Marciel Alberto Gomes

**Algoritmos de adaptação do padrão de marcha utilizando
redes neurais**

Dissertação apresentada à Escola de Engenharia de São Carlos
da Universidade de São Paulo, como parte dos requisitos para
obtenção do título de Mestre em Engenharia Mecânica

Área de Concentração: Dinâmica de Máquinas e Sistemas
Orientador: Prof. Dr. Adriano Almeida Gonçalves Siqueira

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTES
TRABALHOS, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO,
PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca – EESC/USP

G633a Gomes, Marciel Alberto
 Algoritmos de adaptação do padrão de marcha utilizando
 redes neurais / Marciel Alberto Gomes ; orientador
 Adriano Almeida Gonçalves Siqueira. -- São Carlos, 2009.

 Dissertação (Mestrado-Programa de Pós-Graduação em
Engenharia Mecânica e Área de Concentração em Dinâmica de
Máquinas e Sistemas) -- Escola de Engenharia de São
Carlos da Universidade de São Paulo, 2009.

 1. Algoritmos de adaptação. 2. Padrão de marcha.
3. Gerador de trajetórias. 4. Exoesqueleto. 5. Redes
neurais artificiais. 6. Otimização. 7. Método do
gradiente. I. Título.

FOLHA DE JULGAMENTO

Candidato: Licenciado MARCIEL ALBERTO GOMES.

Dissertação defendida e julgada em 09/10/2009 perante a Comissão Julgadora:

Prof. Dr. **ADRIANO ALMEIDA GONÇALVES SIQUEIRA (Orientador)**
(Escola de Engenharia de São Carlos/USP)

APROVADO

Prof. Associado **GLAUCO AUGUSTO DE PAULA CAURIN**
(Escola de Engenharia de São Carlos/USP)

APROVADO

Prof. Dr. **ADELARDO ADELINO DANTAS DE MEDEIROS**
(Universidade Federal do Rio Grande do Norte/UFRN)

APROVADO

Prof. Associado **JONAS DE CARVALHO**
Coordenador do Programa de Pós-Graduação em
Engenharia Mecânica

Prof. Titular **GERALDO ROBERTO MARTINS DA COSTA**
Presidente da Comissão da Pós-Graduação da EESC

Dedicatória

Aos meus avós maternos
Ignácio e Maria Guimarães,
(in memoriam)
exemplos de trabalho, fé e esperança.

Agradecimentos

Ao meu distinto orientador Prof. Dr. Adriano Almeida Gonçalves Siqueira, por toda a atenção dispensada, sugestões e ideias transferidas e por todo o trabalho e paciência no decorrer do desenvolvimento deste trabalho.

Aos amigos do Laboratório de Mecatrônica pela amizade, paciência, companheirismo e colaborações durante a realização deste trabalho.

Ao Guilherme Silveira pela excelente contribuição neste trabalho.

À Daiane por estar sempre ao meu lado e me apoiar em todas os momentos.

À minha mãe por acreditar que o estudo é capaz de modificar as pessoas.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pela concessão da bolsa de mestrado.

“O sucesso nasce do querer, da determinação e persistência em se chegar a um objetivo. Mesmo não atingindo o alvo, quem busca e vence obstáculos, no mínimo fará coisas admiráveis.”

José de Alencar - Escritor Brasileiro

Resumo

GOMES, M. A. **Algoritmos de adaptação do padrão de marcha utilizando redes neurais**. 2009. Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2009.

Este trabalho apresenta o desenvolvimento de algoritmos de adaptação do padrão de marcha com a utilização de Redes Neurais Artificiais para uma órtese ativa para membros inferiores. Trajetórias estáveis são geradas durante o processo de otimização, considerando um gerador de trajetórias baseado no critério do ZMP (*Zero Moment Point*) e no modelo dinâmico do equipamento. Três redes neurais são usadas para diminuir o tempo de cálculo do modelo e da otimização do ZMP, e reproduzir o gerador de trajetórias analítico. A primeira rede aproxima a dinâmica do modelo fornecendo a variação de torque necessária para a realização do processo de otimização dos parâmetros de adaptação da marcha; a segunda rede trabalha no processo de otimização, fornecendo o parâmetro otimizado de acordo com a interação paciente-órtese; a terceira rede reproduz o gerador de trajetórias para um determinado intervalo de tempo do passo que pode ser repetido para qualquer quantidade de passos. Além disso, um controle do tipo torque calculado acrescido de um controle PD é usado para garantir que as trajetórias atuais estejam seguindo as trajetórias desejadas da órtese. O modelo dinâmico da órtese na sua configuração atual, com forças de interação incluídas, é usado para gerar resultados simulados.

