entre si e com o ambiente é indispensável para a credibilidade da animação e tem um impacto direto na qualidade da percepção do espectador. Por exemplo, na animação de uma luta entre dois personagens, espera-se que o personagem atingido por um golpe exiba uma reação compatível com o golpe desferido por seu oponente.

A utilização de movimentos, capturados em estúdio, executados por atores reais está bastante difundida na indústria da computação gráfica. Entretanto, apesar do maior realismo apresentado pelos dados capturados, ainda é um desafio adaptar essas animações, de forma fácil, às possíveis interações com o ambiente sem introduzir artefatos que destruam o realismo. Por sua vez, o emprego de simulação física permite gerar mais facilmente respostas interativas a perturbações do ambiente, através da aplicação de forças de colisão ao modelo. No entanto, a dificuldade de se projetar controladores para modelos complexos, tais como humanóides, que sejam capazes de produzir movimentos com qualidade semelhante à obtida por captura de movimentos, limita sua aplicabilidade.

A combinação das duas abordagens – movimentos capturados e simulação física – busca aproveitar o que essas duas técnicas têm de melhor. Zordan e Hodgins [22] introduziram controladores capazes de seguir movimentos capturados. Em cada passo da simulação, controladores do tipo PD (*Proportional Derivative*) calculam torques para corrigir o erro entre o estado atual, obtido a partir da simulação (*feedback*), e o estado desejado, determinado a partir dos dados capturados. Esses controladores simulam os músculos responsáveis pelo movimento nas juntas (articulações) e seus ganhos (k_s e k_d) correspondem às rigidezes desses músculos. Vale salientar, no entanto, que atribuir valores a esses ganhos é uma tarefa difícil, já que a atribuição de valores elevados leva o modelo a reagir de maneira rígida e inflexível às perturbações do ambiente; enquanto que a atribuição de valores baixos impossibilita seguir com precisão o movimento capturado.

Diante da inexistência de valores intermediários capazes de satisfazer simultaneamente essas duas exigências, Zordan e Hodgins [22] mantêm altos os

valores dos ganhos para que o modelo siga o movimento capturado com precisão, e os reduzem temporariamente apenas na ocorrência de um impacto. Contudo, o uso predominante de ganhos altos não é realístico em sistemas biológicos [20].

Em síntese, o problema abordado neste artigo é: "Como simular reações flexíveis a perturbações no ambiente enquanto o personagem virtual segue fielmente movimentos capturados, isto é, sem perder as sutilezas e o estilo que conferem realismo ao movimento?".

1.2. Solução proposta

De acordo com [20], sistemas biológicos usam controle *feedforward* (controle de laço aberto, ou seja, que não tem acesso ao estado atual do sistema) na realização da maior parte do trabalho, e usam controle *feedback* de baixos ganhos apenas na correção de pequenos possíveis erros em relação à trajetória desejada. Yin et al. [20] mostram que, se um termo *feedforward* for adicionado à equação de controle, controladores *feedback* de baixos ganhos podem ser capazes de seguir fielmente movimentos capturados. Assim, quando um impacto ocorre, o modelo reage de maneira flexível, sem que seja necessário atualizar os ganhos, como ocorre no método proposto em [22].

A solução proposta neste trabalho toma como base o trabalho de Yin et al. [20] e substitui o método de cálculo do termo *feedforward* por um novo método, mais simples, mais geral e mais eficiente, que permite a obtenção do termo *feedforward* em tempo de execução. Assim, ao invés de usar dinâmica inversa como em [20], calcula-se o termo *feedforward* através de uma simulação física auxiliar do modelo a qual usa controladores *feedback* de altos ganhos e é executada simultaneamente à simulação física principal. Os impactos inesperados são levados em conta apenas na simulação física principal que, por sua vez, usa controladores *feedback* de baixos ganhos. Em cada passo da simulação, os termos (torques) *feedforward,* obtidos pela simulação auxiliar, são incorporados ao conjunto de torques resultantes dos controladores *feedback* de baixos ganhos e aplicados ao modelo durante a simulação principal (Figura 1). O método proposto também permite a simulação de reações a impactos esperados como, por exemplo, a defesa de um soco. Para esses casos, onde o modelo deve reagir deliberadamente de maneira rígida, o impacto deve ser considerado tanto na simulação principal quanto na simulação auxiliar.

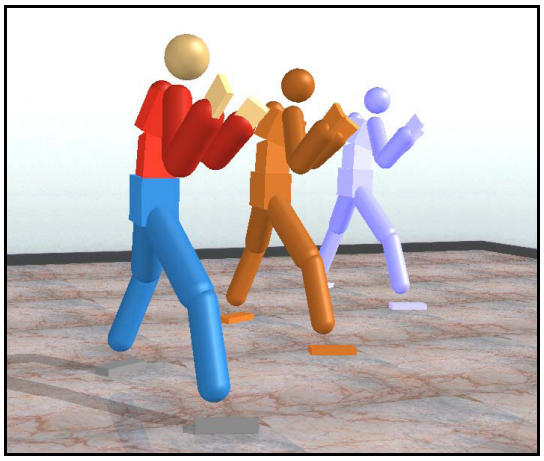

Figura 1. Movimento capturado original (direita); Simulação auxiliar (centro) calcula o termo *feedforward* usando *feedback* de altos ganhos; Simulação principal (esquerda) combina *feedforward* com *feedback* de baixos ganhos.

1.3. Organização do artigo

O restante deste artigo está organizado do seguinte modo: a Seção 2 contém uma discussão dos trabalhos relacionados; a Seção 3 consiste da explicação detalhada do novo método proposto para calcular o termo *feedforward*; a Seção 4 contém descrições de aspectos importantes de implementação; a Seção 5 consiste da apresentação e análise dos testes realizados; a Seção 6 é dedicada às discussões sobre as limitações do método proposto e sobre possíveis melhorias; e a Seção 7 contém as considerações finais sobre o trabalho.

