Um Modelo de Sistema Nervoso para o Controle de Animação por Dinâmica Direta

Yuri Lenon Barbosa Nogueira

Dissertação apresentada ao
Mestrado em Ciência da Computação
Universidade Federal do Ceará

Orientador: Joaquim Bento Cavalcante Neto, Dr.
Co-Orientador: Creto Augusto Vidal, PhD.

Fortaleza, Ceará, Brasil
Abril 2007

Universidade Federal do Ceará
Mestrado em Ciência da Computação
Departamento de Computação
UFC

Dissertação de Mestrado

Um Modelo de Sistema Nervoso para o Controle de Animação por Dinâmica Direta

Yuri Lenon Barbosa Nogueira

Fortaleza – CE, 2007

Yuri Lenon Barbosa Nogueira

Um Modelo de Sistema Nervoso para o Controle de Animação por Dinâmica Direta

Dissertação apresentada ao Mestrado de Ciência da Computação da Universidade Federal do Ceará (UFC), como requisito parcial para a obtenção do título de Mestre em Ciência da Computação

Orientador:
Joaquim Bento Cavalcante Neto

Fortaleza – CE, 2007

A todos aqueles que
acreditaram neste trabalho.

AGRADECIMENTOS

Aos meus pais, Rita Alice e Mauro, que me deram oportunidades durante minha vida para que eu pudesse chegar até aqui. Ao meu irmão Oruam, que apresentou-me à ciência da computação, e toda a minha família. À minha namorada Camila, fonte de força para a realização deste trabalho e importante presença, e a sua família.

Aos amigos do laboratório, que se divertiram com as quedas dos modelos enquanto aprendiam a andar. Ao Gás Hélio e ao Pablo, pela ajuda com a montagem deste texto. Ao Diego, pela direção de audio do video demonstrativo. Aos amigos presentes na defesa: Gilvan, Ítalo, Levi, Markos, Rubens, Roberto e Rômulo. Ao Leonardo e ao Victor, que não puderam estar presentes e usaram o Skype, e a todos que, de alguma forma, me apoiaram, acreditaram e torceram pelo meu sucesso.

A todos os professores que tive durante minha vida, em especial aos meus orientadores Joaquim Bento e Creto Vidal. Ao professor Luiz Marcos, pelas sugestões que puderam engrandecer este trabalho após a defesa. À humanidade, por todo o conhecimento construído e compartilhado que fez parte deste trabalho e da minha formação na vida, em especial aos autores dos trabalhos referenciados na bibliografia.

Ao Mestrado em Ciência da Computação da Universidade Federal do Ceará, pela oportunidade, e à CAPES, pelo apoio financeiro.

RESUMO

A animação por dinâmica direta consiste em sintetizar os movimentos de um modelo, a partir da especificação de suas propriedades físicas (massa e momento de inércia), das condições de vínculo entre suas partes componentes, das condições de contato com outros corpos, e das forças que nele atuam. Essa abordagem tem a vantagem de gerar animações com realismo físico. O problema, que continua relevante como objeto de investigação, é o de controle do modelo: "**Que forças devem ser aplicadas ao modelo para gerar o movimento desejado?**".

A solução do problema proposto apresentada neste trabalho assume que o modelo estudado constitui-se de uma estrutura de corpos rígidos articulados cujos movimentos são gerados por atuadores internos, com suas forças definidas por um sistema nervoso. Com o uso de redes neurais artificiais e computação evolucionária, o controlador proposto é capaz de adaptar-se para controlar diferentes modelos articulados, e para gerar variados tipos de movimentos enquanto mantém a estabilidade mesmo quando há pequenas variações do terreno.

O modelo proposto possui, em seu núcleo, um gerador central de padrões (CPG - Central Pattern Generator) baseado em osciladores neurais, e o mesmo tem sua atividade regulada por módulos sensoriais, para permitir o equilíbrio da estrutura e estabilidade do movimento, respondendo às variações do ambiente.

Para a adaptação à estrutura articulada e aprendizagem de movimentos, o controlador possui ainda um módulo cognitivo, responsável pela busca dos parâmetros neurais, através de algoritmos genéticos, e das redes de retroalimentação (sensoriamento), com programação genética.

Resultados são apresentados em associação ao controle dos modelos humanóide, *cheetah*, sapo, *luxo* e *luxo*-2, sendo esses dois últimos iguais topologicamente, mas com variações nos tamanhos dos corpos e liberdade das juntas. Todos os modelos são testados em terreno plano e com rampa.

ABSTRACT

Direct dynamics animation consists of synthesizing the movements of a model from the specification of its physical properties (mass and moment of inertia), the conditions of bond between its contracting parties, the conditions of contact with other bodies and the forces that acts on it. This approaching has the advantage to generate animations with physical realism. The problem, that continues relevant as inquiry object, is the control of the model: "**What forces must be applied to the model to generate the desired movement?**".

The solution of the problem, presented in this work, assumes that the studied model consists of a structure of rigid link bodies whose movements are generated by internal actuators, with its forces defined by a nervous system. With use of artificial neural networks and evolutionary computation, the proposed controller is capable of adapting itself to control different articulated models, and to generate varied types of movements while it keeps the stability even with small variations of the terrain.

The presented model possesses, in its core, a Central Pattern Generator (CPG) based on neural oscillators, that has their activities regulated by the sensorial module, to allow the balance of the structure and stability of the movement, responding to environment variations.

For the adaptation to the articulated structure and learning of movements, the controller has a cognitive module, responsible for the search of neural parameters, through genetic algorithms, and the feedback networks (sensorial answers to environment variations), with genetic programming.

Results are presented related to the control of models humanoid, *cheetah*, frog, *luxo* and *luxo*-2, having these last two ones equal topologies, but with variations in the sizes of the bodies and freedom of the joints. All the models are tested in plain land and with slope.

Sumário

Agradecimentos ii

Resumo iii

Abstract iv

Sumário v

Lista de Figuras vii

Lista de Tabelas ix

1 Introdução 1
- 1.1 Motivação . . . 1
- 1.2 Animação de Modelos Virtuais . . . 2
  - 1.2.1 Representação . . . 2
  - 1.2.2 Classificação das Técnicas . . . 2
- 1.3 O Problema do Controle de Animação por Dinâmica Direta . . . 4
  - 1.3.1 Descrição . . . 4
  - 1.3.2 Controladores . . . 4
  - 1.3.3 Geração Manual de Controladores . . . 5
  - 1.3.4 Geração Automática de Controladores . . . 5
- 1.4 Solução Proposta . . . 6
- 1.5 Organização da Dissertação . . . 7

2 Controladores Inteligentes 8
- 2.1 Introdução . . . 8
- 2.2 Redes Neurais aprendizes de poses . . . 9
- 2.3 Representação Estímulo-Resposta . . . 10
- 2.4 As Redes de Sensores-Atuadores . . . 12
- 2.5 O Modelo Nervo-Músculo-Ósseo . . . 13

2.5.1 Descrição 13
2.5.2 O Sistema Nervoso 15
2.5.3 Auxílio de Computação Evolucionária 19
2.6 Estudo Comparativo 21

3 Um Modelo de Sistema Nervoso para o Controle de Animação por Dinâmica Direta 23
3.1 Introdução 23
3.2 O Sistema Músculo-Ósseo 24
3.3 O Sistema Nervoso 24
3.3.1 Visão Geral 24
3.3.2 Módulo Neural 25
3.3.3 Módulo Sensorial 28
3.3.4 Módulo Cognitivo 29
3.4 Considerações Finais 34

4 O Modelo Orientado a Objetos 36
4.1 Introdução 36
4.2 Classe RLBModel 36
4.3 Classe NervousSystem 38
4.4 Classe OscillatorUnit 38
4.5 Classe NeuralRhythmGenerator 38
4.6 Classe Sensor 39
4.7 Classe Individual 39
4.8 Classe Cognitive 40
4.9 Considerações Finais 41

5 Estudos de Casos 42
5.1 Considerações Gerais 42
5.2 Modelos Analisados 43
5.2.1 Modelo Humanóide 43
5.2.2 Modelo da *Cheetah* 44
5.2.3 Modelo do Sapo 45
5.2.4 Modelos *Luxo* e *Luxo*-2 46
5.3 Resultados 47

6 Conclusões 54
6.1 Avaliação dos Resultados 54
6.2 Trabalhos Futuros 55

Referências Bibliográficas 57

Lista de Figuras

1.1 À esquerda, o jogo The Sims. À direita, uma cena do filme Shrek. . . . 1
1.2 Estrutura de Corpos Rígidos Articulados representando um modelo humanóide. [9] 2
1.3 Controlador *PD*. 4
1.4 Animação dinâmica por Controladores. 5

2.1 Mapeamento de entrada e saída em modelo humano [29]. 9
2.2 Acima, os quadros-chave capturados para o comportamento de andar. Abaixo, o resultado alcançado pelas redes neurais [29]. 9
2.3 Representação Estímulo-Resposta [16]. 10
2.4 Solução estímulo-resposta construída manualmente [16]. 11
2.5 Alguns modos de locomoção usando SAN [30]. 12
2.6 Rede de Sensores-Atuadores [30]. 12
2.7 Modelo Nervo-músculo-ósseo. 14
2.8 Sistema Músculo-Ósseo. (a) Estrutura de Corpos Rígidos Articulados. (b) Geometria muscular. [9] 14
2.9 Oscilador neural com dois neurônios e sua atividade rítmica [12]. 15
2.10 Sistema Nervoso [9]. 15
2.11 Gerador Neural de Ritmo para locomoção bípede [28]. 16
2.12 Sistema Nervoso com o Gerador de Movimentos Discretos e Controlador de Postura [27]. 17
2.13 Sistema Nervoso com o Controlador de Estabilidade [18]. 18
2.14 Conexões motoras entre controlador e modelo bípede [23]. 18
2.15 Evoluindo a rede de retroalimentação [20]. 20

3.1 Modelo Nervo-músculo-ósseo. 23
3.2 Estrutura de Corpos Rígidos Articulados representando um modelo humanóide. [28] 24
3.3 Visão geral do modelo de sistema nervoso. 25
3.4 Atividade de um neurônio. (a) sem adaptação e (b) com adaptação. . . 26

3.5 Atividade de um oscilador neural com dois neurônios. (a) Neurônio 1, (b) Neurônio 2 e (c) sinal combinado. 27
3.6 Simulação com Gerador Neural de Ritmo sem sensoriamento. 28
3.7 Módulo Sensorial. 28
3.8 Cromossomo com codificação dos parâmetros. 30
3.9 Cromossomo com codificação de uma função *feed()*. 31
3.10 Processo de evolução. 32
3.11 Mesmo programa representado por uma árvore completa e uma incompleta. 33

4.1 Diagrama de Classes modelando o controlador proposto. 37

5.1 Estruturas de corpos rígidos articulados: (A) humanóide, (B) *cheetah* [3], (C) sapo, (D) *Luxo* e *Luxo*-2. 43
5.2 Resultado obtido para o modelo humanóide. À esquerda, o caminhar em terreno plano. À direita, em terreno com rampa. 49
5.3 Resultado obtido para o modelo *cheetah*. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa. 50
5.4 Resultado obtido para o modelo sapo. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa. 51
5.5 Resultado obtido para o modelo *luxo*. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa. 52
5.6 Resultado obtido para o modelo *luxo*-2. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa. 53

Lista de Tabelas

1.1 Resumo das Técnicas de Animação. 2

2.1 Conjuntos de Funções e Terminais [20]. 19
2.2 Comparação dos trabalhos baseados em CPG. 21

3.1 Conjuntos de Funções e Terminais. 31
3.2 Funções e Terminais utilizados pelo MC. 33

5.1 Parâmetros do Módulo Cognitivo. 42
5.2 Relações de Massa e Comprimento do humanóide. 44
5.3 Ângulos mínimos e máximos de cada junta do humanóide. (em *rad*) . . 44
5.4 Relações de Massa e Comprimento da *cheetah*. 45
5.5 Ângulos mínimos e máximos de cada junta da *cheetah*. (em *rad*) 45
5.6 Relações de Massa e Comprimento do sapo. 45
5.7 Ângulos mínimos e máximos de cada junta do sapo. (em *rad*) 46
5.8 Relações de Massa e Comprimento do *luxo*. 46
5.9 Ângulos mínimos e máximos de cada junta do *luxo*. (em *rad*) 46
5.10 Relações de Massa e Comprimento do *luxo*-2. 47
5.11 Ângulos mínimos e máximos de cada junta do *luxo*-2. (em *rad*) 47
5.12 Resumo das simulações. 47

1

Introdução

1.1. Motivação

A possibilidade de visualizar dados em forma de imagens na tela de um computador faz da Computação Gráfica uma das áreas mais importantes dentro da Ciência da Computação. Tais dados podem ser extraídos do mundo real, quando o objetivo é ter um modelo computadorizado do mesmo, ou podem ser gerados para exibição de imagens concebidas pela imaginação.

Modelar mundos no computador e com eles fazer simulações é uma das motivações de outra área importante: a Realidade Virtual. Tanto a criação de novos mundos para, por exemplo, ambientar jogos ou filmes (Figura 1.1), como a reprodução do mundo real para visitação de locais inacessíveis (seja por motivos econômicos, físicos ou de perigo), exigem fidelidade não só na geração da imagem, mas também nos movimentos dos objetos do ambiente virtual. Isso é importante para que esses mundos sejam convincentes àqueles que os observarem, criando um bom grau de envolvimento com eles.

Figura 1.1. *À esquerda, o jogo The Sims. À direita, uma cena do filme Shrek.*

A Animação por Computador é a subárea da Computação Gráfica responsável pela reprodução dos movimentos de modelos animados. A animação realística é essencial para alcançar o objetivo das aplicações já citadas e o presente trabalho traz um estudo sobre a movimentação de personagens virtuais humanóides.

1.2. Animação de Modelos Virtuais

1.2.1. Representação

Para a animação de modelos virtuais, geralmente usa-se uma representação simplificada, caracterizada por um modelo de "esqueleto". Os movimentos são primeiro calculados para esse modelo e, após alcançado o movimento desejado, pode-se "vestí-lo" com malhas deformáveis representando a pele ou roupas.

O esqueleto é uma estrutura de corpos rígidos hierarquicamente conectados por articulações. Uma definição matemática de sua animação é dada por Multon et al. [14]: Considerando-se $\theta = (\theta_1, ..., \theta_N)$ o conjunto dos parâmetros correspondentes a cada grau de liberdade da estrutura, chamado de "vetor estado", sintetizar um movimento do esqueleto consiste então em definir como o vetor estado muda através do tempo. A Figura 1.2 mostra um exemplo de uma estrutura articulada de um modelo humanóide, utilizada no trabalho de Hase et al. [9] para simulação do caminhar humano.

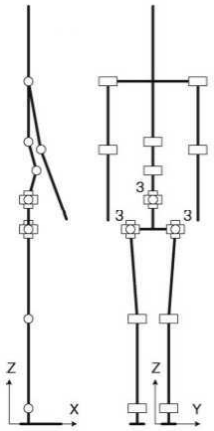

Figura 1.2. *Estrutura de Corpos Rígidos Articulados representando um modelo humanóide. [9]*

1.2.2. Classificação das Técnicas

Uma possível classificação para as técnicas de animação é dada por Pina et al. [21], que classifica as técnicas segundo mecanismo de replicação e mecanismo de modelagem. O primeiro reproduz movimentos e comportamentos de um ator real, enquanto o segundo é baseado em aproximações algoritmicas. (Tabela 1.1)

Abordagem por Replicação	Key-Framing
	Captura de Movimentos
Abordagem por Modelagem	Cinemática Direta/Inversa
	Dinâmica Direta/Inversa

Tabela 1.1. *Resumo das Técnicas de Animação.*

Abordagens por Replicação

No modelo de Captura de Movimentos, a animação do modelo é gerada a partir de dados capturados. Seja por tecnologia óptica ou magnética, posições e orientações de pontos localizados em um ator real são armazenados e passam por um tratamento algoritmico que faz a ligação entre o esqueleto real e a estrutura de corpos rígidos articulada, para adaptar os dados ao novo corpo. A animação gerada é uma réplica dos movimentos do ator real.

Na técnica de Key-Framing, o animador especifica os valores dos parâmetros em quadros-chaves, isto é, em alguns pontos na linha do tempo da animação. Os quadros intermediários são obtidos utilizando-se alguma técnica de interpolação nos parâmetros chaves como, por exemplo, interpolação linear.

Abordagem por Modelagem Cinemática

Na Modelagem Cinemática, o vetor estado é gerado sem preocupação com forças e torques. Existem duas direções a serem seguidas nos algoritmos que se utilizam dessa abordagem: Cinemática Direta e Cinemática Inversa. A primeira consiste em especificar os parâmetros do vetor estado através do tempo. Já a segunda, consiste em, dado um movimento objetivo especificado pelo animador, calcular a variação do vetor estado, gerando a animação. Existem ainda as técnicas híbridas, ou seja, que utilizam as duas direções, onde uma corrige ou "afina" o resultado da outra.

Os algoritmos de modelagem cinemática são geralmente usados para auxiliar o animador na geração de quadros chaves para a técnica de Key-Framing. No caso de modelos humanóides, por exemplo, baseados em conhecimentos empíricos ou biomecânicos do movimento humano, os algoritmos geram parâmetros em quadros-chaves que resultarão em animações realísticas.

Abordagem por Modelagem Dinâmica

Na modelagem Dinâmica, a animação surge a partir da aplicação da mecânica clássica. Assim como na cinemática, existem as abordagens por Dinâmica Direta e Dinâmica Inversa. Na primeira, a animação é gerada a partir de forças e torques fornecidos, enquanto que na segunda, essas forças e torques são calculados a partir de um dado movimento desejado.

A principal vantagem da abordagem dinâmica é a garantia de animações fisicamente realísticas. Assim, os movimentos gerados serão compatíveis com as situações às quais o personagem for submetido, como caminhar por terreno irregular, carregar peso e reagir a empurrões.

O alto custo computacional imposto por essa abordagem, entretanto, foi sua limitação por muito tempo. Porém, com os avanços nas técnicas numéricas e de hardware, simulações de física já são possíveis em tempo real. O motor ODE (Open Dynamics Engine - Motor Aberto de Dinâmica) [26] é um exemplo de implementação que permite a fácil criação de mundos obedecendo as leis da mecânica clássica.

A abordagem dinâmica pode ser utilizada também a posteriori, como ajuste da cinemática [14]. Após uma primeira fase onde obtém-se o movimento a partir do modelo cinemático, as restrições das leis da física são introduzidas através do uso da dinâmica inversa, corrigindo-se assim eventuais erros na animação prévia gerada.

1.3. O Problema do Controle de Animação por Dinâmica Direta

1.3.1. Descrição

A animação por dinâmica direta consiste em sintetizar os movimentos de um modelo, a partir da especificação de suas propriedades físicas (massa e momento de inércia), das condições de vínculo entre suas partes componentes, das condições de contato com outros corpos e das forças que nele atuam. O problema, que continua relevante como objeto de investigação, é o de controle do modelo: "**Que forças devem ser aplicadas ao modelo para gerar o movimento desejado?**".

1.3.2. Controladores

O objetivo dos controladores é sintetizar animações de personagens virtuais utilizando dinâmica direta. Em analogia com o ser humano, eles funcionam como a região do cérebro responsável por enviar os sinais para estimular os músculos (representados pelos atuadores) que, por sua vez, movimentam o esqueleto.

Os atuadores são responsáveis por aplicar torques computados por controladores de baixo nível do tipo derivativo-proporcionais (*Proportional-Derivative - PD*) (Figura 1.3). Esses controladores são representados por um conjunto mola-amortecedor do tipo angular e computam o torque com contribuições do deslocamento angular e da velocidade angular entre o estado atual e o estado desejado final dos corpos rígidos articulados.