Palavras-chave: Algoritmos de adaptação. Padrão de marcha. Gerador de trajetórias. Exoesqueleto. Redes neurais artificiais. Otimização. Método do gradiente.

Abstract

GOMES, M. A. **Gait-Pattern adaptation algorithms using neural network.** 2009. Thesis (Master) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2009.

This work deals with neural network-based gait-pattern adaptation algorithms for an active lower limbs orthosis. Stable trajectories are generated during the optimization process, considering a trajectory generator based on the Zero Moment Point criterion and on the dynamic model. Additionally, three neural networks are used to decrease the time-consuming computation of the model and ZMP optimization and to reproduce the analytical trajectory generator. The first neural network approximates the dynamic model providing the necessary torque variation to gait adaptation parameters process; the second network works in the optimization procedure, giving the adapting parameter according to orthosis-patient interaction; and the third network replaces the trajectory generation for an established step time interval which can be reproduced any time during the walking. Also, a computed torque controller plus the PD controller is designed to guarantee the actual trajectories are following the orthosis desireds trajectories. The dynamic model of the actual active orthosis, with interaction forces included, is used to generate simulation results.

Keywords: Adaptive algorithms. Gait Pattern. Trajectory generator. Exoskeleton. Artificial neural network. Optimization. Gradient descent method.

Publicações

1. GOMES, M. A.; SIQUEIRA, A. A. G., SILVEIRA, G. L. M.; Towards Gait-Pattern Adaptation Algorithms for Exoskeletons based on the ZMP Criterion - In: *The European Control Conference 2009, ECC09*, Budapest, Hungary, 2009, Aug.
2. GOMES, M. A.; SIQUEIRA, A. A. G., SILVEIRA, G. L. M.; Neural Network-Based Gait Adaptation Algorithms For Lower Limbs Active Orthoses - In: *2009 Summer Bioengineering Conference, SBC09*, Squaw Valley, USA, 2009, June.
3. GOMES, M. A.; SIQUEIRA, A. A. G., SILVEIRA, G. L. M.; Gait-Pattern Adaptation Algorithms based on Neural Network for Lower Limbs Active Orthoses - In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS09*, St. Louis, USA, 2009, Oct.
4. GOMES, M. A.; SIQUEIRA, A. A. G., SILVEIRA, G. L. M.; On-line Trajectory Adaptation for Active Lower Limbs Orthoses based on Neural Networks - In: *20th International Congress of Mechanical Engineering, COBEM09*, Gramado, Brasil, 2009, Nov.

Lista de Tabelas

2.1	Parâmetros utilizados para gerar as trajetórias.	25
2.2	Parâmetros dinâmicos da órtese e do paciente.	35

Lista de Figuras

1.1	Evolução do BLEEX - Berkeley University.	5
1.2	Exoesqueletos baseados no BLEEX - Berkeley Bionics.	6
1.3	Exoesqueleto Raytheon - Sarcos Research Corporation.	6
1.4	Exoesqueleto HAL-5 - Tsukuba University.	7
1.5	Exoesqueleto ReWalk - Argo Medical Technologies.	8
1.6	Modelos de órteses <i>RGO</i>	9
1.7	Gravity Balancing Leg Orthosis - Delaware University.	10
1.8	Órtese Lokomat - Swiss Federal Institute of Technology.	11
1.9	Robô bípede Lucy - Vrije Universiteit Brussel.	12
1.10	Órtese RGO e atuador elástico (SEA) - Universidade de São Paulo.	13
2.1	Modelo do robô utilizado neste trabalho (ângulos absolutos).	18
2.2	Fases do caminhar de uma pessoa sem deficiência.	19
2.3	Ciclo de caminhada considerando os parâmetros x_{ed} e x_{sd}	20
2.4	Modelo do robô com as variáveis relacionadas à trajetória do pé.	21
2.5	Modelo do pé do robô.	22
2.6	Modelo do robô com ângulos relativos.	25
2.7	Trajetórias geradas a partir da metodologia baseada no ZMP.	26
2.8	Posição no eixo X do ZMP para as trajetórias geradas.	28
2.9	Convergência do método para um valor inicial próximo ao valor mínimo.	29