2. Trabalhos relacionados

Vários pesquisadores, utilizando simulação física, se empenharam em projetar controladores adequados a movimentos específicos desejados. Hodgins et al. [5], utilizando máquinas de estados finitos, projetaram manualmente uma série de controladores para atletas humanos, os quais geram movimentos tais como corrida, ciclismo e salto sobre cavalo. Alguns autores propuseram representações de controladores que podem ser projetadas automaticamente para modelos menos complexos, utilizando otimização estocástica [12, 16, 17].

Gerar transições entre controladores também não é uma tarefa fácil. Wooten e Hodgins [18] projetaram manualmente controladores parametrizados para as seguintes ações de humanos virtuais: salto, manobras

aéreas acrobáticas, aterrissagem e equilíbrio; e permitiram que transições entre esses controladores fossem realizadas. Faloutsos et al. [2] propuseram um *framework* de composição mais geral que permite a adição de novos controladores. Um controlador supervisor de nível superior é responsável por realizar automaticamente as transições, baseando-se em pré-condições apropriadas para o uso de cada controlador individual.

Neff e Fiume [11] consideram o efeito de forças externas para obter automaticamente os parâmetros dos controladores *feedback* PD, através de uma formulação antagônica equivalente, tornando os controles de tensão e de posição independentes. Entretanto, uma contribuição adicional é exigida para seguir dados capturados usando essa formulação.

Modificar movimentos capturados para usá-los em novas situações é um desafio. Alguns autores sugeriram construir grafos de movimentos [6, 8]. De posse de um repositório, uma busca por quadros semelhantes é realizada, de acordo com uma métrica de similaridade. Transições são geradas entre esses quadros, interligando todo o repositório, para gerar o grafo.

Uma direção de pesquisa existente é desenvolver métodos adequados de pré-processamento do grafo, possibilitando que caminhos de movimentos desejados possam ser encontrados em tempo real. Lee e Lee [9] usaram programação dinâmica para simular lutadores de boxe capazes de atingir alvos específicos interativamente, percorrendo adequadamente um grafo de movimentos em tempo real. Reações a impactos foram simuladas cinematicamente.

Alguns trabalhos combinam as duas abordagens. Shapiro et al. [14] se basearam em [2] para criar um *framework* capaz de gerenciar controladores tanto dinâmicos como cinemáticos. Zordan et al. [24] usaram simulação física para produzir transições entre movimentos capturados a fim de simular reações a impactos.

Assim como o trabalho de Zordan e Hodgins [22], discutido na introdução, alguns outros trabalhos também propuseram estratégias para tratar o problema específico abordado neste artigo.

Pollard e Zordan [13] usaram uma abordagem semelhante à usada em [20] aplicada ao controle de uma mão. Um termo *feedforward* é pré-computado por dinâmica inversa para compensar a influência do movimento do braço e da gravidade.

Yin et al. [21] usam uma variação simplificada da técnica "*feedback error learning*", que é específica para movimentos cíclicos (e.g. um ciclo de um caminhar), e não tratam movimentos mais gerais como os utilizados neste trabalho.

Para seguir os movimentos capturados, Wrotek et al. [19] propuseram aplicar torques *feedback* externos diretamente em todos os corpos, baseados no erro em relação às coordenadas do mundo, ao invés das coordenadas de cada junta, e relataram que assim os ganhos são definidos de maneira mais fácil e mais flexível. Entretanto, para tratar reações a impactos, os ganhos tiveram que ser atualizados, usando a mesma estratégia apresentada em [22].

Allen et al. [1] apresentaram um método automático para calcular os ganhos, levando em conta informações de tempo (sincronização) dadas pelo usuário. Porém, ao seguir dados capturados, o método mantém altos ganhos e, no momento do impacto, também exige a atualização dos ganhos das juntas que devem ser influenciadas. Além disso, os testes realizados não envolvem movimentos atléticos.

Kry e Pai [7] apresentaram uma nova técnica em que, junto com o movimento, forças de contato são também capturadas. Essas informações são usadas para estimar a tensão nas juntas. Em contraste, o método proposto não exige acesso a essas informações.

3. Cálculo do termo feedforward

Para permitir que controladores *feedback* de baixos ganhos sejam capazes de seguir fielmente movimentos capturados, um termo *feedforward* adequado deve ser adicionado à equação de controle. O cálculo do termo *feedforward* é baseado na idéia de que, para movimentos bem treinados, sistemas biológicos já possuem informações que permitem o pré-processamento dos torques necessários à realização desses movimentos e usam controle *feedback* apenas para corrigir pequenos possíveis erros em relação à trajetória desejada.

Portanto, devido à boa qualidade do controle, os movimentos bem treinados são realizados com baixa tensão muscular (o que corresponde a baixos ganhos). Por outro lado, movimentos nunca antes treinados exigem que os músculos estejam mais rígidos para possibilitar a correção de maiores erros decorrentes do controle *feedforward* de baixa qualidade.

O método proposto neste trabalho estima o controle *feedforward* presente nos sistemas biológicos, através de uma simulação física auxiliar executada simultaneamente à simulação principal. A simulação auxiliar usa controladores *feedback* de altos ganhos para produzir torques que fazem o humano virtual seguir fielmente os movimentos capturados. Esse

processamento corresponde ao pré-processamento feito por sistemas biológicos ao realizar movimentos bem treinados. Os termos *feedforward* utilizados na simulação principal são exatamente os torques produzidos na simulação auxiliar. Assim, os torques aplicados no modelo durante a simulação principal são compostos pela soma dos termos *feedforward* com os torques produzidos pelos controladores *feedback* de baixos ganhos utilizados na simulação principal.