Os controladores *PD* têm seu comportamento dinâmico definido pela seguinte expressão [30]:

$$T = k_p(\theta_d - \theta) - k_d\dot{\theta}, \tag{1.1}$$

onde T, k_p, k_d, θ, θ_d e $\dot{\theta}$ são, respecitivamente, o torque, as constantes da mola e do amortecedor, os ângulos atual e final e a velocidade relativa entre os dois corpos conectados pela junta.

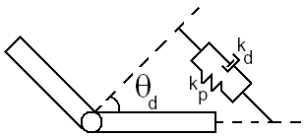

Figura 1.3. *Controlador PD.*

A animação passa a consistir, então, em criar controladores que gerem os devidos sinais para o movimento desejado. Os atuadores aplicam as respectivas forças nas articulações do modelo e, por conseqüência da simulação dinâmica, o personagem virtual

se moverá. Logo, a animação dinâmica por controladores pode ser modelada como mostra a Figura 1.4.

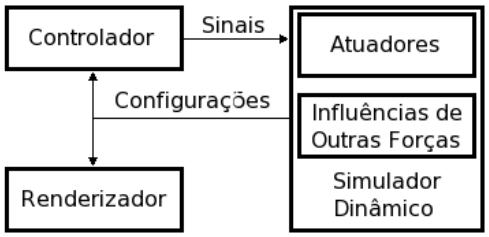

Figura 1.4. *Animação dinâmica por Controladores.*

1.3.3. Geração Manual de Controladores

O método de tentativa e erro é empregado para a geração manual de controladores. Seus parâmetros são procurados guiando-se por leis empíricas que determinam ângulos de alguns movimentos e, após encontrados, passam por uma fase de ajuste fino. Esse método gera controladores tão específicos a determinado comportamento ou personagem virtual, que a reutilização dos mesmos é dificultada.

1.3.4. Geração Automática de Controladores

Nesse método, os controladores são gerados pelo computador. O animador apenas define o movimento desejado (por exemplo, através de uma função objetivo) e o sistema automaticamente gera e otimiza os parâmetros dos controladores para que o objetivo seja alcançado. O trabalho de procurar os parâmetros é então passado para o computador. Logo, métodos de buscas como algoritmos genéticos, gradiente ascendente estocástico e têmpera simulada podem ser empregados.

Inteligência Artificial

Em animação dinâmica por controladores é inevitável buscar inspiração na robótica. Um exemplo bem sucedido nessa área é o robô humanóide Honda Asimo, que possui três níveis de controle [22]: 1) Seleção de fase de estados finitos e regras de perigo, 2) *Proportional-Integral-Differencial* (*PID*) *Controllers* sobre juntas individuais e 3) um comando global que compara forças de reação do chão. Controladores do tipo *PID* são os mais usados quando há perda de energia por atrito nas ligações.

Trabalhos que aplicam essa idéia em animação de humanos virtuais são discutidos no Capítulo 2. Também discute-se no Capítulo 2 os controladores adaptáveis ao ambiente, corpo e comportamento, ou seja, os controladores que, de certa forma, aprendem os sinais a gerar para cumprir um objetivo.

1.4. Solução Proposta

A solução do problema proposto apresentada neste trabalho assume que o modelo estudado constitui-se de uma estrutura de corpos rígidos articulados cujos movimentos são gerados por atuadores internos, com suas forças definidas por um subsistema nervoso.

O modelo proposto segue a representação nervo-músculo-óssea, descrita na literatura [8, 9, 15, 18, 27]. O núcleo do subsistema nervoso proposto possui os seguintes componentes:

- Um Gerador Central de Padrões (CPG), baseado em osciladores neurais, para geração de movimentos rítmicos, como ocorre na locomoção de animais.
- Um módulo sensorial, que regula a atividade do controlador, permitindo o equilíbrio da estrutura e a estabilidade do movimento, mesmo em terrenos irregulares ou com aplicação de forças externas.
- Um módulo cognitivo, baseado em algoritmo genético e programação genética, que permite a aprendizagem de movimentos e a adaptação à estrutura articulada. Esse módulo é responsável pela busca dos parâmetros do sistema nervoso, bem como pela organização topológica do CPG, de forma a gerar os sinais corretos para o controle do movimento desejado.

O modelo de sistema nervoso proposto neste trabalho reúne idéias apresentadas nos trabalhos descritos no Capítulo 2, com o propósito de obter um controlador de aplicações mais genéricas. Esse modelo baseia-se, principalmente, na adaptação e junção das várias idéias para se conseguir um modelo diferente e funcional para o problema.

São as contribuições dadas com esta solução:

- Definição de uma modularização para o modelo nervo-músculo-ósseo;
- Compilação de diversos atributos de modelos propostos em trabalhos anteriores, ficando mais genérico e de maior flexibilidade;
- Definição do Módulo Cognitivo e sua interação com os outros módulos, permitindo maior automatização na geração da animação e tornando indireta a dependência entre o comportamento e a topologia do CPG.

Como resultado principal, tem-se um modelo de controlador capaz de gerar, automaticamente, movimentos de locomoção adequados em diversas estruturas de corpos rígidos articulados. O modelo também é capaz de manter a estabilidade do movimento e equilíbrio da estrutura, mesmo quando há pequenas alterações no ambiente, como terrenos irregulares, reagindo aos estímulos recebidos pelos sensores. No Capítulo 5 são mostrados os resultados com 5 corpos: humanóide, *cheetah*, sapo, *luxo* e *luxo*-2. Todos foram testados em terrenos planos e com rampa.

1.5. Organização da Dissertação

Os capítulos restantes estão estruturadas da seguinte maneira. No Capítulo 2, apresentam-se as propostas de solução encontradas na literatura. No Capítulo 3, é feita uma breve exposição do subsistema músculo-ósseo e descrevem-se o subsistema nervoso e seus componentes. No Capítulo 4, alguns detalhes interessantes de implementação do controlador são discutidos. No Capítulo 5, estudos de casos ilustram a solução proposta. Finalmente, no Capítulo 6, apresentam-se as conclusões sobre o trabalho.

2

Controladores Inteligentes

2.1. Introdução

A utilização de controladores para resolver o problema de calcular as forças que devem ser aplicadas a um modelo articulado de modo a produzir movimentos desejados não é uma proposta nova, mas continua sendo um tópico de pesquisa bastante ativo [4, 19, 20, 22].

Este capítulo é dedicado ao estudo de trabalhos relacionados ao tópico de controladores inteligentes, grupo no qual se inclui o controlador proposto no Capítulo 3. Neste trabalho, um controlador inteligente é definido como aquele capaz de aprender um movimento e adaptar-se ao ambiente e ao corpo. Tais controladores têm sua inspiração em estudos de áreas como Inteligência Artificial, Robótica, Neurofisiologia, Psicologia, Biomecânica, entre outras.

São aspectos observados nos controladores inteligentes:

- Geradores Centrais de Padrão: Foi mostrado que os animais possuem geradores centrais de padrão [22], os CPGs (*Central Pattern Generator*), responsáveis por enviar os sinais básicos dos movimentos.

- Redes Neurais Artificiais (RNAs): As RNAs são usadas nas implementações dos CPGs em alguns controladores. Essas mesmas redes podem ser também utilizadas para aprendizagem supervisionada dos movimentos.

- Sensoriamento: Um componente necessário aos controladores inteligentes para realizar a adaptação ao meio é a entrada sensorial. Ela é responsável pela interação entre ambiente e controlador, possibilitando que o mesmo receba estímulos e reaja respondendo a eles. No caso dos CPGs, as atividades são reguladas pelos sensores, que podem disparar ou mudar fases de locomoção. [22]

- Otimização: Técnicas de otimização, como gradiente ascendente estocástico e computação evolucionária, são utilizadas para busca de parâmetros do controlador, possibilitando a aprendizagem de movimentos.

Neste capítulo são mostrados trabalhos que se utilizam das idéias apresentadas nesta seção para a construção de controladores.

2.2. Redes Neurais aprendizes de poses

Taha et al. [29] propôs uma técnica de Key-Framing utilizando abordagem dinâmica, conseguindo, assim, maior realismo na animação. Essa faz uso de Redes Neurais supervisionadas de retropropagação para aprender como controlar o modelo através de poses em quadros-chaves.

As informações necessárias para identificar um movimento são as seguintes:

> I = {Índice do comportamento, índice do subsistema, velocidade, posição do ciclo}

Onde Índice do comportamento é um valor único que identifica o movimento (0 - Esperar, 2 - Andar, 4 - Correr), a posição do ciclo diz em que porcentagem do mesmo aquela pose corresponde e o subsistema é uma parte da estrutura articulada (perna esquerda, perna direita, etc.). As duas últimas informações são obtidas através das coordenadas das juntas. Tem-se como saída: forças, velocidades, deslocamento das juntas e acelerações (Figura 2.1).

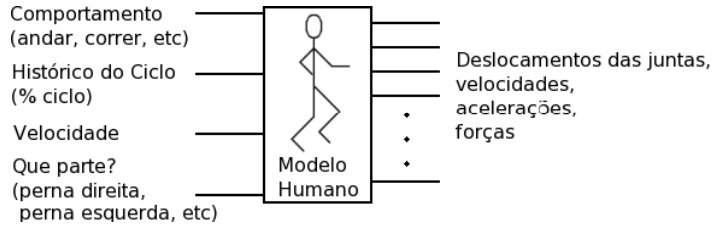

Figura 2.1. *Mapeamento de entrada e saída em modelo humano [29].*

A rede neural é então treinada para fazer esse mapeamento, utilizando-se a técnica de retropropagação em rede completamente conectada. O treinamento acaba quando o erro entre as poses dos quadros-chave e aquelas alcançadas pelo controlador chegar a níveis aceitáveis, usando-se distância euclidiana para o cálculo desse erro. A Figura 2.2 mostra o resultado da aplicação dessa técnica.

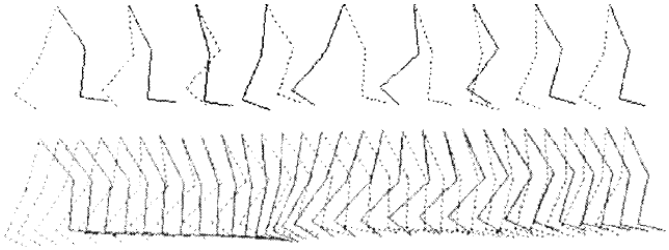

Figura 2.2. *Acima, os quadros-chave capturados para o comportamento de andar. Abaixo, o resultado alcançado pelas redes neurais [29].*

2.3. Representação Estímulo-Resposta

Skinner [25] sugeriu que todo comportamento em um animal é resultado da combinação de reflexos condicionados, ou seja, para um dado estímulo recebido do ambiente, o animal responde com um certo comportamento, o qual pode ser um movimento.

Ngo e Marks [16] utilizam-se dessa mesma idéia para a síntese de movimentos. Foi demonstrado como regras de estímulo-resposta poderiam gerar animações fisicamente realistas. Os principais conceitos nessa representação são: variáveis de sentido, estímulos, respostas e avaliação da solução.

Variável de sentido é uma função nos reais que mapeia as informações recebidas do simulador físico em um ponto no espaço dos sentidos, condição para uma ação. A função estímulo mapeia um elemento no espaço dos sentidos em uma resposta, ou seja, uma ação. A Figura 2.3 ilustra o algoritmo para a representação Estímulo-Resposta.

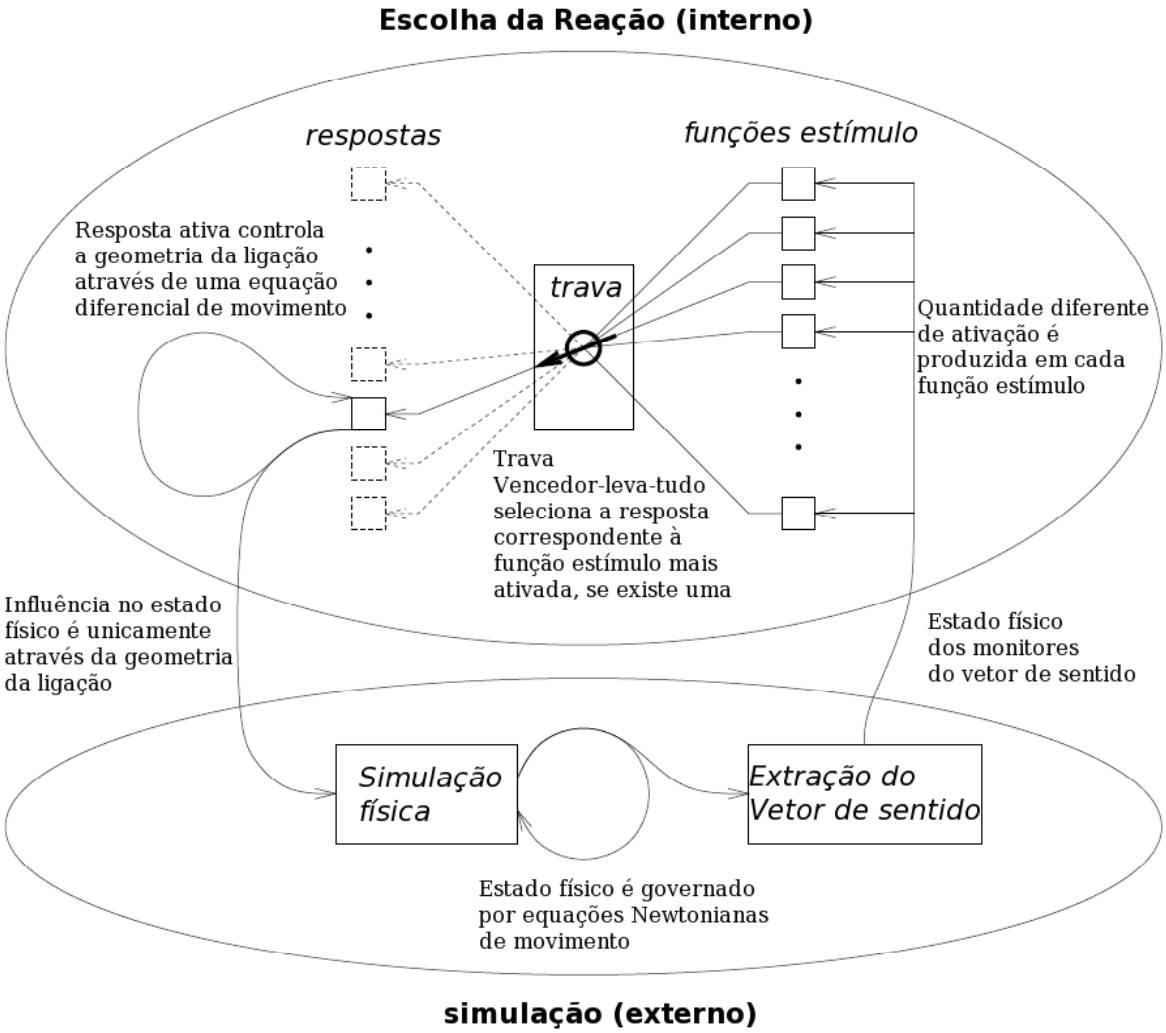

Figura 2.3. *Representação Estímulo-Resposta [16].*

Os passos seguidos para a representação Estímulo-Resposta a cada instante de tempo são, então, os seguintes:

1. Computa o ponto no espaço do sentidos;
2. Seleciona a resposta correspondente à função estímulo que teve o maior valor escalar naquele ponto ou, se negativo, escolhe a mesma do passo anterior;
3. Emprega a resposta selecionada para determinar a configuração interna no próximo passo;
4. Passa os valores para o simulador físico computar a configuração externa, isto é, da estrutura articulada, no passo seguinte.

Um exemplo ilustrativo é a animação de uma pessoa pulando o mais alto possível. Apenas duas variáveis de sentido são empregadas: a posição vertical e a velocidade do centro de massa do modelo.

A Figura 2.4 mostra as regiões das funções segundo as duas variáveis sensitivas empregadas para alcance do objetivo. Esta solução foi construída manualmente, mas pode-se usar técnicas de otimização (no caso, algoritmo genético) para geração automática.

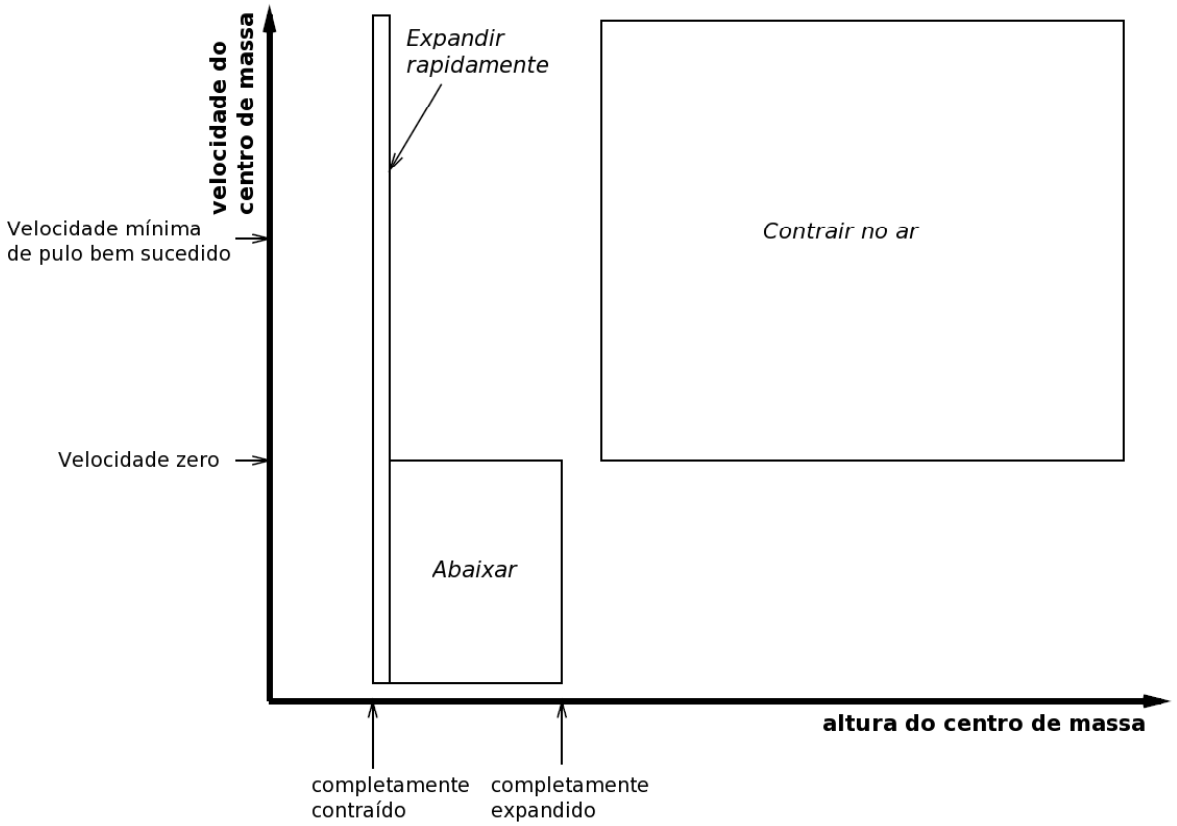

Figura 2.4. *Solução estímulo-resposta construída manualmente [16].*

As regiões sensitivas das funções estímulos e suas respostas associadas são:

- Expandir – Se o centro de massa está baixo, expandir rapidamente irá impulsionar a pessoa para cima;
- Abaixar – Se o centro de massa está muito alto para expansão, ele deve primeiro abaixar-se;
- Contrair no ar – Se o centro de massa está muito alto e a pessoa não consegue tocar no chão para expandir, então contrair-se fará com que seja alcançada uma maior altura do centro de massa.