2.10	Convergência do método para um valor inicial distante do valor mínimo, passando por fora do domínio.	30
2.11	Análise de convergência do método para diferentes valores iniciais.	31
2.12	Variação da superfície de J_{zmp} em relação a T_c	32
2.13	Variação da superfície de J_{zmp} em relação a D_s	32
2.14	Redefinição da altura máxima do quadril (H_{hmax}).	33
2.15	Superfícies de J_{zmp} com o domínio corrigido pela correlação empírica de H_{hmax}	34
2.16	Interface gráfica do simulador em Matlab (verde = perna direita; vermelho = perna esquerda; azul = tronco).	36
2.17	Animação do movimento com sobreposição.	36
2.18	Trajetórias absolutas, simuladas e desejadas do pé, tíbia, fêmur e torso.	37
2.19	Posição do ZMP simulado.	38
2.20	Posição do ZMP simulado com controle na junta do quadril.	39
3.1	Diagrama do sistema de adaptação baseado na dinâmica inversa.	45
3.2	Trajetórias dos pés - dinâmica inversa.	47
3.3	Trajetórias das tíbias - dinâmica inversa.	48
3.4	Trajetórias dos fêmures - dinâmica inversa.	48
3.5	Trajetórias do torso - dinâmica inversa.	49
3.6	Trajetórias dos pés - dinâmica direta.	50
3.7	Trajetórias das tíbias - dinâmica direta.	50
3.8	Trajetórias dos fêmures - dinâmica direta.	51
3.9	Trajetórias do torso - dinâmica direta.	51
4.1	Modelo de um perceptron.	54
4.2	Análise gráfica do Perceptron.	55
4.3	Modelo de rede perceptron multicamadas.	57
4.4	Saída da rede sem <i>overfitting</i> e com <i>overfitting</i>	64

4.5	Diagrama do sistema com o uso de MLPs.	66
4.6	Geração analítica dos dados de treinamento da rede NN1.	67
4.7	Arquitetura da rede MLP NN1.	67
4.8	Erro da rede NN1 treinada com 5.000 épocas.	68
4.9	Validação cruzada para a rede NN1 com normalização das saídas.	69
4.10	Validação cruzada para a rede NN1 sem normalização das saídas.	70
4.11	Mínimo de $\nabla J_{inv} - T_c$ desejado 0,82 s.	70
4.12	Mínimo de $\nabla J_{inv} - T_c$ desejado 0,87 s.	71
4.13	Rede MLP NN2, treinada <i>online</i>	72
4.14	Erro da rede NN2 depois do primeiro passo.	72
4.15	Geração analítica dos dados de treinamento da rede NN3.	74
4.16	Rede NN3 que substitui o gerador de trajetórias.	75
4.17	Erro de treinamento da rede NN3.	76
4.18	Validação Cruzada para os dados de treinamento da rede NN3.	77
5.1	Trajetórias dos pés - T_c desejado fixo.	81
5.2	Trajetórias das tíbias - T_c desejado fixo.	81
5.3	Trajetórias dos fêmures - T_c desejado fixo.	81
5.4	Trajetórias do torso - T_c desejado fixo.	82
5.5	Trajetórias dos pés - T_c desejado variando em cada passo.	83
5.6	Trajetórias das tíbias - T_c desejado variando em cada passo.	83
5.7	Trajetórias dos fêmures - T_c desejado variando em cada passo.	83
5.8	Trajetórias do torso - T_c desejado variando em cada passo.	84
5.9	Trajetórias dos pés - T_c desejado fixo e adaptação durante o passo.	85
5.10	Trajetórias das tíbias - T_c desejado fixo e adaptação durante o passo.	85
5.11	Trajetórias dos fêmures - T_c desejado fixo e adaptação durante o passo.	85
5.12	Trajetórias do torso - T_c desejado fixo e adaptação durante o passo.	86

5.13	Trajetórias dos pés - T_c desejado variando e adaptação durante o passo.	86
5.14	Trajetórias das tíbias - T_c desejado variando e adaptação durante o passo.	87
5.15	Trajetórias dos fêmures - T_c desejado variando e adaptação durante o passo.	87
5.16	Trajetórias do torso - T_c desejado variando e adaptação durante o passo.	87
A.1	Região estável e margem de estabilidade.	97