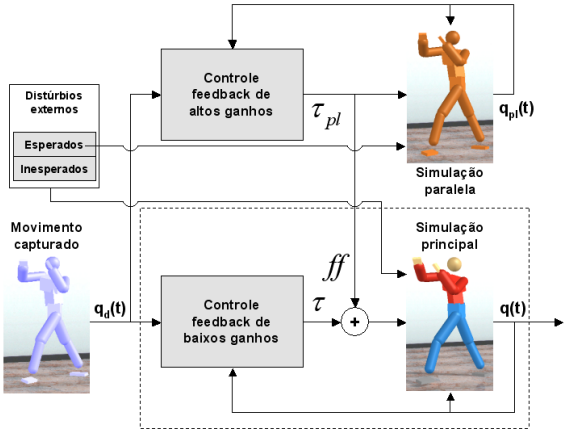

Figura 2. Esquema geral do método.

O esquema geral mostrado na Figura 2 ilustra os passos do método proposto: 1) Obtenção do estado desejado, $q_d(t)$, do modelo a partir do movimento capturado; 2) Cálculo dos torques, τ_{pl}, pelos controladores *feedback* de altos ganhos da simulação auxiliar, para corrigir o erro entre o estado atual, $q_{pl}(t)$, e o estado desejado, $q_d(t)$; 3) Cálculo dos torques, τ, pelos controladores *feedback* de baixos ganhos da simulação principal, para corrigir o erro entre o estado atual, $q(t)$, e o estado desejado, $q_d(t)$; 4) Aplicação dos torques, τ_{pl}, ao modelo da simulação auxiliar e repasse de τ_{pl} (torques *feedforward*, *ff*) à simulação principal; 5) Aplicação da soma τ + *ff* ao modelo da simulação principal; 6) Integração no tempo das duas simulações, considerando os distúrbios externos, e atualização dos estados atuais do modelo.

As duas simulações são atualizadas simultaneamente, o que permite calcular o termo *feedforward* em tempo de execução e dispensar, portanto, qualquer fase de pré-processamento, evitando, por exemplo, o grande armazenamento dos dados pré-computados. Apesar de o cálculo do termo *feedforward* em tempo de execução exigir um custo computacional adicional (maior do que o acesso direto aos dados armazenados pré-computados em [20]), esse custo adicional não compromete o desempenho e não é um problema no método proposto.

3.1. Simulação de reações

As reações que se pretende simular são de dois tipos: as flexíveis, que ocorrem em decorrência dos distúrbios inesperados; e as rígidas, que ocorrem em decorrência dos distúrbios esperados.

Na simulação principal, o controle *feedforward* recebido é adequado para seguir fielmente os movimentos capturados, mas, por não depender do estado atual do modelo principal, não é capaz de corrigir sua trajetória, mesmo para pequenos distúrbios externos. O controle *feedback* assume essa função e, através do uso de baixos ganhos, permite que as reações ocorram de maneira flexível.

É importante relembrar que o método proposto calcula os torques *feedforward* usando controladores *feedback* de altos ganhos, que têm acesso ao estado atual da simulação auxiliar (vide Figura 2). Assim, se a simulação auxiliar levar em conta os distúrbios externos inesperados, os torques, τ_{pl}, produzidos para corrigir a trajetória do modelo auxiliar encapsularão esses distúrbios e os conduzirão à simulação principal, já que os torques, τ_{pl}, são enviados à simulação principal como torques *feedforward* a serem aplicados ao modelo principal. No entanto, como os torques, τ_{pl}, são calculados por controladores de altos ganhos, as reações obtidas nas duas simulações (simulação auxiliar e simulação principal) serão rígidas e inflexíveis, o que é indesejável no caso de distúrbios externos inesperados. Para evitar esse efeito indesejável, é importante ter o cuidado de desconsiderar os distúrbios externos inesperados durante a simulação auxiliar.

Vale salientar que, além de impactos inesperados, existem também os impactos esperados, para os quais a reação natural é rígida. Por exemplo, quando alguém dá um soco, mantém os músculos do braço rígidos e, da mesma forma, quando alguém se defende de um soco com o braço, deve mantê-lo firme para impedir que o soco o atinja.

O controle *feedback* ligado ao modelo principal a ser simulado é um controle de baixos ganhos, que, portanto, é adequado a reações flexíveis. Assim, para simular reações rígidas seria necessário aumentar deliberadamente os valores dos ganhos no momento do impacto. Porém, essa estratégia suscita uma série de questões difíceis de serem tratadas, como, por exemplo:

1. Quais juntas deveriam ter seus ganhos aumentados?
2. De quanto seria o aumento de cada ganho?
3. Em que instante os valores dos ganhos deveriam retornar aos baixos valores originais?

Além disso, na prática, há situações em que tanto impactos inesperados quanto impactos esperados ocorrem simultaneamente. Realizar esses ajustes e tratar situações como essa seria trabalhoso.

O método aqui proposto permite que impactos esperados pelo modelo resultem em reações propositalmente rígidas e inflexíveis, sem necessidade de modificação dos ganhos, além de tratar eficazmente e de forma automática situações com impactos simultâneos. Para isso, é suficiente considerar os impactos esperados tanto na simulação principal quanto na simulação auxiliar. Na Figura 2, pode-se observar que todos os distúrbios externos são considerados na simulação principal, enquanto que na simulação paralela, apenas os esperados são levados em conta.