2.4. As Redes de Sensores-Atuadores

As redes de Sensores-Atuadores (SANs - *Sensor-Actuator Networks*) foram propostas por van de Panne e Fiume [30] para o controle de animação dinâmica. A principal vantagem dessa abordagem é a geração de vários tipos de movimentos independente da criatura virtual a ser animada (Figura 2.5). Apenas com o objetivo de, por exemplo, alcançar a maior distância e um cálculo simplificado do gasto de energia, locomoções adaptadas à estrutura modelada de corpos rígidos são geradas.

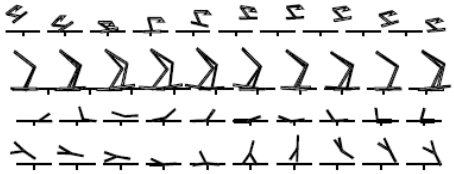

Figura 2.5. *Alguns modos de locomoção usando SAN [30].*

Uma SAN, ilustrada na Figura 2.6, liga sensores a atuadores através de uma rede de conexões ponderadas. Os sensores são binários, isto é, produzem o valor 1 se ligados ou 0 caso contrário, podendo ser de toque, de ângulo, de visão ou de tamanho. Os atuadores são associados a controladores PD nas articulações.

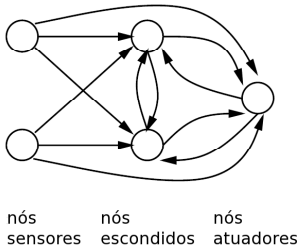

Figura 2.6. *Rede de Sensores-Atuadores [30].*

A rede consiste de nós sensores, nós escondidos e nós atuadores (Figura 2.6). Os primeiros têm seus valores nos sensores associados e são completamente conectados

aos outros dois. O número de nós escondidos geralmente é escolhido para ser igual ao número de nós sensores. Para cada nó atuador, tem-se um atuador associado.

Um nó soma os valores em suas entradas ponderadas, semelhante à função executada em redes neurais, e dá como saída 1 se a soma é positiva e 0 caso contrário. Os valores dos pesos pertencem a um intervalo inteiro, por exemplo, [-2,2]. Um nó atuador faz uso direto do somatório em suas entradas para determinar o valor do ângulo no atuador, através de mapeamento linear. É ainda implementado um atraso que tem controle por duas constantes, k_1 e k_2, dando ao nó um comportamento de oscilador, assemelhando a rede aos CPGs.

As constantes de atraso devem ser escolhidas segundo à duração do ciclo esperado da locomoção. Por exemplo, as constantes em um elefante devem ser muito maiores do que em um rato. Tipicamente, bons resultados podem ser obtidos com $1/k_1 = -1/k_2 = 0,25T_c$, onde T_c é a duração esperada de um ciclo.

Métodos de busca são aplicados para atribuir os valores dos pesos das conexões, que definirão o comportamento da criatura. Para tal, utiliza-se como função de avaliação, no caso da locomoção, a distância alcançada pelo modelo controlado pela rede em uma determinada quantidade de tempo: $f_{aval} = |x(t_{final})|$. Aqueles que geram os maiores valores, são os mais interessantes. Pode-se também adicionar como condição, só serem válidos os movimentos sem quedas, considerando-se $f_{aval} = 0$ quando isso ocorrer.

A busca é feita em duas fases: a primeira de procura dos pesos e a segunda de afinação para os melhores resultados. A avaliação é feita escolhendo-se valores aleatórios e testando-os através da simulação do funcionamento do controlador. São escolhidos os que obtiverem maior valor de f_{aval}. O tamanho do espaço de busca é n^w, onde n é a quantidade de valores que uma conexão pode assumir (5, no exemplo do intervalo [-2,2]) e w é o número de conexões ponderadas. Logo, é inviável testar todas as alternativas e, por esse motivo, utilizam-se métodos de busca como Têmpera Simulada e Gradiente Ascendente Estocástico.

Na segunda fase, regula-se parâmetros dos nós, dos sensores e dos atuadores. São feitos pequenos ajustes nos valores k_1 e k_2 que controlam os atrasos dos nós da rede, nos valores limiares para considerar um sensor ligado, nas constantes elásticas e de amortecimento e nos ângulos máximos e mínimos dos atuadores.

Embora não seja especificada como tal, poder-se-ia classificar uma SAN como um controlador baseado no modelo Nervo-Músculo-Ósseo (Seção 2.5). Um estudo mais aprofundado sobre isso é recomendado, em especial devido à obtenção de resultados semelhantes ao controlador proposto no Capítulo 3. Um estudo comparativo é feito na Seção 2.6.

2.5. O Modelo Nervo-Músculo-Ósseo

2.5.1. Descrição

Uma série de trabalhos [9, 27, 28, 8, 18] explora a representação nervo-músculo-óssea, que engloba os sistemas nervoso, muscular e ósseo em interação entre si e com o ambiente, para geração de animações do caminhar humano. Esses três sistemas são mo-

delados, respectivamente, como os controladores, os atuadores e a estrutura de corpos rígidos articulados. Uma generalização simplificada do modelo nervo-musculo-ósseo é mostrada na Figura 2.7.

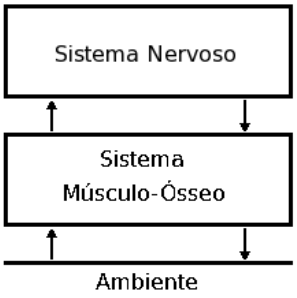

Figura 2.7. *Modelo Nervo-músculo-ósseo.*

Nos trabalhos que seguem o modelo Nervo-Músculo-Ósseo, o sistema ósseo é especificado por uma estrutura de corpos rígidos articulados, enquanto o sistema muscular é representado por atuadores nas articulações do esqueleto. Hase et al. [9] propõe que, quanto mais detalhadas forem as representações dos sistemas muscular e ósseo, mais realística será a movimentação do humano virtual. Em seu trabalho, modelos precisos (em relação aos sistemas biológicos humanos) desses sistemas são especificados e então controlados por um sistema nervoso, especificado na subseção 2.5.2. É possível, porém, conseguir resultados bastante satisfatórios com modelos mais simples, como observado em outros trabalhos [28, 27]. A Figura 2.8 mostra a especificação para os sistemas muscular e ósseo feita por Hase et al. [9] para um modelo humanóide.

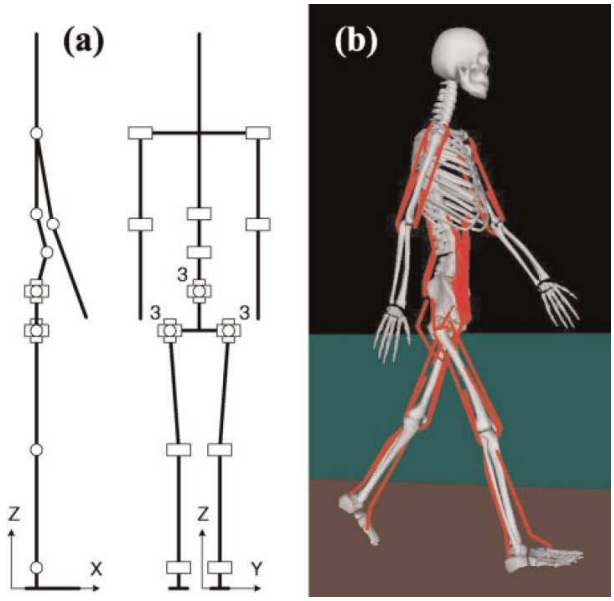

Figura 2.8. *Sistema Músculo-Ósseo. (a) Estrutura de Corpos Rígidos Articulados. (b) Geometria muscular. [9]*

2.5.2. O Sistema Nervoso

Sistemas Nervosos baseados em Osciladores Neurais

Os osciladores neurais são a representação do CPG. Trata-se de uma rede neural formada por neurônios mutuamente inibitórios (Figura 2.9) capaz de gerar oscilações sustentáveis e auto-adaptação, isto é, de regular sua freqüência oscilatória.

Um estudo completo do funcionamento dos osciladores neurais é feito por Matsuoka [12]. No Capítulo 3, é feita uma descrição mais detalhada, pois o modelo proposto neste trabalho tem seu CPG baseado nesse tipo de rede neural.

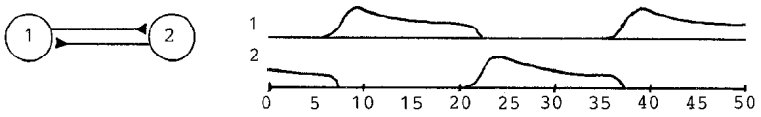

Figura 2.9. *Oscilador neural com dois neurônios e sua atividade rítmica [12].*

Nos trabalhos com sistemas nervosos baseados em Osciladores Neurais, cada dois neurônios ligados por conexões mutuamente inibitórias controlam, respectivamente, os movimentos de flexão e extensão associados ao grau de liberdade de rotação de uma junta da estrutura articulada. A Figura 2.10 mostra o sistema nervoso proposto por Hase et al [9].

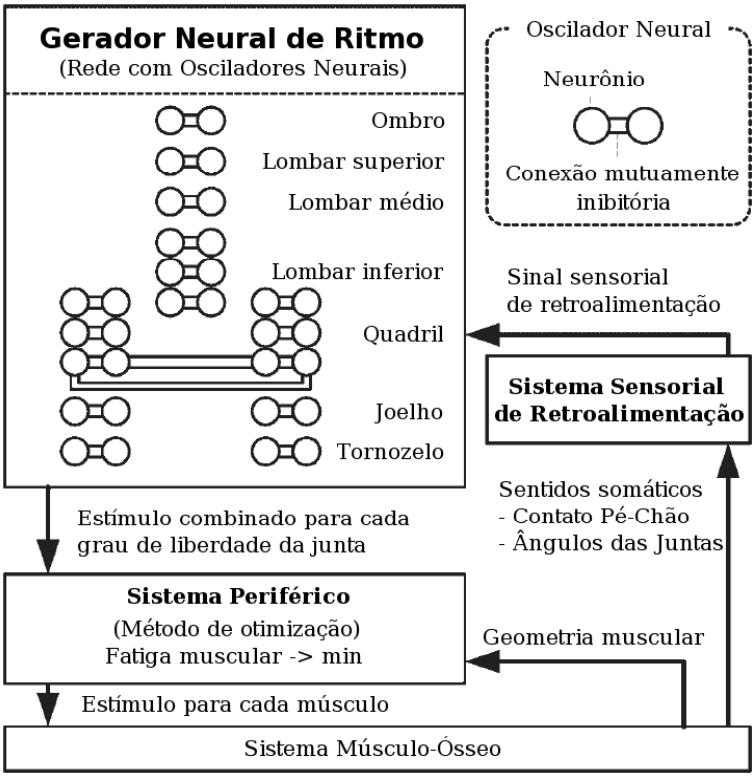

Figura 2.10. *Sistema Nervoso [9].*

O núcleo desse sistema nervoso é chamado de Gerador Neural de Ritmo (GNR), por ser baseado nos osciladores neurais. As atividades do GNR são controladas pelo Sistema Sensorial de Retroalimentação, recebendo sinais somáticos, como os ângulos das juntas e o contato pé-chão, corrigindo o ritmo para locomoções em terrenos acidentados, por exemplo. A função que trabalha com esses sinais não é bem definida matematicamente, embora, no trabalho de Hase et al. [9], haja uma tentativa de formalização. Uma maior discussão nesse aspecto será feita no Capítulo 3. O Sistema Periférico é responsável pela otimização dos parâmetros internos da rede e é discutida na subseção 2.5.3.

O movimento gerado por sistemas nervosos baseados em osciladores neurais depende da topologia da rede geradora de ritmo. Taga et al. [28] propôs uma rede específica para o controle do caminhar de um humanóide (Figura 2.11), com os parâmetros escolhidos manualmente e topologicamente consistente com o modelo hipotético para a rede espinhal de locomoção proposta por Grillner [5]. Seus resultados são bastante satisfatórios, mesmo com o sistema músculo-ósseo modelado de forma simplificada.

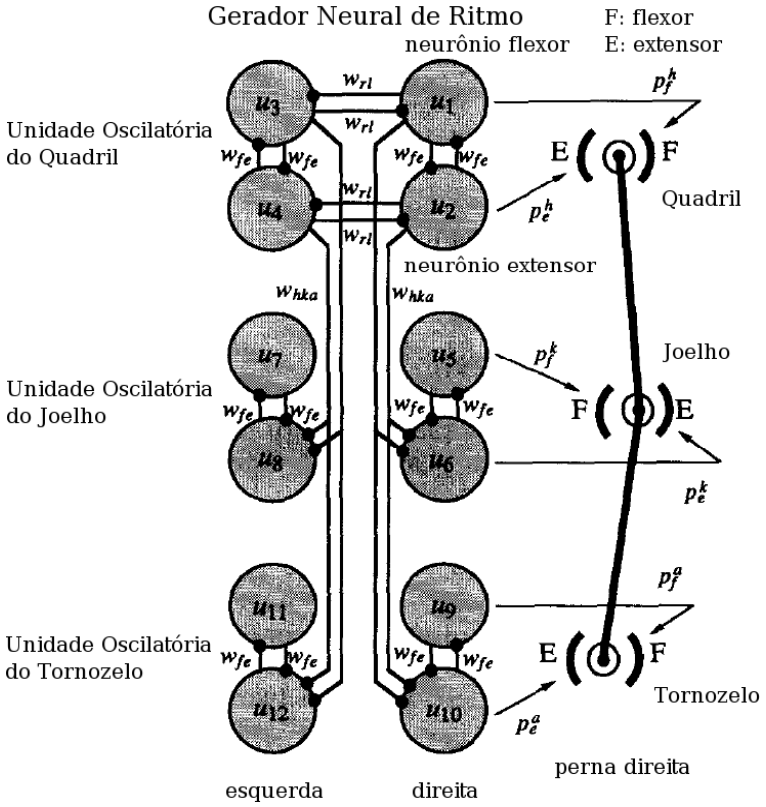

Figura 2.11. *Gerador Neural de Ritmo para locomoção bípede [28].*

Taga [27] propõe ainda o Gerador de Movimentos Discretos (DM) e o Controlador de Postura (PC). O primeiro é um módulo no sistema neural que recebe informações de sensores visuais e envia sinais para músculos específicos visando a modificação do padrão de locomoção para ultrapassagem de obstáculos. O PC é responsável pela manutenção do equilíbrio.

A visão geral do funcionamento do sistema nervoso proposto por Taga [27] é mos-

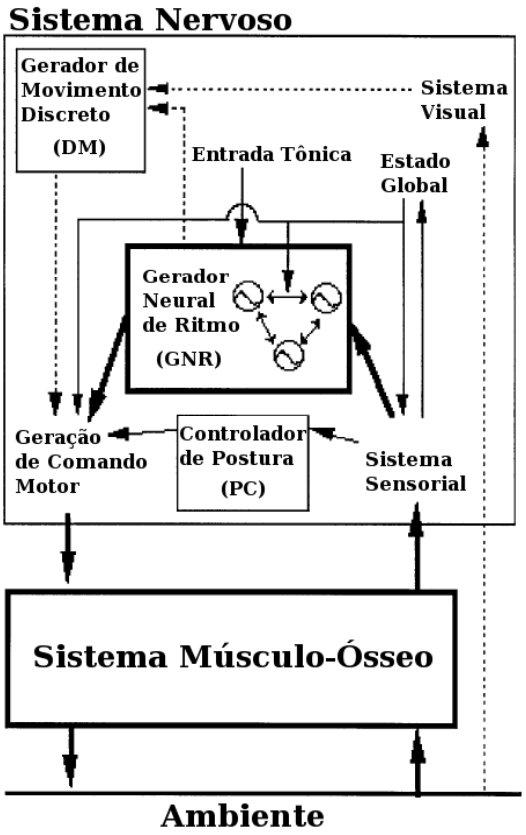

Figura 2.12. *Sistema Nervoso com o Gerador de Movimentos Discretos e Controlador de Postura [27].*

trado na Figura 2.12. Nesse modelo, os módulos DM e PC enviam sinais específicos direto para as juntas, de acordo com a necessidade, corrigindo o ritmo básico gerado pelo GNR. Com as informações recebidas do Sistema Visual, de distância e altura do obstáculo a ultrapassar, são feitas modificações no movimento pelo DM:

1. Modulação do tamanho do passo quando aproxima-se de um obstáculo; e
2. Modificação da trajetória dos membros enquanto o ultrapassa.

Trabalho semelhante foi proposto por Ni e Kato [18], introduzindo o Controlador de Estabilidade (CE), que imita a habilidade humana de auto-balanceamento, conseguindo assim maior estabilidade e alcançando, também, o objetivo de ultrapassagem de obstáculos. O torque do tornozelo é utilizado pelo CE para controlar a velocidade do centro de gravidade do corpo. A visão geral do modelo é mostrada na Figura 2.13.

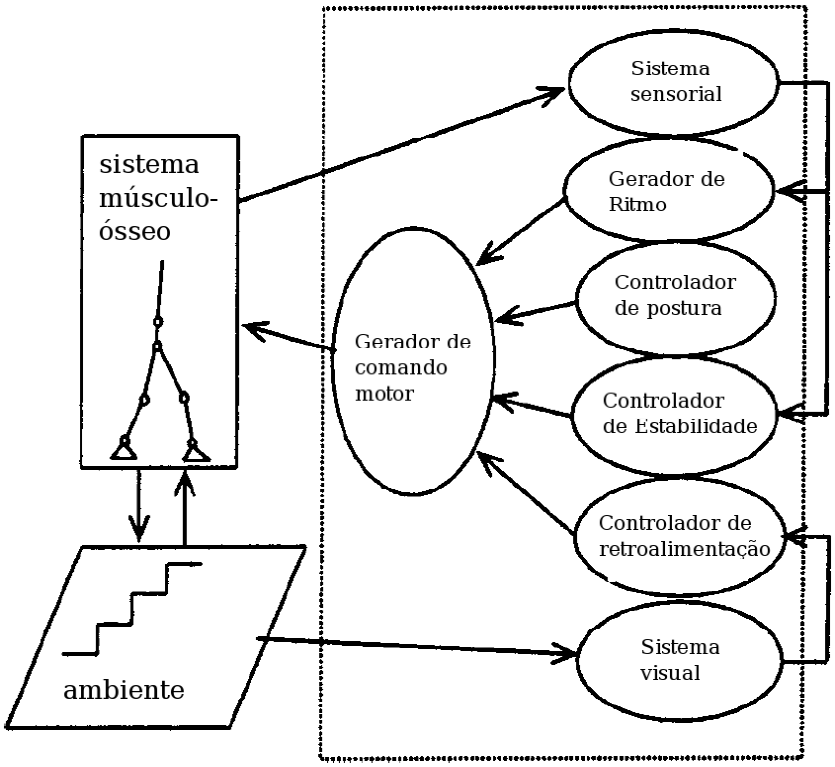

Figura 2.13. *Sistema Nervoso com o Controlador de Estabilidade [18].*

Outros Sistemas Nervosos

Embora não seja especificado como tal, o trabalho de Reil e Husbands [23] também pode ser classificado como baseado no modelo nervo-músculo-ósseo. A geração de sinais, entretanto, é feita por uma rede neural chamada de neurocontroladores recorrentes, composta de neurônios completamente interligados por conexões ponderadas. A Figura 2.14 mostra a arquitetura da rede e a relação entre os neurônios e os graus de liberdade. As conexões contínuas controlam as oscilações no eixo lateral e as tracejadas, no sagital.