Lista de Abreviaturas e Siglas

Termo	-	Descrição
ZMP	-	Zero Moment Point
DARPA	-	Defense Advanced Research Project Agency
EHPA	-	Exoskeleton for Human Performance Augmentation
LEE	-	Lower Extremity Exoskeleton
BLEEX	-	Berkeley Lower Extremity Exoskeleton
HULC	-	Human Universal Load Carrier
HAL	-	Hybrid Assistive Limb
RGO	-	Reciprocating Gait Orthosis
AVC	-	Acidente Vascular Cerebral
ETHZ	-	Swiss Federal Institute of Technology
LSU	-	Lousiana State University
FRS	-	Force Resistor Sensor
RBF	-	Radial Basis Functions
MLP	-	Multilayer Perceptron
EQM	-	Erro Quadrático Médio
SEA	-	Series Elastic Actuator

Lista de Símbolos

Termo	-	Descrição
q_f	-	Ângulo em que o pé deixa o solo.
q_b	-	Ângulo em que o pé toca o solo.
q_{foot}	-	Ângulo absoluto do pé.
q_{femur}	-	Ângulo absoluto do fêmur.
q_{tibia}	-	Ângulo absoluto da tíbia.
q_{torso}	-	Ângulo absoluto do torso.
X_a	-	Coordenada parametrizada do tornozelo (ankle).
$x_a(t)$	-	Posição do tornozelo no eixo X no instante de tempo t .
$z_a(t)$	-	Posição do tornozelo no eixo Z no instante de tempo t .
$\theta_a(t)$	-	Ângulo absoluto do pé no instante de tempo t .
X_h	-	Coordenada parametrizada do quadril (hip).
$x_h(t)$	-	Posição do quadril no eixo X no instante de tempo t .
$z_h(t)$	-	Posição do quadril no eixo Z no instante de tempo t .
$\theta_h(t)$	-	Ângulo absoluto do quadril no instante de tempo t .
T_c	-	Tempo de duração de um passo, em segundos.
T_d	-	Tempo de duração da fase de duplo suporte, em segundos.
T_m	-	Instante de tempo em que o pé atinge a posição mais alta durante a fase de suporte simples.
D_s	-	Metade do comprimento de um passo.
x_{ed}	-	Distância no eixo X entre o quadril e a linha vertical a partir do pé de trás.
x_{sd}	-	Distância no eixo X entre o quadril e a linha vertical a partir do pé da frente.
q_{gs}	-	Ângulo de inclinação do solo em contato com o pé de apoio no início do passo (start).
q_{ge}	-	Ângulo de inclinação do solo em contato com o pé de apoio no fim do passo (end).
L_{ao}	-	Distância no eixo X do início do passo ao ponto de maior altura do pé.
H_{ao}	-	Posição mais alta do pé no eixo Z .
l_{af}	-	Distância no eixo X entre a junta do tornozelo e a ponta do pé.
l_{an}	-	Altura do pé, medida do solo ao tornozelo.
l_{ab}	-	Distância no eixo X entre a junta do tornozelo e o calcanhar.

Termo	-	Descrição
h_{gs}	-	Altura do solo no início do passo, em relação ao pé de suporte.
h_{ge}	-	Altura do solo no fim do passo, em relação ao pé de suporte.
H_{hmin}	-	Altura mínima do quadril durante o passo.
H_{hmax}	-	Altura máxima do quadril durante o passo.
L_{sh}	-	Comprimento da tíbia.
L_{th}	-	Comprimento do fêmur.
d_{zmp}	-	Distância entre o ZMP e o centro da região de estabilidade.
J_{zmp}	-	Funcional para otimização do ZMP.
q	-	Coordenadas generalizadas das juntas do robô.
$M(q)$	-	Matriz de inércia definida positiva.
$C(q, \dot{q})$	-	Vetor de torques centrífugos e de Coriolis.
$G(q)$	-	Vetor de torques gravitacionais.
J_{inv}	-	Funcional para otimização do algoritmo de adaptação baseado na dinâmica inversa.
J_{dir}	-	Funcional para otimização do algoritmo de adaptação baseado na dinâmica direta.
w_{lij}	-	Peso sináptico conectando o j -ésimo neurônio da camada l ao i -ésimo neurônio da camada $(l-1)$.
η	-	Taxa de aprendizagem de métodos numéricos de otimização.
∇f	-	Gradiente da função f (derivada parcial de primeira ordem de f).
$\nabla^2 f$	-	Hessiana da função f (derivada parcial de segunda ordem de f).
t	-	Tempo discretizado.
t_{final}	-	Tempo final de simulação.
τ_i	-	Torque exercido na junta i .
τ_a	-	Torque dos atuadores.
τ_{att}	-	Torque gerado pelo atrito das juntas.
τ_{pac}	-	Torque gerado pela interação paciente-órtese.
$\tau_{pac,at}$	-	Torque exercido pelo paciente (torque paciente-ativo).
$\tau_{pac,pas}$	-	Torque requerido para mover o paciente em modo passivo (torque paciente-passivo).
θ_i	-	Ângulo relativo da junta i .