3.2. Controle feedback

Controladores *feedback* PD (*Proportional Derivative*) não-lineares são usados tanto na simulação auxiliar quanto na simulação principal para corrigir o erro entre o estado atual e o estado desejado do modelo, através da seguinte expressão:

$$\tau = I \cdot \left[k_s f(\theta_e)(\theta_d - \theta) - k_d (\dot{\theta}_d - \dot{\theta}) \right] , \qquad (1)$$

onde θ e θ_d são os ângulos atual e desejado da junta, $\dot{\theta}$ e $\dot{\theta}_d$ são as velocidades atual e desejada da junta, k_s e k_d são os ganhos dos termos proporcional e derivativo, I é o momento de inércia da cadeia de corpos afetados pela junta, $\theta_e = (\theta_d - \theta)$ e $f(\theta_e)$ é o fator não-linear definido em função de θ_e. τ é limitado para evitar instabilidades na simulação dinâmica.

Os valores de θ_d usados nas duas simulações são obtidos diretamente a partir dos movimentos capturados. Os valores de $\dot{\theta}_d$ podem ser estimados por diferenças finitas também a partir dos dados capturados. Entretanto, na simulação auxiliar, devido ao *feedback* de altos ganhos, esses valores podem ser simplesmente zerados sem comprometer a qualidade com que o modelo auxiliar segue os movimentos capturados. Já na simulação principal, devido ao *feedback* de baixos ganhos, a informação de velocidade desejada é importante e o método proposto permite que essa informação seja convenientemente obtida diretamente a partir da simulação auxiliar, já que o modelo auxiliar segue fielmente os movimentos capturados. Portanto, os valores de $\dot{\theta}_d$ adotados na simulação principal são as velocidades $\dot{\theta}$ da simulação auxiliar, que são mais compatíveis do que possíveis estimativas por diferenças finitas a partir dos dados capturados.

O uso do fator de inércia, introduzido por Zordan e Hodgins [22], permite que apenas um par de ganhos (k_s e k_d) seja especificado para todo o modelo, eliminando o processo trabalhoso de especificar ganhos para cada junta. Allen et al. [1] descrevem os detalhes do cálculo desse fator, o qual deve ser atualizado em cada passo da simulação. Um par de ganhos (k_s e k_d) para cada simulação deve ser definido pelo animador.

Observações acerca da influência da razão k_d:k_s sobre os movimentos levaram às seguintes conclusões: 1) se k_d:k_s for baixa, o modelo tende a oscilar nas proximidades da configuração desejada; e 2) se k_d:k_s for alta, o modelo tende a se mover vagarosamente, como se estivesse na água. Considerando essas observações, o ideal é que a razão k_d:k_s seja grande nas proximidades da configuração desejada, para evitar as oscilações; e pequena caso esteja longe da configuração desejada, para evitar que o movimento seja retardado de forma não natural.

Fattal e Lischinski [3] usam controladores PD não-lineares, em que a razão é atualizada modificando-se o termo k_d. No modelo aqui proposto, que também usa controladores PD não-lineares, o valor de k_d:k_s é atualizado modificando-se o termo k_s. Essa estratégia é mais intuitiva, uma vez que quando o modelo recebe um impacto, o que implica no afastamento da configuração desejada, a tendência é ele se tornar mais tenso, correspondendo ao aumento dos ganhos. Assim, o aumento de k_s reduz a razão k_d:k_s, como desejado.

Outra observação interessante é que, após receber um impacto, sistemas biológicos apresentam um pequeno atraso na reação devido ao tempo gasto no fluxo de informação ao longo dos neurônios e, principalmente, através das sinapses entre eles [20]. O modelo aqui proposto inclui esse atraso na definição do fator $f(\theta_e)$,

$$f(\theta_e) = \begin{cases} 1, & \text{se } |\theta_e| \le \lambda \\ \dfrac{|\theta_e|}{\lambda}, & \text{se } |\theta_e| \le M \cdot \lambda \\ M, & \text{caso contrário} \end{cases} , \qquad (2)$$

onde λ corresponde ao atraso na reação e M é o valor máximo permitido para o fator. θ_e é medido em radianos, λ deve ser positivo e M deve ser maior do que 1. Analisando a função, temos que: $f(\theta_e) \geq 1$; $f(\theta_e)$ é linear, para $\lambda \leq |\theta_e| \leq M.\lambda$; $|\theta_e|/\lambda = 1$, quando $|\theta_e|=\lambda$; e $|\theta_e|/\lambda = M$, quando $|\theta_e| = M.\lambda$. Em todos os testes realizados, os valores usados foram $\lambda = 0.1$ e $M = 5$.

3.3. Vantagens em relação ao uso de dinâmica inversa

Yin et al. [20] calculam o termo *feedforward* usando dinâmica inversa em uma fase de pré-processamento. No cálculo de dinâmica inversa, as acelerações são estimadas pela derivada segunda de *splines* cúbicas, geradas a partir dos dados capturados. Os valores resultantes da dinâmica inversa juntamente com os torques *feedback* de baixos ganhos, produzidos em tempo de execução, são usados na simulação física do modelo.