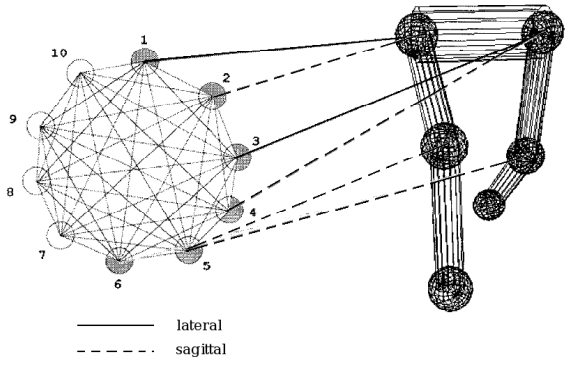

Figura 2.14. *Conexões motoras entre controlador e modelo bípede [23].*

O controle sensorial nessa rede é limitado, tendo apenas sido sugerido uma alteração de sinais através de um sensor de "audição", realizado por um processamento simples, para o modelo seguir uma fonte sonora.

Outra rede que também pode ser utilizada como gerador central de padrões no modelo nervo-músculo-ósseo, como já discutido, é a SAN, exposta na Seção 2.4.

2.5.3. Auxílio de Computação Evolucionária

Diversos trabalhos utilizam algoritmos evolucionários para busca dos parâmetros de seus controladores, conseguindo assim, adaptação ao modelo controlado ou um melhor ajuste do movimento. Nagano et al. [15] e Sellers et al. [24] utilizam Computação Evolucionária em seus trabalhos baseados no modelo nervo-músculo-ósseo para simular e predizer a locomoção do *Australopithecus afarensis*.

Ok e Kim [20] e Miyashita et al. [13] propuseram o uso de programação genética para a busca das redes de retroalimentação, pois as mesmas são definidas como funções que têm como entradas informações sensoriais e saídas um valor para alterar os sinais neurais. Os parâmetros e a topologia dos osciladores neurais foram considerados pré-definidos nesses trabalhos e possuem doze neurônios. Uma especificação mais detalhada das redes de retroalimentação é mostrada no Capítulo 3.

A Tabela 2.1 mostra os conjuntos de funções e terminais utilizados por Ok e Kim [20], baseada nos conjuntos de Miyashita et al. [13]. Os terminais são obtidos através dos sensores do modelo. Foi utilizada Programação Genética com Funções Definidas Automaticamente (ADF - *Automatically Defined Functions*).

Conjunto de Funções	+, -, *, exp, log
	$H(x) = 1$ (quando $x \geq 0$), 0 (caso contrário)
	$Max(x) = x$ (quando $x \geq 0$), 0 (caso contrário)
	c_1, c_2, c_3, c_4, c_5 (Inform. de fase de contato de pé)
Conjunto de Terminais	Posições dos dedos de cada pé
	Velocidades dos dedos de cada pé
	Ângulos absolutos no plano sagital
	Velocidades angulares absolutas no plano sagital
	Posição do centro de gravidade
	Velocidade do centro de gravidade
	Constantes aleatórias efêmeras
	Módulo sensorial de ângulo (ADF)
	Módulo sensorial de velocidade angular (ADF)
	Módulo de fase de contato de pé (ADF)

Tabela 2.1. *Conjuntos de Funções e Terminais [20].*

O fluxo da evolução proposto por Ok e Kim [20] foi definido com o cuidado de minimizar o problema da *Atribuição de Crédito*, isto é: "Dados os vários trechos de

rede (cada um responsável por um neurônio), quais aqueles que realmente contribuiram para a formação de um certo resultado?". A Figura 2.15 mostra o processo de evolução.

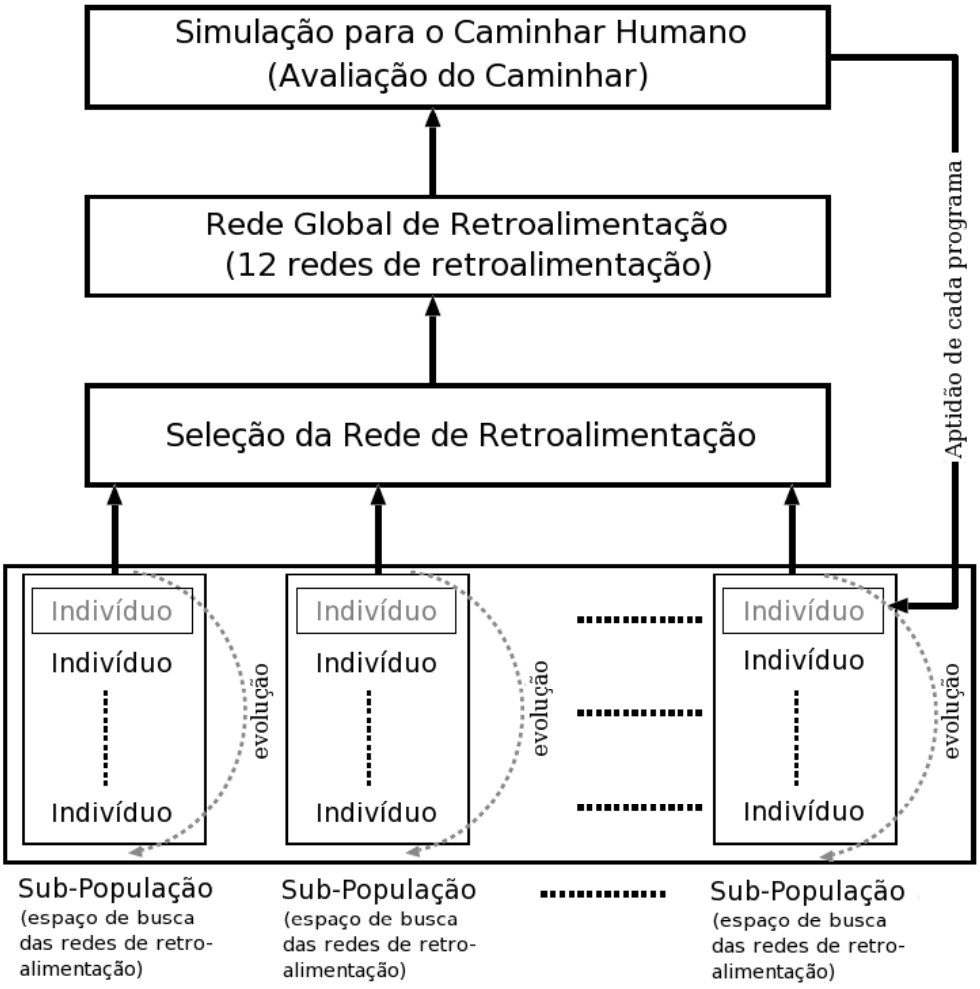

Figura 2.15. *Evoluindo a rede de retroalimentação [20].*

Hase et al. [9] e Hase e Yamazaki [8] utilizam algoritmo genético para a busca dos parâmetros internos do oscilador neural e para simular a evolução do modo de locomoção bípede do chimpanzé ao homem. Os modelos foram representados por cromossomos com genes nos reais e o gasto de energia é considerado na função objetivo, buscando-se minimizar a fatiga muscular. Nesses trabalhos, as redes de sensoriais de retroalimentação são especificadas manualmente.

Reil e Husband [23] utilizam algoritmos genéticos para a busca dos parâmetros do neurocontrolador recorrente. Esses parâmetros são os pesos das conexões, as constantes de tempo e as de desvio de cada neurônio, representados em um cromossomo com genes nos reais.

2.6. Estudo Comparativo

A Tabela 2.2 mostra uma comparação dos trabalhos baseados em CPG segundo três características:

- Genérico: Se o trabalho gera controladores para qualquer tipo de estrutura de corpos rígidos articulados.
- Automático: Se a geração do controlador acontece sem necessidade de entrada de parâmetros manuais.
- Obstáculos: Se o controlador é capaz de gerar locomoções estáveis em terrenos irregulares.

	Ni e Kato [18]	**Taga** [27]	**Reil e Husbands** [23]	**Ok e Kim** [20]	**Hase et al.** [9]	**SAN** [30]	**Proposta**
Genérico	×	×	−	×	−	/	+
Automático	×	×	+	−	−	/	+
Obstáculos	+	+	×	−	−	−	−

× : Não resolve **− : Resolve alguns casos** **/ : Resolve com limitações** **+ : Resolve**

Tabela 2.2. *Comparação dos trabalhos baseados em CPG.*

Os trabalhos de Ni e Kato [18] e Taga [27] têm uma proposta semelhante. Possuem redes e parâmetros especificados manualmente para o controle de uma estrutura humanóide, o que os torna específicos. Porém, incluem em seus sistemas módulos que tornam os modelos capazes não só de caminhar em terrenos irregulares, mas também de ultrapassar obstáculos maiores.

As redes utilizadas como CPG por Reil e Husbands [23] são genéricas o suficiente para controlar qualquer tipo de estrutura, mas algum trabalho manual é necessário para fazer as conexões entre o controlador e as juntas do modelo. O uso de algoritmo genético permite a aprendizagem automática do movimento, sem qualquer intervenção humana extra. O controle sensorial limitado não permite que o controlador ajuste-se ao ambiente para manter a estabilidade com irregularidades.

Ok e Kim [20] utilizam osciladores neurais com parâmetros e topologia pré-definida para estruturas humanóides, tornando o controlador específico para esse tipo de modelo. Porém, a rede sensorial de retroalimentação é definida automaticamente e a presença dessa rede permite que o modelo mantenha estabilidade mesmo com pequenas irregularidades no terreno.

Hase et al. [9] utiliza computação evolucionária para a busca dos parâmetros dos osciladores neurais, permitindo que, embora toda estrutura controlada deva ser humanóide, diversas características possam ser alteradas, com o controlador se adaptando para gerar o movimento adequado e estável. A rede sensorial de retroalimentação, entretanto, deve ser especificada manualmente e a presença dessa rede permite a manutenção da estabilidade em terrenos com pequenas irregularidades.

As SANs necessitam, para obter sucesso na primeira fase de busca de seus parâmetros, que os valores internos de atraso dos nós sejam escolhidos adequadamente para a estrutura a controlar, sendo isso uma limitação na sua característica automática. Uma sugestão de intervalo para esses valores internos é dada no trabalho. Quanto a serem genéricas, Van de Panne e Fiume [30] indicam que suas redes são limitadas quanto ao número de juntas do modelo (no máximo seis) para se obter um movimento adequado. As SANs são capazes de manter a estabilidade em terrenos com pequenas irregularidades. Sua definição, porém, implica, usualmente, na geração de forças e torques não suaves.

O trabalho aqui proposto poderá controlar qualquer tipo de estrutura de corpos rígidos articulada, de maneira completamente automática, e permitirá, devido ao uso de CPG baseado em osciladores neurais com redes sensoriais de retroalimentação, manter a estabilidade do modelo em ambientes com pequenas irregularidades. Não será capaz, porém, de ultrapassar obstáculos, sendo esse um fator discutido futuramente.

3

Um Modelo de Sistema Nervoso para o Controle de Animação por Dinâmica Direta

3.1. Introdução

O presente trabalho apresenta um modelo de controlador inteligente para a animação de humanóides por dinâmica direta. Reunindo os paradigmas utilizados nos estudos discutidos no Capítulo 2, pretende-se que esse modelo seja capaz de controlar comportamentos rítmicos diversos e adaptar-se, de maneira automática, a ambientes e estruturas articuladas de corpos rígidos com diferentes parâmetros automaticamente.

Experimentos neurofisiológicos têm demonstrado que movimentos rítmicos em animais, como a locomoção, são gerados por CPGs [27]. Por esse motivo, o modelo apresentado possui, em seu núcleo, um CPG baseado em osciladores neurais. Além disso, esse modelo tem sua atividade regulada por módulos sensoriais, para permitir o equilíbrio da estrutura e estabilidade do movimento mesmo em terrenos irregulares e com aplicações de pequenas forças externas, seguindo a essência da representação estímulo-resposta. Assim, pode-se caracterizar o controlador como um sistema nervoso, seguindo, então, o modelo geral de animação ilustrado na Figura 3.1, como mostrado no Capítulo 2.

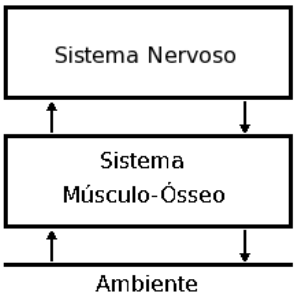

Figura 3.1. *Modelo Nervo-músculo-ósseo.*

Para alcançar o objetivo da aprendizagem de comportamentos e adaptação à estrutura articulada, adiciona-se ainda, ao modelo, um módulo cognitivo. Esse módulo será

o responsável pela busca dos parâmetros do sistema capazes de gerar os sinais corretos para controle do movimento desejado, a organização topológica do CPG e as funções de retroalimentação sensoriais.

Neste capítulo, o modelo será definido, tendo o funcionamento teórico de cada um de seus módulos explicados. Além disso, vai ser mostrado como se dá a interação entre eles. Em capítulos posteriores, detalhes de implementação serão abordados e a aplicação em alguns modelos e ambientes diferentes serão estudados.

3.2. O Sistema Músculo-Ósseo

O modelo que se pretende controlar é formado por um sistema músculo-ósseo. O modelo ósseo é um conjunto de corpos rígidos, articulados em juntas de conexão, compondo a estrutura esquelética do personagem virtual, como exemplificado na Figura 3.2, a qual modela um humanóide. A forma e a massa específica de cada um dos corpos rígidos que compõem o esqueleto determinam propriedades como massa e momentos de inércia, as quais, por sua vez, afetam a intensidade das forças musculares que devem ser aplicadas para produzir movimentos. O sistema muscular é modelado através da colocação de dispositivos atuadores nas juntas do modelo ósseo, capazes de aplicar torques computados por controladores derivativos-proporcionais, conforme descrito no Capítulo 1.

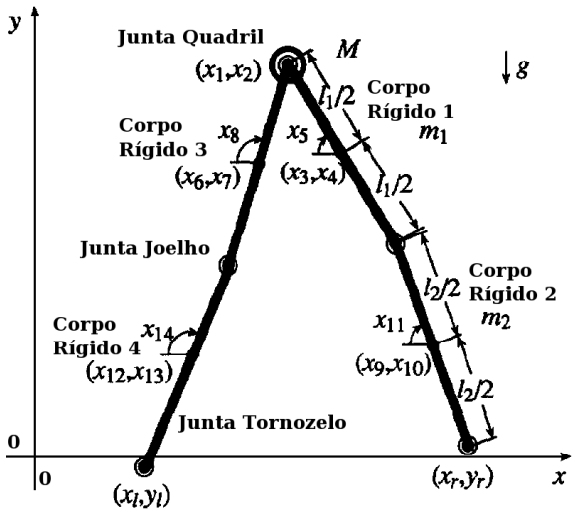

Figura 3.2. *Estrutura de Corpos Rígidos Articulados representando um modelo humanóide. [28]*

3.3. O Sistema Nervoso

3.3.1. Visão Geral

O sistema nervoso é responsável pela geração e controle do movimento do sistema ósseo adequado às condições do ambiente que o personagem percorre. É esse subsistema

que permite a manutenção do equilíbrio da estrutura e que garante a estabilidade do movimento, mesmo em terrenos irregulares e quando sujeitos à ação de forças externas.

O modelo de sistema nervoso proposto é mostrado na Figura 3.3. Seu núcleo – Módulo Neural (MN) – é um Gerador Neural de Ritmo, composto por um Gerador Central de Padrões (CPG) implementado como uma rede neural artificial chamada de oscilador neural.

O Módulo Cognitivo (MC), que é baseado em algoritmo genético e programação genética, tem a responsabilidade de trabalhar a aprendizagem de movimento e sua adaptação à estrutura articulada. Ele busca os parâmetros internos do Módulo Neural, define o modo de tratamento dos sinais oriundos do Módulo Sensorial (MS), bem como organiza a topologia do CPG, de maneira a gerar os sinais corretos para o controle do movimento desejado.

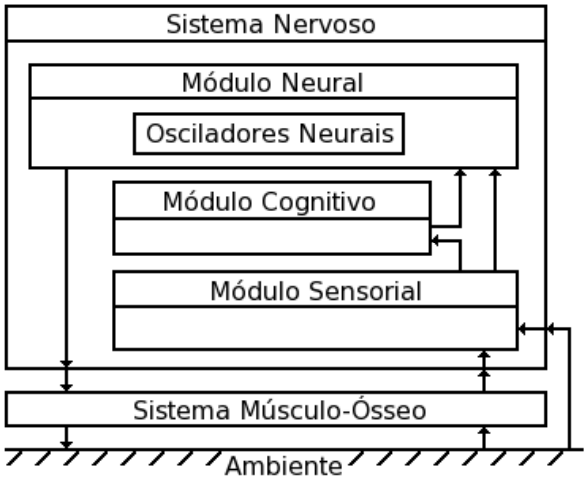

Figura 3.3. *Visão geral do modelo de sistema nervoso.*

3.3.2. Módulo Neural

O Módulo Neural é o gerador central de padrões (CPG) do modelo de controlador proposto. Núcleo deste sistema nervoso, o MN é modelado como uma rede neural artificial chamada oscilador neural, produtora dos sinais rítmicos base dos comportamentos os quais se pretende criar com este trabalho.

Neurônio

Entre os vários modelos representantes da atividade de um neurônio, o mostrado na Equação 3.1 foi escolhido por sua simplicidade matemática [12].

$$\begin{cases} \tau\dfrac{dx}{dt} + x = \sum_{j=1}^{n} w_j s_j - bv \\ \\ T\dfrac{dv}{dt} + v = y \\ \\ y = max(0, x) \end{cases} \tag{3.1}$$

O modelo representa a taxa de disparo de um neurônio por uma variável contínua com o tempo (t), onde x é o estado interno do neurônio, τ é uma constante de tempo, s_j e w_j são, respectivamente, o estímulo recebido e o peso da conexão sináptica j (> 0 para sinapses excitatórias e < 0 para sinapses inibitórias) e y é a saída (taxa de disparo).

Esse neurônio possui a propriedade de adaptação – ou auto-inibição –, isto é, quando sua saída primeiro aumenta e depois decai gradativamente. v é a variável que representa o grau de adaptação e T (> 0) e b (≥ 0) são parâmetros que controlam o curso no tempo dessa característica. A Figura 3.4a mostra o gráfico da atividade de um neurônio sem a propriedade de adaptação, isto é, $b = 0$ e a 3.4b para $b = 2.5$ e $T = 12$.

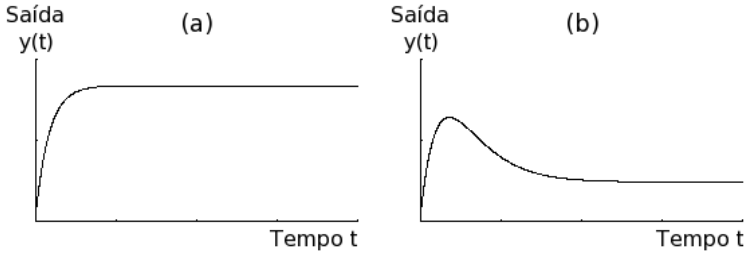

Figura 3.4. *Atividade de um neurônio. (a) sem adaptação e (b) com adaptação.*

Oscilador Neural

A característica de adaptação de um neurônio é essencial na geração de oscilações. Matsuoka [12] analisou essa e outras propriedades, bem como a dinâmica dos osciladores neurais. Esses são formados por neurônios ligados por sinapses inibitórias, sendo sua descrição uma generalização da Equação 3.1, representada pela seguinte equação diferencial:

$$\begin{cases} \tau_i\dfrac{dx_i}{dt} + x_i = -\sum_{j=1}^{n} w_{ij} y_j - bv_i + u_0 \\ \\ T_i\dfrac{dv_i}{dt} + v_i = y_i \\ \\ y_i = max(0, x_i) \qquad\qquad (i = 1, ..., n), \end{cases} \tag{3.2}$$

onde i representa o neurônio i, τ_i e T_i são as constantes de tempo, respectivamente, de seu estado interno e efeito de adaptação, u_0 é um estímulo externo de taxa constante

de um centro superior, como a região locomotora mesoencefálica [9], w_{ij} (≥ 0) é o peso da sinapse que liga o neurônio i ao j (se i = j, $w_{ij} = 0$) e y_j é a saída do neurônio j. O somatório é, na Equação 3.2, multiplicado por -1 pois, pela definição dos osciladores neurais, as sinapses são inibitórias.