Sumário

Resumo	xi
Abstract	xiii
Publicações	xv
Lista de Tabelas	xvii
Lista de Figuras	xix
Lista de Abreviaturas e Siglas	xxiii
Lista de Símbolos	xxv
1 Introdução	1
1.1 Motivação	1
1.2 Revisão bibliográfica	4
1.3 Descrição do trabalho	13
1.4 Disposição dos capítulos	14
2 Gerador de trajetórias com o critério de estabilidade ZMP	17
2.1 Modelagem cinemática do exoesqueleto	17
2.2 Metodologia para obtenção das trajetórias	18
2.2.1 Padrão de caminhada	18
2.2.2 Trajetória do pé	20
2.2.3 Trajetória do quadril	22
2.2.4 Implementação computacional	24
2.3 Otimização da trajetória em relação ao ZMP	27
2.3.1 Método utilizado na minimização	28
2.3.2 Restrições impostas à minimização	29
2.3.3 Análise da convergência em relação a D_s e T_c	31

2.4	Simulação considerando o gerador de trajetórias com critério ZMP	35
2.4.1	Resultados da posição do ZMP utilizando o Matlab	37
2.5	Conclusão sobre o capítulo	39
3	Algoritmos de adaptação do padrão de marcha	41
3.1	Modelagem dinâmica do exoesqueleto	42
3.2	Algoritmo de adaptação baseado na dinâmica inversa	43
3.3	Algoritmo de adaptação baseado na dinâmica direta	46
3.4	Simulações dos algoritmos	47
3.4.1	Resultados com algoritmo baseado na dinâmica inversa	47
3.4.2	Resultados com algoritmo baseado na dinâmica direta	49
3.5	Conclusão sobre o capítulo	52
4	Adaptação do padrão de marcha utilizando redes neurais	53
4.1	Redes Perceptron de camada única	53
4.1.1	Análise matemática do perceptron	54
4.2	Redes Perceptron Multicamadas (MLP)	55
4.2.1	Algoritmo de treinamento Backpropagation	56
4.2.2	Algoritmo de treinamento Levenberght-Marquardt	61
4.2.3	MLPs como aproximadores universais de funções	63
4.2.4	Aspectos de treinamento de redes MLP	63
4.3	Sistema de Adaptação do Padrão de Marcha com uso de Redes MLP	65
4.3.1	Rede MLP aplicada à adaptação do padrão de marcha	66
4.3.2	Rede MLP aplicada ao gerador de trajetórias	73
4.4	Conclusão sobre o capítulo	78
5	Resultados	79
5.1	Simulação do sistema	80
5.1.1	Caso I: Adaptação no final do passo	80
5.1.2	Caso II: Adaptação durante o passo	84
6	Conclusões	89
	Referências Bibliográficas	91
A	Critério de estabilidade ZMP	97
B	Interpolação por <i>splines</i> cúbicas	99