As vantagens do método proposto podem ser enumeradas como a seguir:

- Cálculo mais simples do termo *feedforward.* O cálculo feito pelo modelo proposto é exatamente igual ao cálculo dos torques *feedback*, só que em um modelo auxiliar.
- Consistência com o modelo principal. Os torques *feedforward* são mais compatíveis com o modelo principal, já que o modelo auxiliar é idêntico ao modelo principal e a técnica de simulação adotada é a mesma do modelo principal. Em contraste, o modo de cálculo por dinâmica inversa usa uma abordagem completamente distinta da simulação principal. Por exemplo, na dinâmica inversa, o movimento capturado é considerado nos cálculos, de forma indireta, através das acelerações que são estimadas a partir dele.
- Rastreamento de trajetórias mais gerais. Dinâmica inversa é apropriada para transformar trajetórias de movimentos realistas e detalhados (e.g. movimentos capturados) em funções apropriadas de força. Entretanto, quando a trajetória não corresponde a um movimento que respeita as leis da física ou é esparsa (e.g. keyframes), é complicado achar forças adequadas capazes de seguir uma aproximação da trajetória desejada. Por outro lado, controladores *feedback* são usados eficientemente para produzir torques apropriados para trajetórias mais gerais (inclusive trajetórias que não obedecem à física ou que são esparsas), tais como transições entre movimentos [18, 2, 22, 21] ou keyframes (controle de pose) [17, 21, 1]. Portanto, como o método proposto usa controladores *feedback* para obter os torques *feedforward*, ele é adequado e diretamente aplicável também a movimentos cinemáticos gerais, e não apenas a movimentos capturados.
- Custo computacional mais baixo. Em [20], o cálculo do termo feedforward por dinâmica inversa possui um custo computacional mais alto e não é realizado em tempo de execução. O método proposto realiza os cálculos em tempo de execução, permitindo tratar casos em que as trajetórias não podem ser previstas, tais como aquelas determinadas por controladores de mais alto nível, que dependem de sensores ou de impactos inesperados, para definir suas ações [2]. Essas trajetórias não podem ser pré-processadas com o intuito de obter torques *feedforward*, e, portanto, dinâmica inversa como usada em [20] não se aplica a esses casos.
- Tratamento eficiente em grafos de movimentos [6]. Reações também podem ser simuladas enquanto o modelo percorre grafos de movimentos. O uso de dinâmica inversa nesses casos requer que os torques *feedforward* sejam pré-computados para todos os movimentos capturados e para todas as transições possíveis, o que exige uma longa fase de pré-processamento e muita memória para armazenar os torques *feedforward*. Por sua vez, o método proposto calcula os torques *feedforward* em tempo de execução também nesses casos.
- Tratamento de reações a impactos previstos. O método proposto trata essas reações de forma bastante simples e direta, simplesmente considerando esses impactos nas duas simulações – simulação principal e simulação auxiliar. Em contrapartida, Yin et al. [20] não fornecem um tratamento especial para esse tipo de impacto. Além disso, aplicar a idéia proposta de considerar os impactos previstos a fim de que eles influenciem o termo *feedforward* resultante, usando dinâmica inversa, requer que os impactos sejam pré-processados e é trabalhoso incluí-los nos locais e instantes desejados.

4. Implementação

Um ambiente computacional de testes foi desenvolvido para examinar as reações do modelo, enquanto ele segue os movimentos capturados. Nesse ambiente, com o auxílio do teclado ou do mouse, o usuário pode tanto atirar objetos quanto aplicar

diretamente forças ou torques externos nos corpos selecionados do modelo, em tempo de execução. O usuário também pode escolher se a influência externa inserida será ou não esperada pelo modelo.

O ambiente computacional de testes foi desenvolvido em C++, usando: o motor dinâmico aberto ODE (Open Dynamics Engine) [15] para realizar a simulação física e a detecção de colisão; a biblioteca FOX toolkit [4] para criar a interface gráfica; e a biblioteca OpenGL para renderizar os quadros da animação.

As animações do modelo exibidas no ambiente são obtidas diretamente da simulação física principal, sem nenhum pós-processamento, e apresentam desempenho de tempo interativo. Em um PC com processador AMD Athlon de 1.8 GHz, 512 mb de memória RAM e placa de vídeo NVIDIA GeForce FX 5200 de 256 mb de memória, a taxa média de amostragem é de 10 fps, quando renderizando um quadro a cada 1/30 s de simulação. O intervalo de tempo considerado para cada iteração da simulação é de 0.0005 s, e um quadro é renderizado a cada 67 iterações (67*0.0005 s ≅ 1/30 s). Isso significa que para animar um segundo de simulação são necessários 3 segundos do tempo real.

Os movimentos capturados utilizados nos testes foram obtidos usando captura óptica em uma taxa de 120 fps. As posições dos marcadores ópticos foram convertidas nas posições e orientações globais correspondentes dos corpos do modelo [23], permitindo o cálculo direto dos ângulos das juntas. Para gerar os quadros intermediários necessários à simulação, os ângulos das juntas foram calculados usando interpolação linear esférica (*slerp*).

A realização correta de mudanças de coordenadas é fundamental em situações tais como a combinação dos torques *feedback* e *feedforward* na simulação principal ou a consideração dos impactos esperados na simulação auxiliar.

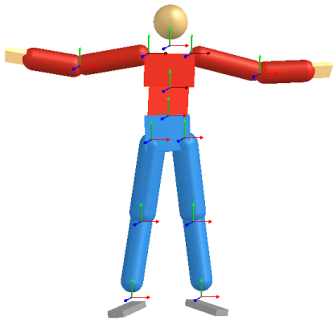

Figura 3. O modelo usado possui 39 DOF internos. Cada junta indicada possui 3 DOF.

5. Resultados

O modelo humanóide usado nos testes (Figura 3) é composto por 16 corpos rígidos e 13 juntas esféricas ("*ball-and-socket joints*"), totalizando 39 graus de liberdade internos e 6 graus de liberdade globais da "raiz" do modelo (pélvis). Os pulsos são fixos.