É importante citar que as constantes de tempo τ_i e T_i influenciam na mudança da freqüência da oscilação, u_0 afeta a amplitude e b, ambos. Na Figura 3.5, são mostrados os sinais de um oscilador neural constituído por dois neurônios ($n = 2$) com $b = 2.5$, $u_0 = 1$, $\tau_i = 1$, $T_i = 12$ e $w_{ij} = 12$.

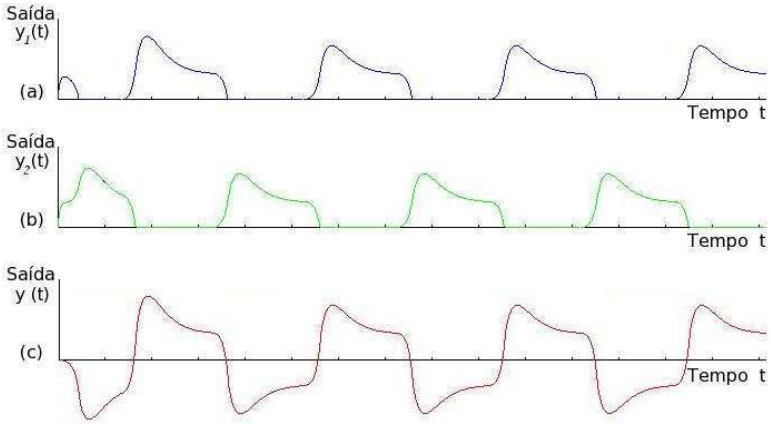

Figura 3.5. *Atividade de um oscilador neural com dois neurônios. (a) Neurônio 1, (b) Neurônio 2 e (c) sinal combinado.*

Gerador Neural de Ritmo

Um oscilador neural composto por dois neurônios mutuamente interligados é chamado Unidade Osciladora (UO). O Gerador Neural de Ritmo é, neste trabalho, uma rede formada por UOs, sendo cada unidade responsável pelo comando de uma junta da estrutura articulada de corpos rígidos, onde um neurônio é correspondente ao movimento de extensão e o outro ao de flexão.

A topologia da rede depende do corpo e do comportamento desejado. A Figura 2.11 (Capítulo 2) mostra o Gerador Neural de Ritmo proposto por Taga et al. [28], consistente com um modelo hipotético para a rede espinhal para locomoção bípede introduzido por Grillner [5], considerando-se apenas as pernas.

A título de exemplo, a Figura 3.6 mostra uma simulação onde a rede citada foi colocada a controlar um modelo humanóide. Tronco e membros superiores foram fixados e uma força aplicada na cabeça para cima impede que a estrutura caia. Embora sem equilíbrio, pois trata-se de um sistema nervoso constituído apenas pelo Gerador Neural de Ritmo, agindo somente nas pernas, sem sensoriamento e com parâmetros definidos manualmente para uma outra estrutura, é possível notar já a ordenação dos movimentos das juntas e, conseqüentemente, dos passos, deslocando-se sem ajuda de força extra.

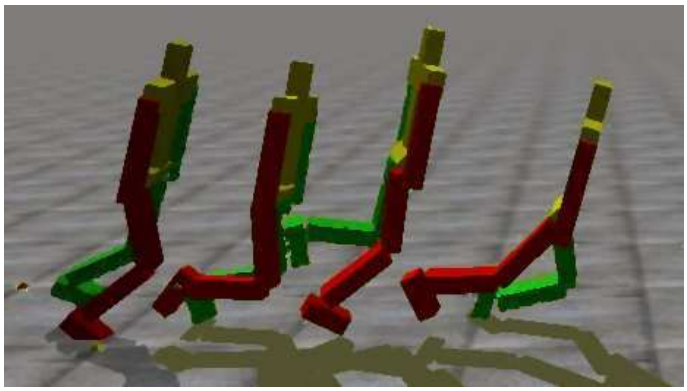

Figura 3.6. *Simulação com Gerador Neural de Ritmo sem sensoriamento.*

Os parâmetros foram escolhidos de tal forma que τ e T das unidades osciladoras do quadril sejam o dobro das dos joelhos e tornozelos, fazendo assim uma excursão de flexão e extensão no primeiro contra duas nos últimos em cada ciclo da caminhada, caracterizando o padrão de movimento bípede dos humanos [28].

As subseções seguintes dedicam-se a explicar cada um dos outros módulos do sistema nervoso proposto, responsáveis pela regulagem dos sinais gerados pelo Gerador Neural de Ritmo.

3.3.3. Módulo Sensorial

São dois os tipos de informações sensoriais recebidos pelo sistema motor humano: proprioceptiva e exteroceptiva. O primeiro tipo diz respeito às informações relativas ao próprio corpo, como, por exemplo, os estados das juntas; já o segundo tipo refere-se às informações de monitoração do ambiente [28]. A função do Módulo Sensorial no modelo de sistema nervoso proposto é captar tanto as informações proprioceptivas quanto as exteroceptivas e enviá-las aos outros módulos, para que eles regulem suas atividades (Figura 3.7).

A distância que o modelo percorre é uma informação exteroceptiva necessária ao sistema. O Módulo Cognitivo, por exemplo, usa essa informação para compor a função objetivo de seu modelo de computação evolucionária.

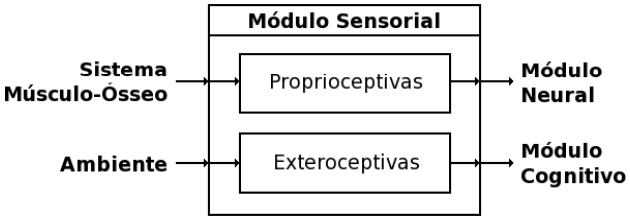

Figura 3.7. *Módulo Sensorial.*

O Módulo Neural pode fazer algumas alterações nos padrões gerados, corrigindo os sinais para manter a estabilidade mesmo com pequenas alterações no ambiente, como

inclinações no piso. Isso é conseguido a partir de informações sensoriais somáticas, através da mudança na amplitude das oscilações. Essa, como estudado na Seção 3.3.2, é afetada por u_0 na Equação 3.2. Assim, adiciona-se a ela a função $feed_i()$ (Equação 3.3), tendo como seus parâmetros os sinais vindos do Módulo Sensorial:

$$\tau_i \frac{dx_i}{dt} + x_i = -\sum_{j=1}^{n} w_{ij} y_j - b v_i + u_0 + feed_i(). \tag{3.3}$$

A função $feed()$ funciona como um reflexo. Cada neurônio i possui sua própria $feed_i()$, e, assim, formam redes de retroalimentação que, por ainda não possuírem modelo matemático bem definido, necessitam de ajuste manual [13, 20]. Hase et al. [9] determina as redes de retroalimentação segundo a regra apresentada na Equação 3.4.

$$[\text{sinal feedback}] = \sum([\text{sinal de contato}] \times [\text{sinal de ganho}]). \tag{3.4}$$

"Sinal de contato" é 1 se houver contato pé-chão e 0, caso contrário; e "sinal de ganho" é uma função de ganho que leva em consideração os deslocamentos angulares, suas compensações e suas velocidades. Deve-se ressaltar que existem exceções onde essa regra não se aplica. Isto fica claro no conjunto de equações utilizadas no mesmo trabalho.

Ok e Kim [20] propuseram o uso de programação genética para gerar as funções $feed_i()$ automaticamente. O modelo de sistema nervoso proposto neste trabalho adota programação genética no Módulo Cognitivo para gerar essas funções $feed_i()$ automaticamente. Na subseção 3.3.4 (Módulo Cognitivo) é explicada a escolha de quais informações proprioceptivas devem ser obtidas com o Módulo Sensorial.

3.3.4. Módulo Cognitivo

O módulo cognitivo é o responsável pela "aprendizagem" do movimento. Isso é feito buscando-se os parâmetros do GNR que geram o movimento desejado. Indiretamente, obtém-se também a topologia adequada, pois $w_{ij} = 0$ se não existe ligação entre os neurônios i e j e $w_{ij} > 0$ caso contrário. As bases do funcionamento deste módulo são: o Algoritmo Genético canônico [31]; e a Programação Genética [11]. Esses dois ingredientes buscam os parâmetros w_{ij}, τ_i, T_i, u_0, b e os parâmetros das funções $feed_i()$ com base em uma função de avaliação de movimento.

Para fazer a busca dos parâmetros (algoritmo genético) juntamente com a busca das funções (programação genética) em um único algoritmo, deve-se, primeiro, representá-los em um único cromossomo com o qual o Módulo Cognitivo irá trabalhar. As subseções seguintes são dedicadas à apresentação das representações de cada trecho do cromossomo.

Os Parâmetros

Os parâmetros, w_{ij}, τ_i, T_i, u_0 e b são distribuídos no cromossomo na forma ilustrada na Figura 3.8.

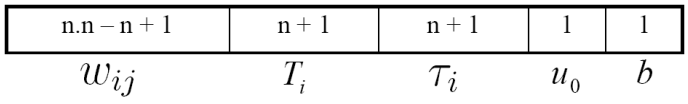

Figura 3.8. *Cromossomo com codificação dos parâmetros.*

O número de genes é definido pela quantidade de neurônios n, sendo, no total, $n.n + n + 5$. Não são considerados os pesos w_{ii} (isto é, quando $j = i$), pois não há sinapse de um neurônio com ele mesmo.

Empiricamente, observa-se que, na locomoção de um animal, suas juntas movem-se proporcionalmente umas às outras. Na literatura, nas redes construídas de forma manual, especificamente para o caminhar humano, observa-se que os valores escolhidos para uns neurônios são múltiplos dos outros [28]. Com base nessa observação, para reduzir o espaço de busca, cada trecho de cromossomo responsável por um tipo de parâmetro neural (w_{ij}, τ_i e T_i) foi modelado de tal forma que em cada gene há um valor inteiro no intervalo $[0; 3]$ para cada neurônio e o último gene é um valor real com o qual se obtém o parâmetro. Por exemplo, se a posição correspondente ao T_3 possui o valor 2 e o último gene do trecho relativo ao efeito de adaptação possui o valor $0,3$, então $T_3 = 0,6$. Os valores reais de τ_i são definidos de tal forma que o resultado esteja dentro do intervalo $0,1T_i \leq \tau_i \leq 0,5T_i$ para que as oscilações sejam estáveis [32].

Reil e Husbands [23] indicam falta de eficiência na aplicação do crossover no domínio desse problema e não o consideram em seu trabalho. Neste trabalho, porém, o crossover é considerado pelas seguintes razões:

1. É possível identificar unidades funcionais na estrutura, tais como: u_0, que altera a amplitude dos sinais e τ_i e T_i, que regulam as freqüências;

2. A topologia da rede influencia diretamente o tipo de movimento gerado.

No entanto, o crossover no modelo aqui proposto não ocorre em pontos de quebra aleatórios, mas sim nas divisões mostradas na Figura 3.8. Nos pesos, essa regra não precisa ser imposta, e as quebras podem ocorrer em pontos aleatórios para gerar maior variação topológica da rede.

Um problema relevante que surge nesse contexto é o problema de *atribuição de crédito*. Esse problema consiste em determinar qual a contribuição de cada agente quando faz uma solução em cooperação com outros. A solução adotada é baseada no trabalho de Ok e Kim para as redes de retroalimentação [20] e é explicada adiante.

Na geração dos genes, seja ela para a criação da população inicial, seja para a mutação, também é importante levar em conta o fato de que a topologia da rede influencia diretamente o tipo de movimento gerado (razão 2 citada anteriormente). Para criar uma variedade de redes topologicamente diferentes, é importante que a probabilidade de haver ligação entre dois neurônios seja a mesma probabilidade de não haver ligação entre eles. Ou seja, a probabilidade de $w_{ij} = 0$ deve ser igual à de $w_{ij} > 0$. Quando dois neurônios pertencem a uma mesma unidade oscilatória, $w_{ij} \neq 0$, obrigatoriamente, e todos os outros genes devem assumir valores maiores do que zero.

A função de avaliação é específica de cada problema. Reil e Husband [23] mostraram que, apesar de sua simplicidade, a distância percorrida pelo modelo é uma boa função de avaliação para o escopo da locomoção, objetivo principal deste trabalho. Assim, essa é a função objetivo adotada:

$$S = D, \tag{3.5}$$

onde S é a função de avaliação e D é a distância percorrida pelo modelo.

As Funções de retroalimentação e o Processo de Evolução

Para a busca das funções $feed_i()$, utiliza-se Programação Genética [20, 13], que se trata de uma técnica onde se procura em um espaço de busca composto por todos os programas de computador possíveis, aquele capaz de resolver o problema. Isso é feito representando os programas por árvores, onde seus nós internos são operações (funções) e suas folhas são os parâmetros de entrada, sendo assim chamados de terminais. As operações genéticas (duplicação, crossover e mutação) são então aplicadas às árvores.

Wolff e Nordin [33] sugeriram uma representação linear do cromossomo para Programação Genética. Cada gene tem em seu valor o número de um registrador onde se encontra um terminal ou uma função (e, portanto, um valor inteiro). No modelo aqui sugerido, essa idéia de linearização é utilizada para representar cada $feed_i()$ na forma mostrada na Figura 3.9.

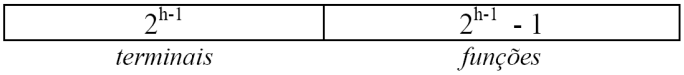

Figura 3.9. *Cromossomo com codificação de uma função feed().*

O número de genes é definido de acordo com a altura máxima h que teria a árvore. Todos os programas têm o mesmo tamanho e seus cromossomos são traduzidos da forma explicada a seguir.

Em Programação Genética, os programas são gerados a partir de elementos de dois conjuntos dados: um conjunto de funções; e um conjunto de terminais (vide exemplo da Tabela 3.1).

Funções	**Terminais**
+, -, *	5, 1, 0, 4, 7

Tabela 3.1. *Conjuntos de Funções e Terminais.*

Supondo-se, por exemplo, os conjuntos de funções e terminais indicados na Tabela 3.1, e um programa representado por uma árvore de altura 3; o programa a ser gerado tem 2^{3-1} funções e $(2^{3-1} - 1)$ terminais, totalizando um cromossomo de 7 genes. Considerando-se então o cromossomo:

3, 0, 1, 3, 0, 2, 1

Sua tradução é equivalente ao programa:

```
x = 4 + 5;
y = 1 * 4;
retorna x - y;
```

Dessa forma, pode-se concatenar os programas no cromossomo que determina o indivíduo, que passa a ter $n.n + n + 5 + n.p$ genes, onde p é o número de genes de um programa.

Do mesmo modo que nos grupos funcionais dos parâmetros, o crossover não quebra programas ao meio, fazendo apenas combinações desses para gerar a rede de retroalimentação do indivíduo, e deixando a variação de cada $feed_i()$ por conta da mutação. Mais uma vez surge o problema de *atribuição de crédito*. Para resolvê-lo, idéia semelhante à sugerida por Ok e Kim [20] é aplicada, modificando-a para o modelo evolucionário como descrito até então, tanto relativo ao modo de codificação de um indivíduo, quanto relativo às restrições de crossover. O processo está ilustrado na Figura 3.10.

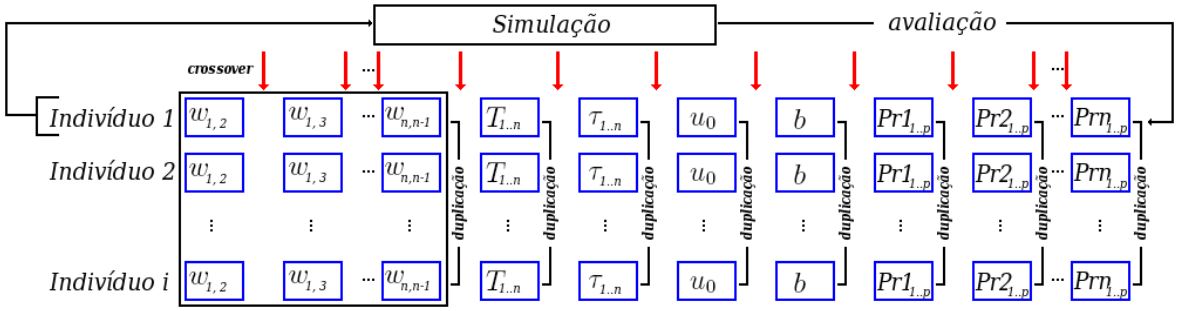

Figura 3.10. *Processo de evolução.*

Para a duplicação, cada bloco funcional deve ser considerado um cromossomo diferente – exceto pelos pesos, que são agrupados como um só – e cada um deles recebe o mesmo valor de avaliação, pois representam apenas um indivíduo. No crossover, os possíveis pontos de quebra estão marcados na Figura 3.10 com as setas. A mutação altera o valor aleatoriamente de um gene com probabilidade menor que 1%.

Baseado nos trabalhos de Ok e Kim [20] e Miyashita et al. [13] sobre evolução das redes de retroalimentação e na Equação 3.4, são montados os conjuntos de funções e terminais utilizados pelo módulo cognitivo para a Programação Genética. Tem-se, nos conjuntos, 5 funções e $t = 7q + 2b + 10$ terminais, onde q e b são, respectivamente, o número de pés e de corpos da estrutura articulada (Tabela 3.2).

	Descrições	**Qtd**
Conjunto de Funções	*+, -, **	*3*
	$H(x) = 1$ *(quando* $x \geq 0$*),* 0 *(caso contrário)*	*1*
	$Max(x) = x$ *(quando* $x \geq 0$*),* 0 *(caso contrário)*	*1*
Conjunto de Terminais	*Posições dos pés (x, y, z)*	$3 * q$
	Velocidades dos pés (x, y, z)	$3 * q$
	Contatos Pé-Chão (0 ou 1 para cada pé)	q
	Ângulos dos corpos no plano Sagital	b
	Velocidades angulares dos corpos	b
	Posição do Centro de Gravidade (x, y, z)	*3*
	Velocidade do Centro de Gravidade (x, y, z)	*3*
	4 constantes aleatórias	*4*

Tabela 3.2. *Funções e Terminais utilizados pelo MC.*

As funções unárias $H(x)$ e $M(x)$ têm como argumento o elemento mais à esquerda. É importante notar que, com elas, é possível encontrar, no modelo de representação completa linear, programas que seriam representados por árvores incompletas (Figura 3.11).

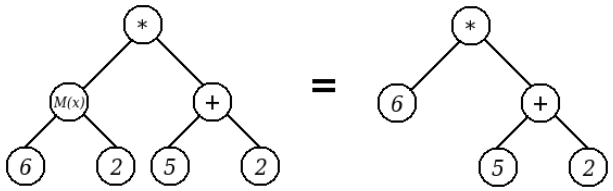

Figura 3.11. *Mesmo programa representado por uma árvore completa e uma incompleta.*

No exemplo da Figura 3.11, é importante lembrar que os terminais indicam referência, e não o valor, pois os programas somente serão, de fato, os mesmos, se o terminal 6 só puder assumir valores maiores ou iguais a 0.