Capítulo 1

Introdução

1.1 Motivação

Nas últimas décadas, diversas pesquisas no campo da robótica têm sido desenvolvidas visando tornar a vida do ser humano mais confortável em vários aspectos. Dentre essas pesquisas, incluem-se aquelas que visam auxiliar pessoas que tenham algum tipo de deficiência, entre elas a deficiência motora. Essas pesquisas fazem parte da chamada *engenharia de reabilitação* que tem se tornado um campo promissor para o desenvolvimento de equipamentos mecatrônicos que ajudam no processo de recuperação de pacientes com deficiência, (FERRIS et al. , 2005). Tal campo é extremamente dinâmico sendo beneficiado por novas tecnologias cognitivas, avanços nas áreas de robótica, engenharia de materiais, sistemas inteligentes, sensoramento e medicina. Além disso, o campo da engenharia de reabilitação possui grande demanda ao redor do mundo. Somente no Brasil, estima-se que existam cerca de 8 milhões de pessoas com deficiência motora (incapaz, com alguma ou grande dificuldade de caminhar ou subir escadas), (IBGE , 2003). Essas pessoas necessitam de aparelhos de suporte durante sua reabilitação ou mesmo para suporte permanente, no caso de pessoas que tiveram sequelas irreversíveis devido a acidentes ou doenças que comprometeram seus movimentos. Exercícios de reabilitação funcional são muito utilizados, como parte do processo de recuperação do paciente, para estimular a parte remanescente intacta do sistema nervoso central a reconhecer e a reaprender funções perdidas do organismo (*plasticidade neural*). Entretanto, a maioria dos métodos utilizados na reabilitação de pacientes com deficiência requer acompanhamento constante de especialistas e o esforço que o paciente exerce durante o processo de recuperação não é realizado de uma forma gradativa bem definida. Nesse sentido, as pesquisas voltadas para a reabilitação vêm tentando criar equipamentos *bi-*

oinspirados, ou seja, baseados em sistemas biológicos como o corpo humano, e que reproduzam com precisão os movimentos e funções do mesmo, ou parte dele, e que auxiliem essas pessoas na recuperação dos movimentos.

O uso desses equipamentos é de extrema importância para tornar a vida de um paciente mais confortável, podendo evitar por exemplo, que o mesmo fique sentado por um tempo prolongado em uma cadeira de rodas, o que pode provocar sérios problemas de saúde, tais como osteoporose, atrofia muscular, circulação prejudicada, etc.

Entre esses equipamentos, existem aqueles denominados *Wearable Robots*, que pode ser traduzido como robôs que podem ser “vestidos”. PONS (2008) faz uma classificação relativa a esses equipamentos levando em conta a função que o mesmo realiza quando em cooperação com o ser humano:

- *Empowering Robotics Exoskeletons*: Originalmente chamados de *extenders*, KAZEROONI (1989), são definidos como uma classe de robôs que complementa o corpo humano, aumentando a capacidade de operação do mesmo;
- *Orthotic Robots*: Estrutura mecânica que reproduz algumas funções de membros do corpo humano com o propósito de restaurar a perda de funções.
- *Prosthetic Robots*: Aparelho eletromecânico que substitui membros amputados.

Dentre essas três diferentes denominações, os exoesqueletos são aqueles que apresentam interações físicas e cognitivas mais intensas com o usuário. O aspecto mais singular dos exoesqueletos é que a cinemática dos mesmos é determinada sobre a anatomia do corpo humano. Existe uma correspondência entre todos as juntas dos membros do corpo humano com as juntas do robô, o que torna essa relação o aspecto chave para alcançar uma ergonomia na interface humano-robô.

Tradicionalmente, um exoesqueleto artificial é uma simples armação que cobre o corpo, similar a de um inseto. No campo da engenharia de reabilitação, exoesqueleto é conhecido como um equipamento eletromecânico usado para melhorar as habilidades humanas, podendo levá-las a níveis acima do padrão. Nesse caso, existe uma transferência efetiva de força entre o ser humano e o robô, resultando em uma interação física entre os dois, geralmente alcançada através de forças externas e/ou internas do sistema.

Nesse sentido, os *empowering exoskeletons* são utilizados para multiplicar as forças exercidas pelo usuário, ou seja, no sistema de forças externas a estrutura mecânica do exoesqueleto age

como um suporte de carga e somente uma pequena parte das forças é exercida pelo usuário. Por outro lado, orthotic exoskeletons, utilizados para uma compensação funcional dos membros do corpo humano, trabalham com o princípio de forças internas, no qual a força é aplicada somente entre o exoesqueleto e o membro.