A flexibilidade do modelo e a qualidade com que o modelo segue os movimentos capturados dependem dos valores atribuídos aos ganhos (k_s e k_d) das duas simulações – simulação principal e simulação auxiliar. Os valores dos ganhos da simulação auxiliar devem ser definidos antes, pois os ganhos da simulação principal não interferem nessa escolha. Apesar de definida manualmente (i.e., por tentativa e erro) pelo animador, a escolha desses parâmetros não foi trabalhosa e, além disso, os valores escolhidos foram mantidos fixos em todos os testes. As relações entre os ganhos usados, nas duas simulações, foram: $k_s$principal = 0.05*$k_s$auxiliar e $k_d$principal = $k_d$auxiliar.

Para demonstrar a eficácia do método proposto, três testes foram realizados. Os vídeos acompanhando o artigo permitem visualizar e perceber melhor os movimentos e a qualidade do método.

No primeiro teste, vários objetos são lançados contra o modelo, enquanto ele segue os movimentos capturados. O modelo reage aos distúrbios externos e volta a seguir os movimentos capturados de maneira bastante natural. A Figura 4 mostra um exemplo em que o modelo reage a um impacto ocorrido no braço e, em seguida, a outro na cabeça. O modelo auxiliar não recebe os impactos e pode ser usado como referência para comparação. O movimento capturado original é mostrado nos vídeos acompanhantes.

O segundo teste compara a capacidade dos controladores *feedback* de baixos ganhos de seguir os movimentos capturados: sem usar o termo *feedforward* (Figura 5a); e usando o termo *feedforward* proposto, obtido a partir da simulação auxiliar (Figura 5b). O termo *feedforward* calculado usando o método proposto é eficaz e permite que os controladores *feedback* de baixos ganhos sejam capazes de seguir fielmente os movimentos capturados.

O terceiro teste mostra que o método proposto pode tratar facilmente distúrbios externos que são previstos pelo modelo, os quais devem implicar em reações rígidas e inflexíveis. A Figura 6a ilustra uma situação em que o modelo recebe um impacto inesperado na parte de trás da cabeça e, ao mesmo tempo, um impacto esperado na perna. O impacto esperado na

perna é considerado também no modelo auxiliar, como ilustrado na figura. Os ganhos não são alterados em nenhum momento. Na Figura 6b, os mesmos dois impactos são considerados inesperados, para efeito de comparação. No primeiro caso, a reação ao impacto na perna ocorre de maneira rígida, como se espera, enquanto que, no segundo caso, o joelho se mostra flexível.

O método proposto se mostrou bastante eficaz e natural na reação aos impactos em todas as partes do modelo, inclusive quando mais de um impacto ocorrem simultaneamente ou vários impactos ocorrem em seqüência. Além disso, o método permitiu tratar, de maneira simples e eficaz, distúrbios externos previstos.

6. Discussão e perspectivas

Para manter o equilíbrio do modelo nos testes realizados, forças e torques externos foram aplicados à "raiz" do modelo (pélvis) e aos pés (apenas quando em contato com o chão), para forçá-los a seguir os movimentos capturados. Essas forças e torques foram geradas por controladores *feedback* adicionais de altos ganhos, que foram usados nas duas simulações. Devido ao alto realismo presente em movimentos capturados, essa estratégia foi eficaz. Embora essa estratégia não seja fisicamente correta, o método proposto independe da estratégia de equilíbrio usada e, portanto, permite que outras estratégias realistas sejam empregadas. Entretanto, manter o equilíbrio durante movimentos atléticos, como os que foram apresentados neste trabalho, permanece um problema sem solução satisfatória. Yin et al. [21] propõem uma estratégia de equilíbrio promissora, porém específica, para movimentos em que os pés de apoio são trocados de modo periódico e uniforme (e.g. caminhar, correr).

Tanto limites nas juntas do modelo quanto contatos entre corpos pertencentes a um mesmo modelo apresentaram algumas instabilidades na simulação. Devido a essas instabilidades, limites nas juntas e contatos entre corpos pertencentes ao mesmo modelo não foram tratados nos testes realizados. Entretanto, essa é uma limitação da implementação, e não do método proposto. A superação dessas deficiências está sendo estudada e será apresentada no futuro.

É conveniente testar o método proposto de forma mais geral, acoplando-o, por exemplo, a sistemas capazes de realizar simples transições entre movimentos capturados ou de gerar e percorrer grafos de movimentos [6].

Seria interessante definir um critério geral para determinar se um impacto deve ou não ser esperado (percebido antecipadamente) pelo modelo. As forças de contato com o chão, por exemplo, já devem ser esperadas pelo modelo e, por isso, consideradas na simulação auxiliar.

Outra perspectiva interessante de utilização do método proposto seria permitir o modelo reagir antecipadamente para evitar que os impactos ocorram em partes mais vulneráveis do corpo, como em [25].

7. Conclusão

Este trabalho abordou o problema de simulação, de modo fácil e automático, de reações realistas a perturbações externas em movimentos capturados.

Tomando como base o trabalho de Yin et al. [20], o método proposto usa uma simulação auxiliar do modelo, contendo controladores *feedback* de altos ganhos, que permite que o termo *feedforward* seja obtido em tempo de execução de forma mais simples e mais geral, permitindo reações naturais a distúrbios externos, tanto inesperados como previstos pelo modelo. Além disso, o método proposto é adequado e diretamente aplicável a uma classe mais ampla de movimentos desejados, contendo tanto movimentos cinemáticos como dinâmicos, que exigem serem seguidos com boa precisão; e não apenas a movimentos capturados.