O conjunto de terminais tem seus valores obtidos através do Módulo Sensorial. Logo, é importante que o mesmo tenha informações de "auto-consciência", como número de pés e número de corpos. A cada passo no tempo da simulação, o conjunto de terminais deve ser realimentado com os valores dos sensores do modelo. Concatenam-se, então, os dois trechos de cromossomos descritos anteriormente para formar um indivíduo. Assim, com o cromossomo definido heterogeneamente, contendo informações tanto de parâmetros como de programas (funções), pode-se trabalhá-lo usando o algoritmo genético canônico, observando-se apenas o processo mostrado na Figura 3.10 para o problema da *atribuição de crédito*. O algoritmo que representa o funcionamento do Módulo Cognitivo é, então:

Algoritmo 1. *Funcionamento do Módulo Cognitivo.*

```
Gera populacao aleatoriamente
Repita
   Para cada individuo da populacao
      Execute Simulacao
      Atribua individuo.valor_avaliacao
   Calcula media_avaliacao
   Para cada individuo da populacao
      individuo.aptidao <- individuo.valor_avaliacao/media_avaliacao
   populacao_intermediaria = Duplica(populacao, AMOSTRAGEM_ESTOCASTICA_DO_RESTANTE)
   populacao = Crossover(populacao_intermediaria, PROBABILIDADE_DE_CROSSOVER)
   populacao = Mutacao(populacao, PROBABILIDADE_DE_MUTACAO)
```

Amostragem estocástica do restante é o modo de cálculo para duplicar-se um indivíduo. Dado um valor de aptidão do mesmo, a parte inteira indica quantas cópias haverá na população intermediária e a parte decimal diz a probabilidade de se ter outra cópia [31]. Por exemplo, se um indivíduo tem aptidão igual a $2,3$, então serão feitas duas cópias dele e haverá 30% de chances de existir uma terceira.

3.4. Considerações Finais

O modelo de controlador proposto tem as vantagens de ser flexível ao comportamento (pode gerar diferentes tipos de movimentos, desde que rítmicos), ao corpo (pode controlar estruturas de corpos rígidos com parâmetros diversos), adaptável ao ambiente e automático. Por ser projetado para controlar um sistema músculo-ósseo simulado por dinâmica direta, tem a qualidade ainda de gerar animações fisicamente consistentes.

O modelo também é capaz de gerar outros tipos de movimentos além da locomoção, oscilatórios ou não (esses movimentos podem ser vistos como movimentos oscilatórios com freqüências muito baixas), bastando, para isso, que se encontrem outras funções de avaliação ideais para eles.

São contribuições do modelo proposto:

- Definição de uma modularização para o modelo nervo-músculo-ósseo;
- Compilação de diversos atributos de modelos propostos em trabalhos anteriores, ficando mais genérico e de maior flexibilidade;
- Definição do Módulo Cognitivo e sua interação com os outros módulos, permitindo maior automatização na geração da animação e tornando indireta a dependência entre o comportamento e a topologia do CPG.

São limitações do modelo: a falta de controle do animador no resultado gerado, a incapacidade de ultrapassagem de grandes obstáculos e a necessidade de se modelar cuidadosamente estruturas fisicamente capazes de se movimentar. A última limitação pode ser interessante quando se deseja verificar a capacidade de se mover de um ser. Quanto às duas primeiras, no Capítulo 6 são discutidos trabalhos futuros que podem contribuir para minimizá-las ou resolvê-las.

No Capítulo 4 é discutida a implementação do modelo proposta neste trabalho. Aplicações serão estudadas e demonstradas no Capítulo 5 e, no Capítulo 6, são dadas as conclusões e os possíveis trabalhos futuros.

4

O Modelo Orientado a Objetos

4.1. Introdução

Este capítulo é destinado à discussão sobre aspectos do projeto de implementação do controlador proposto no Capítulo 3. Uma modelagem orientada a objetos é introduzida e suas classes são discutidas. A idéia geral foi transformar cada módulo do modelo teórico em uma classe, adicionando-se algumas classes auxiliares para o trabalho interno dos módulos. Na Figura 4.1 é mostrado o diagrama de classes que modela as implementações, considerando-se apenas a estrutura de corpos rígidos articuladas e o sistema nervoso, para focar a discussão no controlador.

4.2. Classe RLBModel

A classe **RLBModel** (*Rigid Link Body Model*) representa o modelo a ser controlado e possui um sistema nervoso associado. Essa associação é representada pelo atributo *NSystem* do tipo **NervousSystem**.

Na classe **RLBModel** é onde a estrutura de corpos rígidos articulada deve ser descrita. Seus métodos são:

- *RLBModel*, que é o construtor da classe e onde a estrutura é construída;
- *InitStateReturn*, que retorna a estrutura e seus parâmetros para a configuração e posição iniciais, para iniciar a simulação de novo indivíduo neural;
- *Move*, que atualiza a configuração da estrutura segundo os sinais recebidos do Sistema Nervoso;
- *Draw*, que desenha a estrutura.

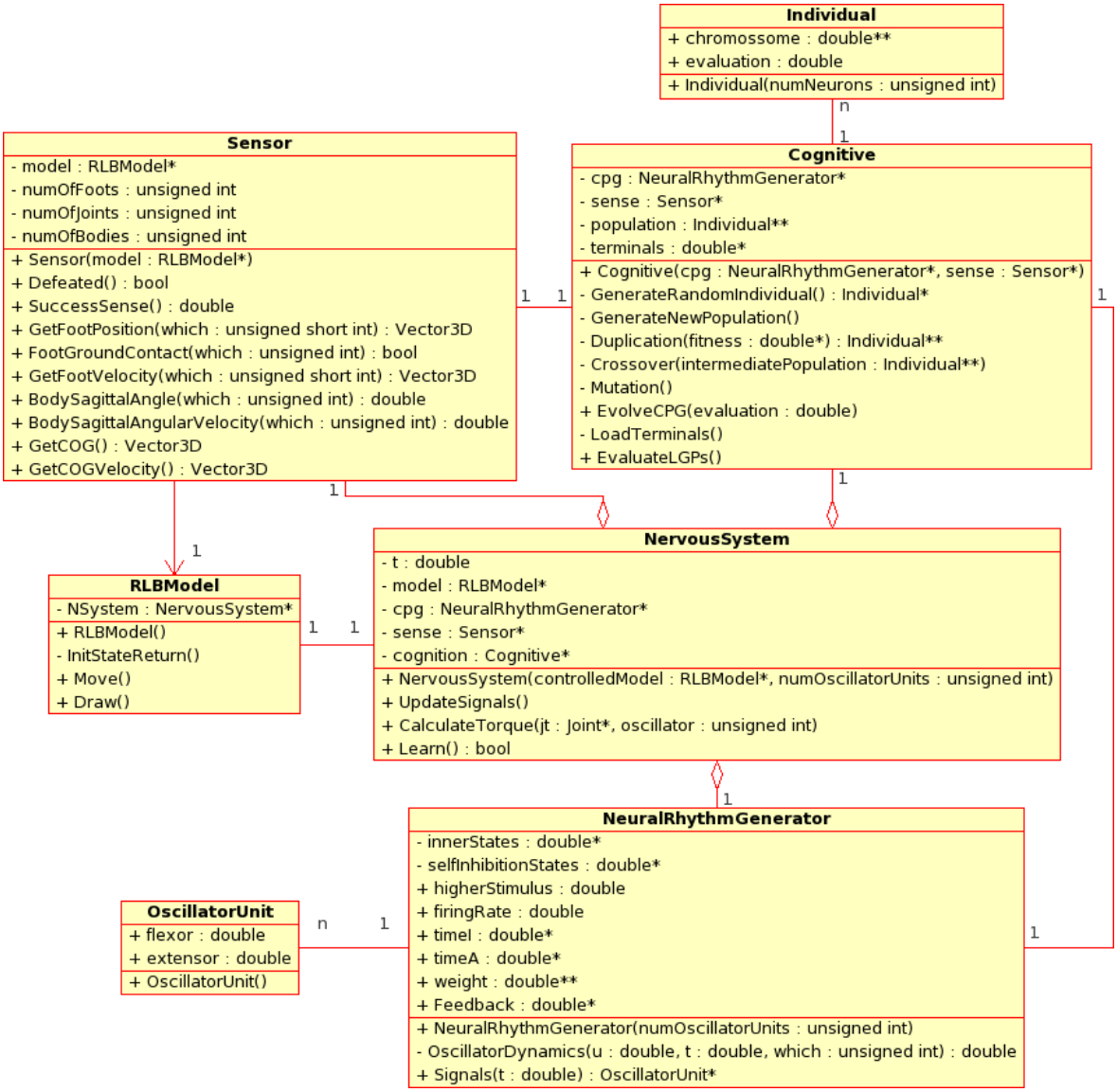

Figura 4.1. *Diagrama de Classes modelando o controlador proposto.*

4.3. Classe NervousSystem

A classe NervousSystem representa o sistema nervoso. É o controlador propriamente dito e agrega os módulos neural, sensorial e cognitivo, representados, respectivamente, pelos atributos *cpg* (**NeuralRhythmGenerator**), *sense* (**Sensor**) e *cognition* (**Cognitive**).

Esta classe possui ainda uma variável t para controle do "tempo de vida" de um indivíduo, isto é, para determinar quando parar sua simulação e passar para o próximo. O sistema nervoso também tem "consciência" da estrutura por ele controlada, representada pelo atributo *model* do tipo **RLBModel**. São seus métodos:

- *NervousSystem*, que é o construtor da classe;
- *UpdateSignals*, que atualiza os valores dos sensores e sinais nas unidades oscilatórias;
- *CalculateTorque*, que calcula o torque a ser aplicado em uma determinada junta segundo o sinal em sua respectiva unidade oscilatória. É utilizado pelo método *Move* da classe **RLBModel**;
- *Learn*, que verifica se a estrutura caiu (através de *sense*) ou se o "tempo de vida" da atual configuração neural, isto é, indivíduo, acabou, para treinar um novo e, assim, continuar o processo de aprendizado.

4.4. Classe OscillatorUnit

A classe **OscillatorUnit** representa uma unidade oscilatória e serve exclusivamente para armazenar os sinais, com um valor inteiro para o neurônio flexor e outro para o extensor. Seu único método é o construtor *OscillatorUnit*.

4.5. Classe NeuralRhythmGenerator

A classe **NeuralRhythmGenerator** implementa o Gerador Neural de Ritmo, modelando o Módulo Neural. Possui como atributos *innerStates*, *selfInhibitionStates*, *timeI*, *timeA*, *firingRate*, *higherStimulus*, *weight* e *Feedback*, representando, respectivamente, y_i, v_i, τ_i, T_i, b, u_0, w_{ij} e $feed_i()$ das equações do oscilador neural.

As funções de retroalimentação têm apenas seus valores armazenados em *Feedback*, sendo esses atualizados a cada instante de tempo pelo método *UpdateSignals* da classe **NervousSystem**, através da entrada obtida pelo módulo sensorial (classe **Sensor**) e função encontrada pelo módulo cognitivo (classe **Cognitive**).

São métodos da classe **NeuralRhythmGenerator**:

- *NeuralRhythmGenerator*, que é o construtor da classe;
- *OscillatorDynamics*, que resolve as equações diferenciais do oscilador neural, implementando o método de Runge-Kutta de Quarta ordem;

- *Signals*, que retorna os valores dos sinais nas unidades oscilatórias.

4.6. Classe Sensor

A classe **Sensor** modela o módulo sensorial. Tem como atributos a estrutura sensoriada (*model*) e suas características: número de pés (*numOfFoots*), número de juntas (*numOfJoints*) e número de corpos (*numOfBodies*).

Esta classe funciona como uma interface entre o controlador e o ambiente e engloba os diversos sensores da estrutura, sendo elas acessadas através de seus métodos. Esses disponibilizam as funções necessárias ao conjunto de funções da Programação Genética executada pelo módulo cognitivo. Os valores são conseguidos através do motor de dinâmica (*ODE*). São os métodos:

- *Sensor*, que é o construtor da classe;
- *Defeated*, que verifica se a estrutura caiu;
- *SucessSense*, que calcula o valor de avaliação do indivíduo (distância percorrida);
- *GetFootPosition*, que retorna a posição de cada pé (x, y, z);
- *FootGroundContact*, que verifica se cada pé está em contato com o chão;
- *GetFootVelocity*, que retorna a velocidade de cada pé (x, y, z);
- *BodySagittalAngle*, que retorna o ângulo de cada junta no plano sagital;
- *BodySagittalAngularVelocity*, que retorna a velocidade angular de cada junta no plano sagital;
- *GetCOG*, que retorna a posição do centro de gravidade (x, y, z);
- *GetCOGVelocity*, que retorna a velocidade do centro de gravidade (x, y, z).

4.7. Classe Individual

A classe **Individual** representa um indivíduo com o qual o módulo cognitivo trabalha para o aprendizado do movimento. Ele possui um cromossomo (*chromossome*) conforme especificado no Capítulo 3, e seu valor de avaliação (*evaluation*). O único método da classe é o construtor *Individual*.

4.8. Classe Cognitive

A classe **Cognitive** representa o módulo cognitivo. Possui um gerador neural de ritmo a evoluir (*cpg*), um sensor (*sense*) para avaliar o indivíduo, uma população (*population*) para trabalhar com seu algoritmo evolucionário e um conjunto de terminais (*terminals*), preenchido pelo sensor, para execução da programação genética.

São seus métodos:

- *Cognitive*, que é o construtor da classe. A população inicial de indivíduos aleatórios é criada aqui;
- *GenerateRandomIndividual*, que gera um indivíduo (cromossomo) aleatoriamente;
- *GenerateNewPopulation*, que cria nova geração da população. O algoritmo evolucionário geral do módulo, conforme discutido na seção 3.3.4, é implementado neste método;
- *Duplication*, que implementa o algoritmo de duplicação por amostragem estocástica do restante;
- *Crossover*, que implementa o crossover conforme regras discutidas na seção 3.3.4;
- *Mutação*, que implementa a mutação;
- *EvolveCPG*, que carrega os valores no *cpg* para cada indivíduo para simulá-lo e avaliá-lo. Executa *GenerateNewPopulation*, que ao fim dos testes de toda uma população;
- *LoadTerminals*, que requisita ao sensor os valores dos terminais para preencher o conjunto;
- *EvaluateLGPs*, que executa os programas $feed_i()$ para os valores de terminais carregados com *LoadTerminals* e atribui os resultados às variáveis *Feedback* dos neurônios no *cpg* (**NeuralRhythmGenerator**). É chamado pelo método *UpdateSignals*, que da classe **NervousSystem**.

A amostragem estocástica do restante foi implementada usando um método chamado de amostragem universal estocástica [31]. Em um vetor de 100 posições, são distribuídos os indivíduos conforme sua probabilidade de duplicação. Por exemplo, se um um indivíduo tem 20% de chances de ser duplicado, então 20 posições em seqüência são destinadas a ele. N números são então escolhidos aleatoriamente no intervalo $[1; 100]$ e, para cada um, o indivíduo localizado no índice sorteado é selecionado, até preencher toda a nova população. Isso é semelhante a uma roleta com um gráfico do tipo "pizza", onde são feitas N rodadas, e a fatia na qual o ponteiro cair em cada uma diz o indivíduo selecionado.

Um ponto importante a comentar é o fato do método *LoadTerminals* exigir informações proprioceptivas, diferente do especificado no modelo teórico apresentado no

Capítulo 3, Seção 3.3.3. Isso acontece apenas como estratégia de implementação, mas não a torna inconsistente, pois o resultado da operação somente servirá, de fato, à classe **NeuralRhythmGenerator**, que implementa o módulo neural, através do atributo *Feedback*.

4.9. Considerações Finais

Neste capítulo foi mostrada uma implementação para o modelo teórico apresentado no Capítulo 3. É importante citar que essa não é a única solução possível, porém é interessante devido à sua semelhança com a modelagem teórica, ou seja, para cada módulo, há uma classe. Além disso, fica claro o papel do sistema nervoso como agregante dos outros módulos e há uma classe (*RLBModel*) que serve de interface genérica para qualquer que seja o modo de implementação da estrutura de corpos rígidos articulados.

5

Estudos de Casos

5.1. Considerações Gerais

O modelo de controlador proposto é aplicável a qualquer estrutura constituída de corpos rígidos articulados e cria automaticamente seu modo de andar.

Os estudos de casos apresentados nesta seção tratam da locomoção de cinco modelos: Humanóide, *Cheetah* [3], Sapo, *Luxo* e *Luxo*-2. As implementações foram feitas em C++ no sistema operacional Linux em dois computadores: CP1 - Pentium D 2.6 GHz com 1 GB de RAM (todos os modelos); e CP2 - Athlon XP 1.5 GHz com 256 MB de RAM (humanóide e *cheetah*).

As estruturas dos modelos foram descritas com o *Open Dynamics Engine* (*ODE*) [26], motor usado para a simulação dinâmica e que também possui funções sensoriais que servem ao Módulo Sensorial. Dessa forma, o Módulo Sensorial funciona, principalmente, como uma interface entre os outros Módulos do sistema nervoso e o motor dinâmico.

A simulação no motor ODE foi realizada com passo de 0,001 para solução das equações de movimento. As colisões entre o chão e as partes da estrutura articulada foram consideradas. No entanto, as colisões entre partes da estrutura articulada foram desativadas para possibilitar os movimentos no mesmo plano.

O método de solução das equações diferenciais do Gerador Neural de Ritmo utilizado foi o Runge-Kutta de 4ª ordem, com passo $h = 0,001$. Os parâmetros do Módulo Cognitivo são descritos na tabela 5.1.

Parâmetros	**Valores**
Valor máximo de um gene	15,0
População	100 indivíduos
Probabilidade de Crossover	60%
Tipo de Crossover	Multiponto
Probabilidade de Mutação	0,1%
Altura das árvores dos programas de $feed_i()$	10

Tabela 5.1. *Parâmetros do Módulo Cognitivo.*

Com base nesses dados, o tamanho de um programa $feed_i()$ é:

$$p = 2^{10-1} + 2^{10-1} - 1 = 1023 \; genes.$$

A avaliação de cada indivíduo para no momento em que 10.000 passos da simulação são atingidos ou quando o modelo cai. A definição de queda varia de acordo com o modelo:

- Humanóide: Colisão com o chão de qualquer parte do corpo diferente dos pés;
- *Cheetah*: Colisão da cabeça com o chão;
- Sapo: Colisão com o chão de qualquer parte do corpo diferente dos membros inferiores (considerados os pés do modelo);
- *Luxo*: Colisão com o chão de qualquer parte do corpo diferente da base;
- *Luxo*-2: Não existe queda. A simulação do indivíduo para apenas pelo número de passos da simulação.

5.2. Modelos Analisados

As estruturas de corpos rígidos articulados são ilustradas na Figura 5.1. As juntas, representadas por círculos, estão no mesmo plano (Sagital), cada uma com 1 grau de liberdade de rotação, representando, então, modelos 2D.

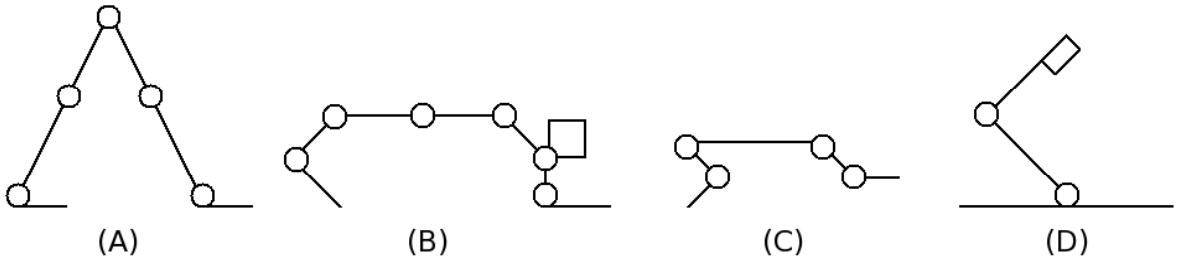

Figura 5.1. *Estruturas de corpos rígidos articulados: (A) humanóide, (B) cheetah [3], (C) sapo, (D) Luxo e Luxo-2.*

A figura 5.1 exibe apenas a representação topológica dos modelos, não possuindo as mesmas relações de comprimentos entre os corpos que os implemetados e, assim, desconsiderando as diferenças entre *Luxo* e *Luxo*-2.

5.2.1. Modelo Humanóide

O modelo humanóide possui 5 juntas (cintura, joelhos esquerdo e direito e tornozelos esquerdo e direito) e 6 corpos rígidos (pés esquerdo e direto, pernas esquerda e direita, e coxas esquerda e direita). As relações de massa e comprimento utilizadas são mostradas

na Tabela 5.2 e seus valores foram baseados no trabalho de Laszlo [10]. A tabela 5.3 mostra os ângulos mínimos e máximos permitidos a cada junta.

O sistema nervoso para controlar esse modelo possui 10 neurônios, um par em cada uma das 5 unidades oscilatórias associadas aos graus de liberdade. O número de genes de parâmetros do oscilador neural é:

$$r_h = 10 \cdot 10 + 10 + 5 = 115 \; genes.$$

Assim, o número total de genes em seu cromossomo é:

$$g_h = r_h + n_h \cdot p = 115 + 10 \cdot 1023 = 10345 \; genes.$$

O número de elementos no conjunto de terminais do humanóide é:

$$t_h = 7q_h + 2b_h + 10 = 7 \cdot 2 + 2 \cdot 6 + 10 = 36 \; terminais.$$

	Coxas	**Pernas**	**Pés**
Massa	8,40	4,16	1,20
Comprimento	0,45	0,40	0,25

Tabela 5.2. *Relações de Massa e Comprimento do humanóide.*

	Mínimo	**Máximo**
CE-CD	-1,5	1,5
CE-PE	0,1	2,5
CD-PD	0,1	2,5
PE-PeE	-0,9	0,5
PD-PeD	-0,9	0,5

C : "coxa" **P : "perna"** **Pe : "pé"**

E : "esquerda" **D : "direita"**

Tabela 5.3. *Ângulos mínimos e máximos de cada junta do humanóide. (em rad)*

5.2.2. Modelo da Cheetah

O modelo da *cheetah* possui 6 juntas articuladas e 8 corpos rígidos (um dos corpos rígidos é a cabeça, que é representada por um quadrado na Figura 5.1) e foi baseado no trabalho de van de Panne et al. [3]. As relações de massa e comprimento utilizadas são mostradas na Tabela 5.4. A tabela 5.5 mostra os ângulos mínimos e máximos permitidos a cada junta.

	Corpo 1	Corpo 2	Corpo 3	Corpo 4	Corpo 5	Corpo 6	Corpo 7	Cabeça
Massa	0,06	0,07	0,15	0,15	0,15	0,15	0,10	0,10
Comprimento	0,60	0,60	0,60	0,60	0,60	0,60	0,60	0,20

Corpos 1 a 7 da esquerda para a direita na figura 5.1B. Cabeça representada pelo quadrado.

Tabela 5.4. *Relações de Massa e Comprimento da cheetah.*

	Mínimo	Máximo
C1-C2	-2,2	0
C2-C3	-1,0	0
C3-C4	-0,7	0
C4-C5	-0,7	-0,2
C5-C6	-2,0	0
C6-C7	0	2,0
C5-Cb	fixa	fixa

C : "corpo" Cb : "cabeça"

Tabela 5.5. *Ângulos mínimos e máximos de cada junta da cheetah. (em rad)*

O sistema nervoso para controlar esse modelo possui 12 neurônios, um par em cada uma das 6 unidades oscilatórias associadas aos graus de liberdade. O número de genes de parâmetros do oscilador neural é:

$$r_c = 12 \cdot 12 + 12 + 5 = 161 \; genes.$$

Assim, o número de genes em seu cromossomo é:

$$g_c = r_c + n_c \cdot p = 161 + 12 \cdot 1023 = 12437 \; genes.$$

O número de elementos no conjunto de terminais da *cheetah* é:

$$t_c = 7q_c + 2b_c + 10 = 7 \cdot 2 + 2 \cdot 8 + 10 = 40 \; terminais.$$

5.2.3. Modelo do Sapo

O modelo do sapo possui 4 juntas e 5 corpos rígidos. As relações de massa e comprimento utilizadas são mostradas na Tabela 5.6, enquanto a 5.7 mostra os ângulos mínimos e máximos permitidos a cada junta.

	Corpo 1	Corpo 2	Corpo 3	Corpo 4	Corpo 5
Massa	0,10	0,15	1,00	0,15	0,10
Comprimento	0,25	0,25	0,50	0,25	0,25

Corpos 1 a 5 da esquerda para a direita na figura 5.1C.

Tabela 5.6. *Relações de Massa e Comprimento do sapo.*

	Mínimo	Máximo
C1-C2	-2,5	-0,1
C2-C3	-0,5	1,3
C3-C4	-0,5	1.3
C4-C5	0,1	2,5

Tabela 5.7. *Ângulos mínimos e máximos de cada junta do sapo. (em rad)*

O sistema nervoso para controlar esse modelo possui 8 neurônios, um par em cada uma das 4 unidades oscilatórias associadas aos graus de liberdade. O número de genes de parâmetros do oscilador neural é:

$$r_s = 8 \cdot 8 + 8 + 5 = 77 \; genes.$$

Assim, o número total de genes em seu cromossomo é:

$$g_s = r_s + n_s \cdot p = 77 + 8 \cdot 1023 = 8261 \; genes.$$

O número de elementos no conjunto de terminais do humanóide é:

$$t_s = 7q_s + 2b_s + 10 = 7 \cdot 2 + 2 \cdot 5 + 10 = 34 \; terminais.$$

5.2.4. Modelos Luxo e Luxo-2

Topologicamente, os modelos *Luxo* e *Luxo*-2 são iguais. Ambos possuem 2 juntas articuladas e 4 corpos rígidos. As relações de massa e comprimento utilizadas são mostradas nas tabelas 5.8 (*luxo*) e 5.10 (*luxo*-2), enquanto as 5.9 (*luxo*) e 5.11 (*Luxo*-2) mostram os ângulos mínimos e máximos permitidos a cada junta.

	Corpo 1	Corpo 2	Corpo 3	Corpo 4
Massa	0,087	0,170	0,500	0,020
Comprimento	3,000	1,500	1,500	0,300

Corpos 1 a 4 de baixo para cima na figura 5.1D.

Tabela 5.8. *Relações de Massa e Comprimento do luxo.*

	Mínimo	Máximo
C1-C2	-1,2	0,0
C2-C3	0,0	2,2
C3-C4	fixa	fixa

Tabela 5.9. *Ângulos mínimos e máximos de cada junta do luxo. (em rad)*

	Corpo 1	Corpo 2	Corpo 3	Corpo 4
Massa	0,087	0,170	0,500	0,020
Comprimento	1,800	1,200	1,200	0,300

Corpos 1 a 4 de baixo para cima na figura 5.1D.

Tabela 5.10. *Relações de Massa e Comprimento do luxo-2.*

	Mínimo	Máximo
C1-C2	-1,3	1,3
C2-C3	-2,6	0,0
C3-C4	fixa	fixa

Tabela 5.11. *Ângulos mínimos e máximos de cada junta do luxo-2. (em rad)*

O sistema nervoso para controlar esses modelos possui 4 neurônios, um par em cada uma das 2 unidades oscilatórias associadas aos graus de liberdade. O número de genes de parâmetros do oscilador neural é:

$$r_l = 4 \cdot 4 + 4 + 5 = 25 \; genes.$$

Assim, o número total de genes em seu cromossomo é:

$$g_l = r_l + n_l \cdot p = 25 + 4 \cdot 1023 = 4117 \; genes.$$

O número de elementos no conjunto de terminais do humanóide é:

$$t_l = 7q_l + 2b_l + 10 = 7 \cdot 1 + 2 \cdot 4 + 10 = 25 \; terminais.$$

5.3. Resultados

Os tempos médios aproximados de simulação necessários para avaliar-se uma população de cada um dos modelos testados, e as respectivas quantidades de gerações com as quais foram obtidos os movimentos adequados, encontram-se na Tabela 5.12. Apenas o humanóide e a *cheetah* foram também testados nos CP1 e CP2. Em ambos os computadores, as simulações dos modelos rodaram satisfatoriamente em tempo real.

	CP1	CP2	Gerações
humanóide	$15s$	$30s$	3000
cheetah	$2m24s$	$7m12s$	300
sapo	$2m11s$	–	2000
luxo	$3m17s$	–	750
luxo-2	$4m08s$	–	40

Tabela 5.12. *Resumo das simulações.*

A evolução mais lenta (em número de gerações) do modelo humanóide deve-se à menor estabilidade do modelo, quando comparado aos outros modelos. Com um modelo menos estável se pode descartar muitos indivíduos baseado no critério de "queda", sendo menor, então, o tempo médio de avaliação de uma população de humanóides quando comparado com os outros.

Nas páginas seguintes são mostradas figuras com os movimentos obtidos para cada um dos modelos após treinados, bem como o comportamento deles em um terreno com rampa. Sobre cada modelo, tem-se a comentar:

- **Humanóide** (Figura 5.2): No terreno plano ou com rampa, o modelo alcançou mesma estabilidade, sem treino adicional para o segundo caso. O resultado obtido foi superior ao trabalho de Ok e Kim [20] em número de passos, e com a característica de definir automaticamente parâmetros e topologia do gerador neural de ritmo, o que não acontece no trabalho citado;

- **Cheetah** (Figura 5.3): Nota-se que, ao tentar ultrapassar o obstáculo, o modelo realiza tentativas e ajustes, conseguindo finalmente ultrapassá-lo, mantendo o mesmo ritmo obtido no terreno plano;

- **Sapo** (Figura 5.4): Tanto no terreno plano quanto na rampa, locomove-se da mesma maneira;

- **Luxo** (Figura 5.5): Adquiriu o modo de locomover-se através de pulos e ultrapassa a rampa mantendo o mesmo ritmo do terreno plano;

- **Luxo-2** (Figura 5.6): Com a definição diferente do seu corpo em relação ao *Luxo* quanto aos tamanhos dos corpos e liberdades das juntas, o modelo adquiriu o modo de locomover-se impulsionando seu corpo e depois contraindo para rodar. Observa-se que, na rampa, quando a velocidade é perdida, o modelo novamente impulsiona-se, conseguindo ultrapassar o obstáculo.

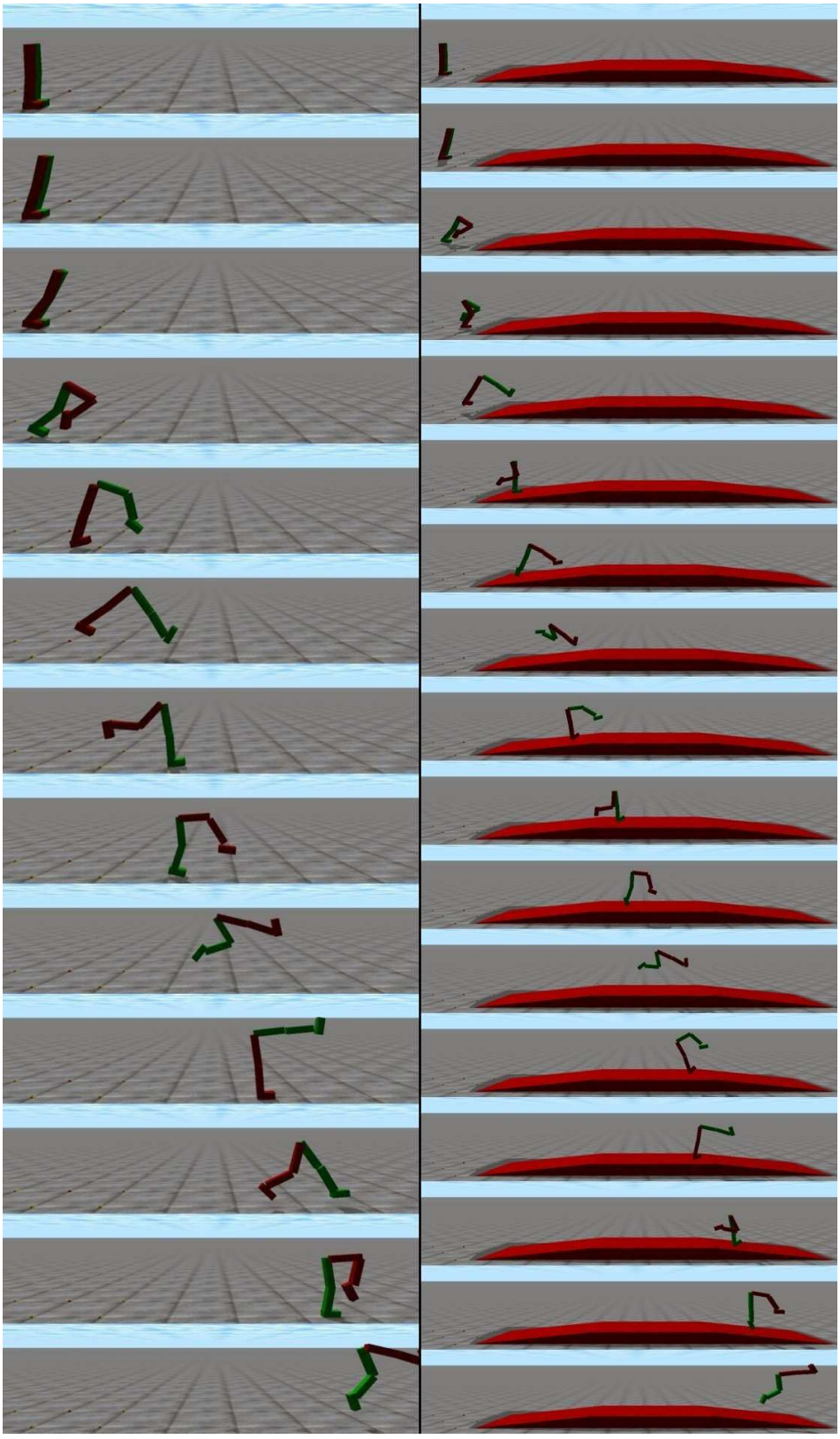

Figura 5.2. *Resultado obtido para o modelo humanóide. À esquerda, o caminhar em terreno plano. À direita, em terreno com rampa.*

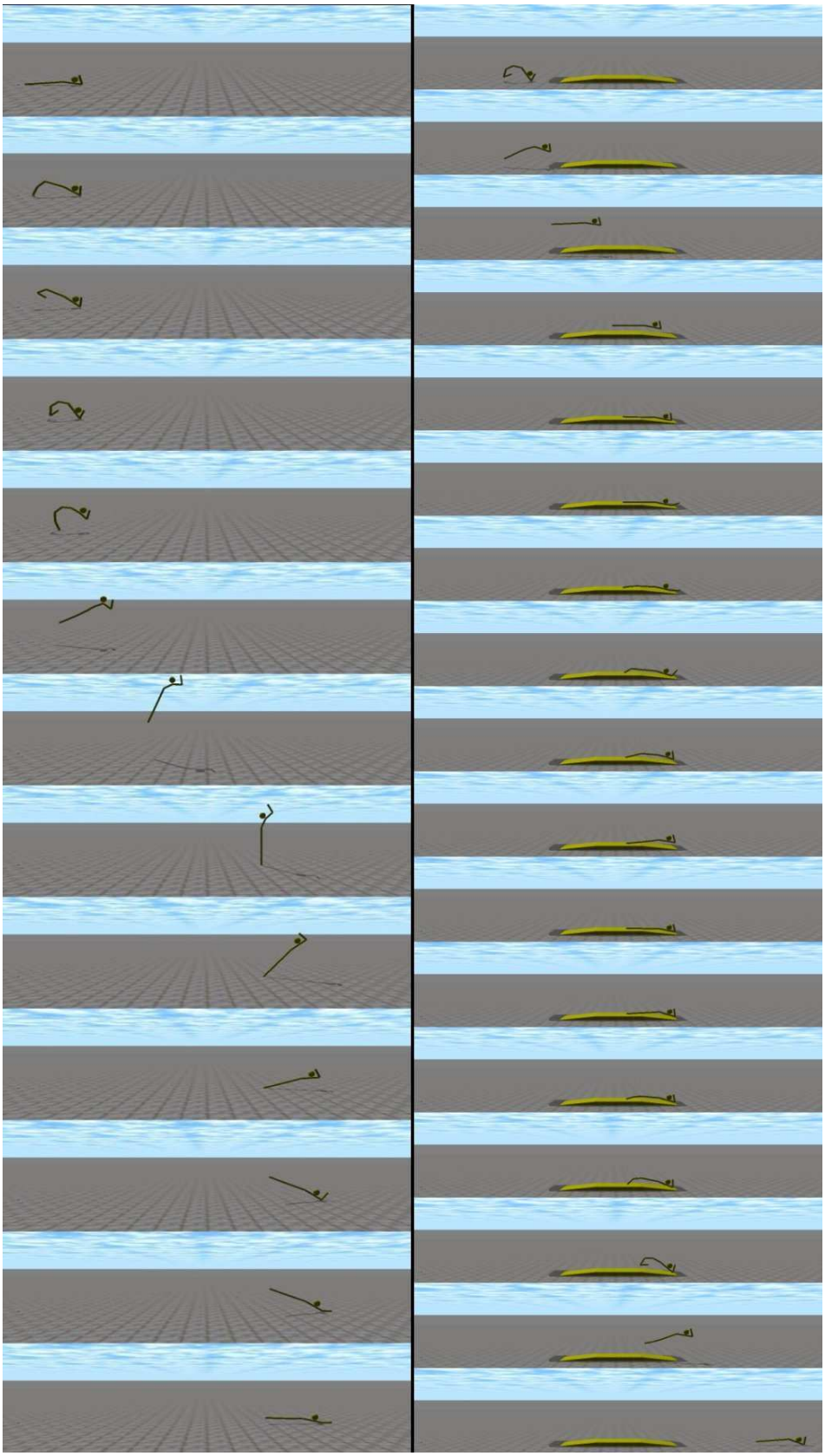

Figura 5.3. *Resultado obtido para o modelo cheetah. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa.*