Ao treinar um tipo de movimento, sistemas biológicos armazenam informações mais abstratas, e não explicitamente os torques *feedforward*. Em muitos casos, o movimento desejado específico a ser realizado é determinado (imaginado) com detalhes pouco tempo antes de ser executado, baseando-se muitas vezes no *feedback* obtido do ambiente, através dos sentidos. Conseqüentemente, para esses casos, o pré-processamento *feedforward* da ação correspondente, usando as informações armazenadas no treinamento, também ocorre pouco tempo antes da execução da ação. Portanto, calcular o termo *feedforward* em tempo de execução mantém o método proposto mais próximo de sistemas biológicos do que pré-processar todas as ações possíveis que o modelo possa executar.

8. Agradecimentos

Os autores agradecem ao CNPq e à CAPES pelo apoio financeiro para a realização deste trabalho.

9. Referências

[1] Allen, B., Chu, D., Shapiro, A. & Faloutsos, P., "On the beat! Timing and tension for dynamic characters". Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Diego, CA, USA, 2007, pp. 239-247.

[2] Faloutsos, P., van de Panne, M. & Terzopoulos, D., "Composable controllers for physics-based character animation", Proceedings of ACM SIGGRAPH, Los Angeles, CA, USA, 2001, pp. 251-260.

[3] Fattal, R. & Lischinski, D., "Pose Controlled Physically Based Motion", *Computer Graphics Forum*, 2006, 25(4), 777-787.

[4] FOX toolkit, http://www.fox-toolkit.org/, 2008.

[5] Hodgins, J.K., Wooten, W.L., Brogan, D.C. & O'Brien, J.F., "Animating human athletics", Proceedings of ACM SIGGRAPH, Los Angeles, CA, USA, 1995, pp. 71-78.

[6] Kovar, L., Gleicher, M. & Pighin, F., "Motion graphs", *ACM Transactions on Graphics*, 2002, 21(3), pp. 473-482.

[7] Kry, P.G. & Pai, D.K., "Interaction capture and synthesis", *ACM Transactions on Graphics*, 2006, 25(3), pp. 872-880.

[8] Lee, J., Chai, J., Reitsma, P.S.A., Hodgins, J.K. & Pollard, N.S., "Interactive control of avatars animated with human motion data", *ACM Transactions on Graphics*, 2002, 21(3), pp. 491-500.

[9] Lee, J. & Lee, K.H., "Precomputing avatar behavior from human motion data", Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Grenoble, FRA, 2004, pp. 79-87.

[10] Magnenat-Thalmann, N. & Thalmann, D., "Virtual humans: thirty years of research, what next?", *The Visual Computer*, 2005, 21(12), pp. 997-1015.

[11] Neff, M. & Fiume, E., "Modeling tension and relaxation for computer animation", Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Antonio, Texas, USA, 2002, pp. 81-88.

[12] Ngo, J.T. & Marks, J., "Spacetime Constraints Revisited", Proceedings of ACM SIGGRAPH, Anaheim, USA, 1993, pp. 343-350.

[13] Pollard, N.S. & Zordan, V.B., "Physically based grasping control from example", Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, New York, NY, USA, 2005, pp. 311-318.

[14] Shapiro, A., Pighin, F. & Faloutsos, P., "Hybrid control for interactive character animation", Proceedings of IEEE Pacific Conference on Computer Graphics and Applications, Canmore, Alberta, CAN, 2003, pp. 455-461.

[15] Smith, R., Open dynamics engine, http://www.ode.org/, 2008.

[16] Van de Panne, M. & Fiume, E, "Sensor-actuator networks", Proceedings of ACM SIGGRAPH, Anaheim, USA, 1993, pp. 335-342.

[17] Van de Panne, M., "Parameterized gait synthesis", *IEEE Computer Graphics and Applications*, 1996, 16(2), pp. 40-49.

[18] Wooten, W.L. & Hodgins, J.K., "Simulation of leaping, tumbling, landing, and balancing humans", Proceedings of IEEE International Conference on Robotics and Automation, San Francisco, USA, 2000, pp. 656-662.

[19] Wrotek, P., Jenkins, O.C. & McGuire, M., "Dynamo: Dynamic data-driven character control with adjustable balance". Proceedings of ACM SIGGRAPH Video Games Symposium, Boston, USA, 2006, pp. 61-70.

[20] Yin, K., Cline, M.B. & Pai, D.K., "Motion perturbation based on simple neuromotor control models", Proceedings of IEEE Pacific Conference on Computer Graphics and Applications, Canmore, Alberta, CAN, 2003, pp. 445-449.

[21] Yin, K., Loken, K. & van de Panne, M., "SIMBICON: Simple biped locomotion control", *ACM Transactions on Graphics*, 2007, 26(3), article 105.

[22] Zordan, V.B. & Hodgins, J.K., "Motion capture-driven simulations that hit and react", Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Antonio, Texas, USA, 2002, pp. 89-96.

[23] Zordan, V.B. & Horst, N.V.D., "Mapping optical motion capture data to skeletal motion using a physical model", Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Diego, California, USA, 2003, pp. 245-250.

[24] Zordan, V.B., Majkowska, A., Chiu, B. & Fast, M., "Dynamic Response for Motion Capture Animation", *ACM Transactions on Graphics*, 2005, 24(3), pp. 697-701.

[25] Zordan, V.B., Macchietto, A., Medina, J., Soriano, M., Wu, C., Metoyer, R. & Rose, R., "Anticipation from Example", Proceedings of ACM Virtual Reality Software and Technology, Newport Beach, CA, USA, 2007, pp. 81-84.

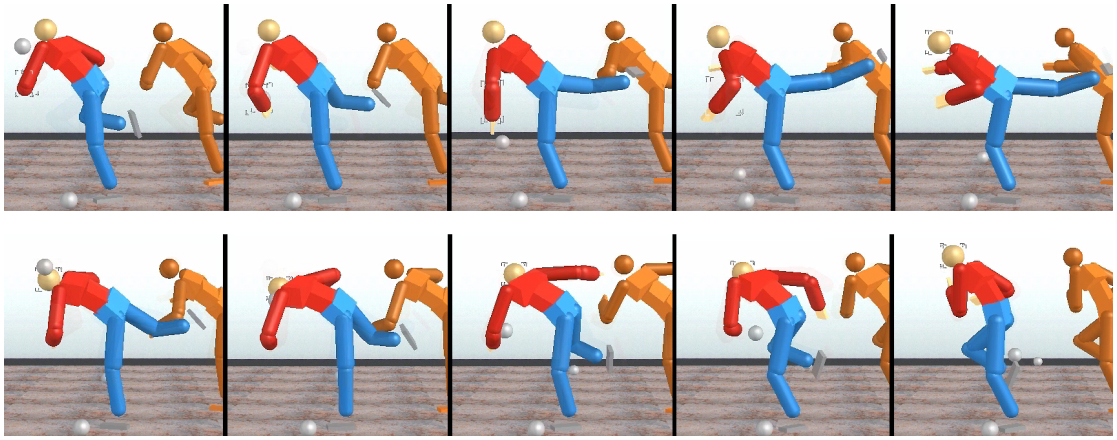

Figura 4. O modelo é atingido inesperadamente no braço e, em seguida, na cabeça, reagindo flexivelmente aos impactos e voltando a seguir o movimento capturado de maneira natural.

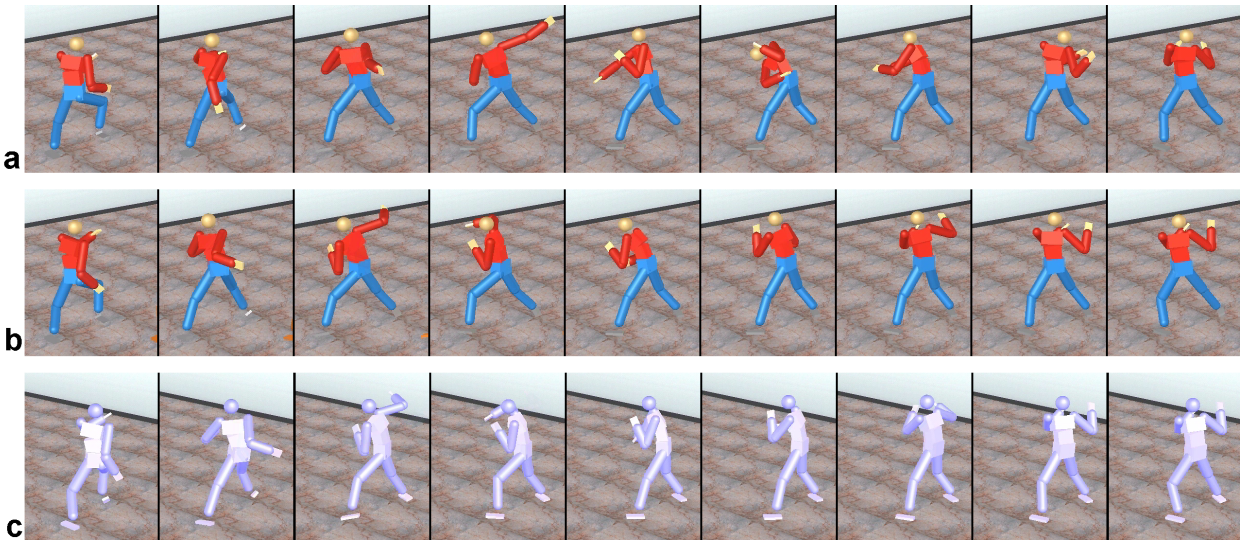

Figura 5. (a) Controle *feedback* de baixos ganhos sem o termo *feedforward*; (b) controle *feedback* de baixos ganhos com o termo *feedforward*; (c) movimento capturado original.

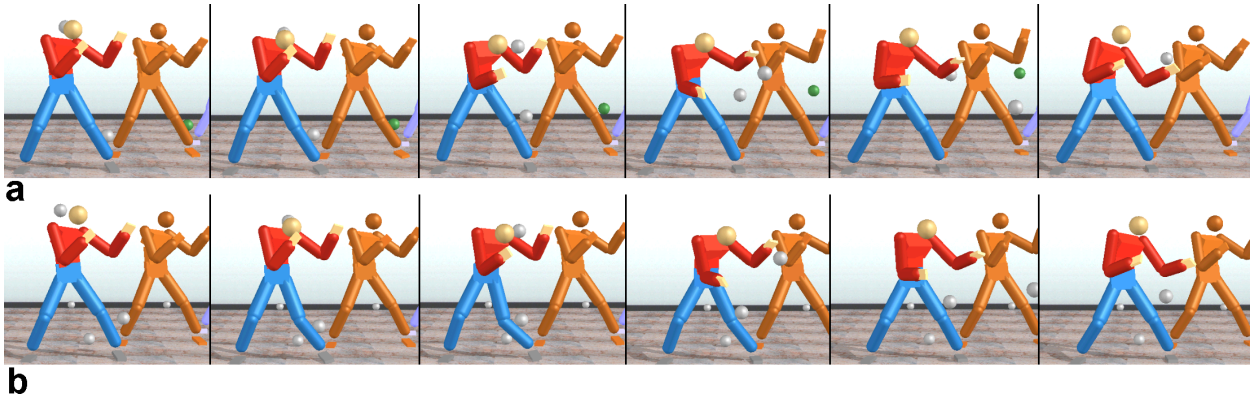

Figura 6. O modelo recebe, ao mesmo tempo, um impacto inesperado na parte de trás da cabeça e um impacto (a - esperado; b - inesperado) na perna.