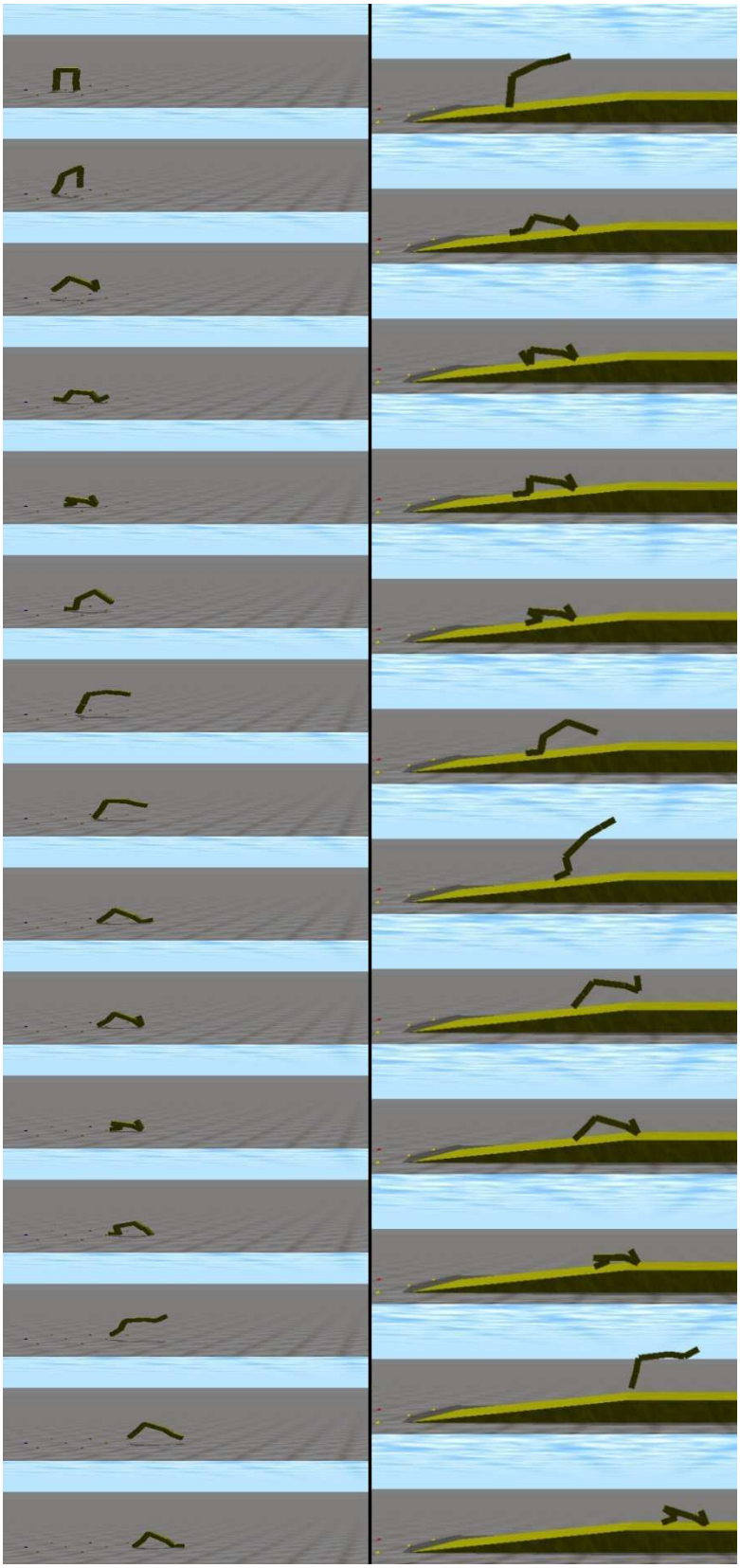

Figura 5.4. *Resultado obtido para o modelo sapo. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa.*

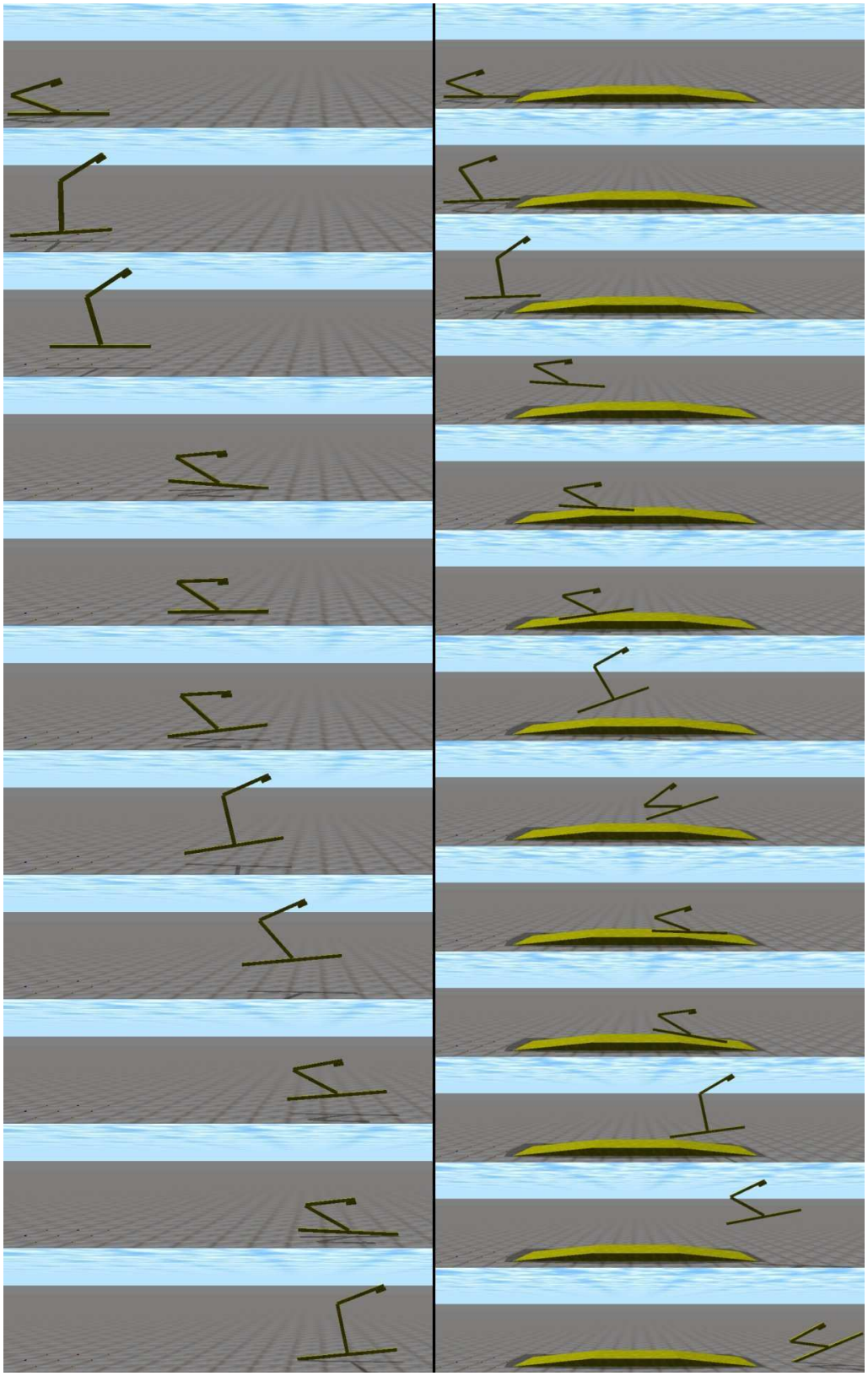

Figura 5.5. *Resultado obtido para o modelo luxo. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa.*

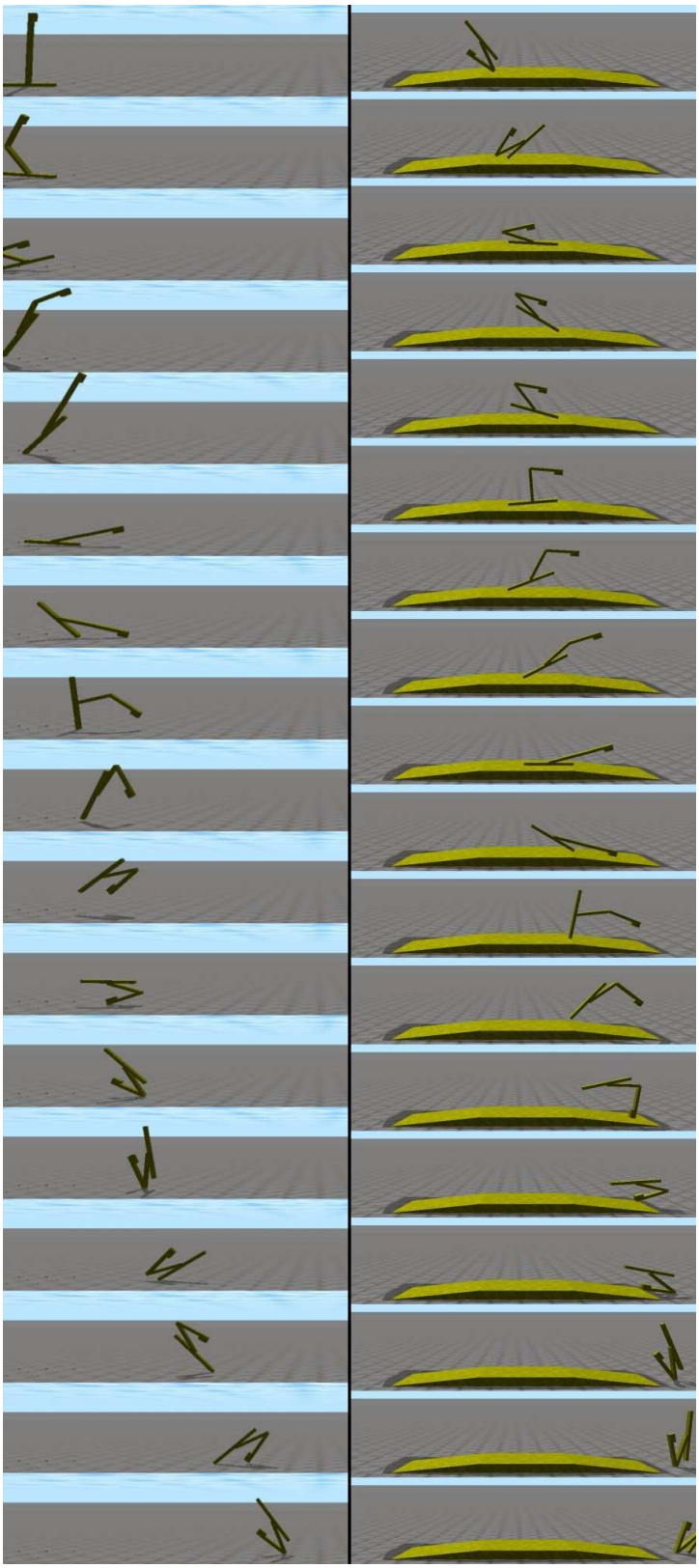

Figura 5.6. *Resultado obtido para o modelo luxo-2. À esquerda, a locomoção em terreno plano. À direita, em terreno com rampa.*

6

Conclusões

Este trabalho propõe um controlador genérico para o problema de animação por dinâmica direta. A solução do problema proposto apresentada nesse trabalho assume que o modelo estudado constitui-se de uma estrutura de corpos rígidos articulados, cujos movimentos são gerados por atuadores internos, com suas forças definidas por um sistema nervoso.

O modelo proposto segue a representação nervo-músculo-óssea, descrita na literatura [8, 9, 15, 18, 27]. O núcleo do sistema nervoso proposto é baseado em um gerador central de padrões (CPG), possuindo ainda um módulo sensorial e um módulo cognitivo. Foi apresentado também, nesse trabalho, além da estrutura do modelo, uma implementação baseada em orientação a objetos, onde todas as classes do sistema são mostradas e discutidas.

6.1. Avaliação dos Resultados

Com relação à validação do modelo proposto, foram feitos estudos de casos em cinco modelos: Humanóide, *cheetah*, sapo, *luxo* e *luxo*-2. Todos eles alcançaram o objetivo de locomoção, o que atesta o sucesso do modelo proposto. Os modelos *cheetah*, sapo e *luxo* mostraram movimentos bastante satisfatórios, perto do esperado. Com relação ao modelo humanóide, a sua evolução mais lenta deveu-se à menor estabilidade desse modelo, quando comparado aos outros modelos. Ainda assim, o humanóide obteve um resultado superior ao de Ok e Kim [20], em número de passos, com um controlador mais genérico.

Como observado nos estudos de casos, o objetivo de gerar movimentos para diferentes estruturas sem a atuação de um animador foi alcançado. Entre os requisitos necessários para a construção de um modelo baseado na representação nervo-músculo-óssea que alcançasse esse objetivo, está a necessidade de se evoluir a topologia e os parâmetros da rede de osciladores neurais, bem como da rede de retroalimentação sensorial, simultaneamente. Tal requisito foi preenchido com a definição do módulo cognitivo descrita no Capítulo 3.

São contribuições do modelo proposto:

- Definição de uma modularização para o modelo nervo-músculo-ósseo;
- Compilação de diversos atributos de modelos propostos em trabalhos anteriores, ficando mais genérico e de maior flexibilidade;
- Definição do Módulo Cognitivo e sua interação com os outros módulos, permitindo maior automatização na geração da animação e tornando indireta a dependência entre o comportamento e a topologia do CPG.

A principal deficiência do controlador proposto é sua forte dependência de um modelo articulado bem definido, exigindo um profundo estudo biomecânico na construção do mesmo, se for esperado um certo tipo de movimento. Essa restrição, entretanto, pode ser vista como uma utilidade quando se pretende avaliar capacidades e limitações de locomoção em seres arbitrários.

6.2. Trabalhos Futuros

Seria interessante fazer um estudo para verificar se a inclusão, na função de avaliação, de um termo que leve em conta o gasto de energia muscular consegue evitar movimentos bruscos, como, por exemplo, a tendência a movimentação por saltos mostrada nos modelos testados e assim obter resultados mais naturais.

Outros modelos, com graus de liberdade nas 3 dimensões em suas juntas, também precisam ser testados com o sistema nervoso proposto, para validar seu sucesso na geração de modos de locomoção arbitrários. Pode-se também testar a implementação de alguns movimentos interessantes, não necessariamente oscilatórios (esses movimentos podem ser vistos como movimentos oscilatórios com freqüências muito baixas), bastando, para isso, que se encontrem outras funções de avaliação ideais para eles. Para tratar movimentos levando-se em conta a existência de grandes obstáculos, pode-se estudar a acoplagem de um Módulo Gerador de Movimentos Discretos, nos moldes do proposto por Taga [27]. Para isso, sensores de visão devem ser considerados também no Módulo Sensorial.

Grzeszczuk e Terzopoulos [7], definem em seu trabalho controladores de baixo e alto nível, onde o primeiro é responsável por gerar um determinado comportamento, como por exemplo, mover-se para frente, e o segundo é uma combinação de vários controladores de baixo nível para resultar em uma animação completa, como correr em uma pista de atletismo, que exige os comportamentos de correr fazendo uma curva e correr em linha reta. Seguindo essa definição, também seria interessante um estudo sobre máquinas de estados finitos para a criação de controladores de alto nível. Isto é, cada estado seria um controlador de baixo nível e a transição entre eles seria definida de acordo com as situações encontradas no ambiente.

Gruau [6] sugeriu a modularização de redes neurais. Pequenos módulos funcionais das mesmas são armazenados em strings (semelhante ao DNA) e, quando combinadas, montam controladores completos. Tal especificação permite a criação de uma estratégia

evolucionária para gerar automaticamente controladores de alto nível segundo idéia apresentada anteriormente utilizando-se máquinas de estado finito.

As redes neurais aprendizes de pose poderiam ser exploradas para o uso em interface humano-máquina. Um caminho seria estudar a possibilidade de utilizar-se uma certa pose entrada pelo animador como função objetivo de treinamento para geração de movimentos.

Referências Bibliográficas

[1] Michael A. Arbib, editor. *The Handbook Of Brain Theory and Neural Networks*. The MIT Press, second edition, 2002.

[2] George A. Bekey. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. The MIT Press, 2005.

[3] M. Van de Panne, R. Kim, and E. Fiume. Virtual wind-up toys for animation. In *Proceedings of Graphics Interface '94*, pages 208–215, 1994.

[4] V. DeSapio, J. Warren, O. Khatib, and S. Delp. Simulating the task-level control of human motion: a methodology and framework for implementation. *The Visual Computer*, 21(5):289–302, 2005.

[5] S. Grillner. Control of locomotion in bipeds, tetrapods and fish. *Brooks VB (ed) Handbook of Phisiology, Sect I: The nervous system, vol. II: Motor Control*, pages 1179–1236, 1981.

[6] Frédéric Gruau. Modular genetic neural networks for 6-legged locomotion. *Artificial Evolution Lecture Notes In Computer Science*, 1063:201–219, 1996.

[7] Radek Grzeszczuk and Demetri Terzopoulos. Automated learning of muscle-actuated locomotion through control abstraction. *Computer Graphics*, 29(Annual Conference Series):63–70, 1995.

[8] K. Hase and N. Yamazaki. Computational evolution of human bipedal walking by a neuro-musculo-skeletal model. In *Proceedings of the Third International Symposium on Artificial Life and Robotics*, pages 174–177, 1998.

[9] Kazunori Hase, Kazuo Miyashita, Sooyol Ok, and Yoshiki Arakawa. Human gait simulation with a neuromusculoskeletal model and evolutionary computation. *Journal of Visualization and Computer Animation*, 14(2):73–92, 2003.

[10] Joseph Laszlo. Controlling Bipedal Locomotion for Computer Animation. MSc Thesis., University of Toronto, Department of Computer Science, 1996.

[11] J. Koza. Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Dept. of Computer Science, Stanford University, June 1990.

[12] Kiyotochi Matsuoka. Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biological Cybernetics*, 52(6):367–376, 1985.

[13] Kazuo Miyashita, Sooyol Ok, and Kazunori Hase. Evolutionary generation of human-like bipedal locomotion. *Mechatronics*, 13(8-9):791–807, 2003.

[14] Franck Multon, Laure France, Marie-Paule Cani, and Gilles Debunne. Computer animation of human walking: a survey. *Journal of Visualization and Computer Animation (JVCA)*, 10:39–54, 1999.

[15] Akinori Nagano, Brian R. Umberger, Mary W. Marzke, and Karin G. M. Gerritsen. Neuromusculoskeletal computer modeling and simulation of upright, straight-legged, bipedal locomotion of australopithecus afarensis (a.l.288-1). *American journal of physical anthropology*, 126(1):2–13, 2005.

[16] J. Thomas Ngo and Joe Marks. Physically realistic motion synthesis in animation. *Evolutionary Computation*, 1(3):235–268, 1993.

[17] J. Thomas Ngo and Joe Marks. Spacetime constraints revisited. *Computer Graphics*, 27(Annual Conference Series):343–350, 1993.

[18] Jiangsheng Ni and Atsuo Kato. A model of neuro-musculo-skeletal system for human locomotion under position constraint condition. *ASME Journal of Biomechanical Engineering*, 125:499–506, 2003.

[19] R. F. Nunes, C. A. Vidal, and J. B. Cavalcante-Neto. Uma representação flexível de controladores para animação fisicamente realista de personagens virtuais. *Proceedings of VIII Symposium on Virtual Reality*, pages 197–208, 2006.

[20] Sooyol Ok and DuckSool Kim. Evolution of the cpg with sensory feedback for bipedal locomotion. In *ICNC (2)*, pages 714–726, 2005.

[21] Alfredo Pina, Eva Cerezo, and Francisco J. Serón. Computer animation: from avatars to unrestricted autonomous actors (a survey on replication and modelling mechanisms). *Computer & Graphics*, 24(2):297–311, 2000.

[22] Arthur Prochazka. The man-machine analogy in robotics and neurophysiology. *Journal of Automatic Control*, 12:4–8, 2002.

[23] T. Reil and P. Husbands. Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6(2):159–168, April 2002.

[24] W.I. Sellers, G.M. Cain, W. Wang, and R.H. Crompton. Stride lengths, speed and energy costs in walking of australopithecus afarensis: using evolutionary robotics to predict locomotion of early human ancestors. *J R Soc Interface*, 2(5):431–441, 2005.

[25] B. F. Skinner. Cognitive science and behaviorism. *British Journal of Psychology*, 76:291–301, 1985.

[26] Russel Smith. Open Dynamics Engine (ODE). http://www.ode.org/, em 15 de fevereiro de 2006.

[27] G. Taga. A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance. *Biological Cybernetics*, 78(1):9–17, 1998.

[28] G. Taga, Y. Yamaguchi, and H. Shimizu. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics*, 65(3):147–159, 1991.

[29] Z. Taha, R. Brown, and D. Wright. Realistic animation of human figures using artificial neural networks. *Medical, Engineering & Physics*, 18(8):662–669, 1996.

[30] Michiel van de Panne and Eugene Fiume. Sensor-actuator networks. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 335–342. ACM Press, 1993.

[31] Darrell Whitley. A genetic algorithm tutorial. *Statistics and Computing*, 4:65–85, 1994.

[32] M. Williamson. Designing Rhythmic Motions using Neural Oscillators. In *IROS*, 1999.

[33] Krister Wolff and Peter Nordin. Learning biped locomotion from first principles on a simulated humanoid robot using linear genetic programming. In *Genetic and Evolutionary Computation – GECCO-2003*, volume 2723 of *LNCS*, pages 495–506. Springer-Verlag, 2003.