Nas pesquisas com exoesqueletos utilizados na reabilitação de membros inferiores, há pelo menos duas preocupações a serem consideradas para o seu correto funcionamento: a geração de uma trajetória de referência, geralmente baseada no caminhar de uma pessoa considerada saudável, e a adaptação dessa trajetória através da interação com o usuário. A ideia de adaptação é fazer com que o paciente tenha interação com o equipamento, de forma que este forneça potência somente quando necessário, ou seja, o usuário é forçado a treinar seus músculos e suas movimentações de forma gradativa. Este tipo de interação necessita de atuadores com rigidez controlável, para gerar movimentos antropomórficos e evitar lesões ao usuário e danos ao equipamento. (PRATT et al. , 1997) e (PALUSKA , 2000) utilizaram atuadores elásticos para desenvolvimento de seus robôs bípedes, respectivamente Flamingo e M2. Esses atuadores reproduzem certas características dos músculos humanos, e basicamente são compostos de um motor elétrico ou hidráulico ligado em série ao efetuator através de molas, (ROBINSON et al. , 1999; PRATT e WILLIAMSON , 1995). Com um controle de força apropriado, é possível alterar a impedância do atuador de acordo com a necessidade do usuário. Com essas características, é possível implementar controles de adaptação dos movimentos com a utilização de algoritmos capazes de captar as informações dadas pelo usuário e transmiti-las de forma precisa aos atuadores do equipamento.

Neste trabalho é dado enfoque ao desenvolvimento e implementação computacional de algoritmos de adaptação do padrão de marcha, em particular no caso em que é feito uso da dinâmica inversa do modelo, permitindo efetuar a interação paciente-órtese de modo que a força exercida pelo paciente seja a mínima necessária para mover-se. Um gerador de trajetórias estáveis com o critério de estabilidade ZMP (*Zero Moment Point*) também é apresentado, e redes neurais artificiais são utilizadas para aproximar as saídas do gerador e efetuar a otimização do algoritmo, diminuindo o tempo computacional gasto no processo como um todo.

1.2 Revisão bibliográfica

Exoesqueletos, Robôs Bípedes e suas características

Os esforços despendidos na construção de equipamentos que permitem simular e reproduzir os movimentos humanos têm mostrado resultados bastante satisfatórios em vários centros de pesquisas do mundo. No caso particular de exoesqueletos, já existem modelos desenvolvidos em alguns países, para uso na reabilitação e outros para uso militar. Estes últimos têm como finalidade aumentar a capacidade física de soldados, permitindo que os mesmos possam carregar pesos acima da sua capacidade normal, sem que tenham que fazer esforço adicional para isso. Com o mesmo equipamento, um bombeiro pode ser capaz de retirar pessoas de escombros, ou resgatá-las de terrenos acidentados onde veículos com rodas não conseguem se locomover.

Uma das mais importantes pesquisas dessa área surgiu no ano de 2000, quando o Departamento de Defesa dos EUA (*Defense Advanced Research Projects Agency*, DARPA) lançou o programa EHPA (*Exoskeleton for Human Performance Augmentation*), com o objetivo de criar um exoesqueleto para melhorar a capacidade de carga, mobilidade e autonomia de soldados em missões de guerra. Cinco elementos básicos foram enfatizados pela DARPA no desenvolvimento do exoesqueleto: estrutura, força, controle, desempenho em vários tipos de terreno e biomecânica. A locomoção em vários tipos de terrenos foi um dos critérios enfatizados pelo EHPA pois apenas 3% da superfície do mundo têm acessibilidade através de um veículo, mas cerca de 85% podem ser acessados por uma pessoa a pé. Em 2003 a *Universidade da Califórnia*, Berkeley e a *Sarcos Research Corporation* foram selecionadas para desenvolver esse projeto.

O exoesqueleto BLEEX (*Berkeley Lower Extremity Exoskeleton*) foi desenvolvido na Universidade da Califórnia e teve seus primeiros testes realizados em 2003, sob a supervisão do Prof. Kazerooni, diretor do *Berkeley Robotics and Human Engineering Laboratory*. Na sua primeira versão, o BLEEX, inicialmente denominado apenas de *LEE* (*Lower Extremity Exoskeleton*), Figura 1.1(a), era equipado com um motor a gasolina com autonomia de apenas 15 minutos. Os atuadores hidráulicos eram expostos nas pernas, o que tornava o equipamento extremamente volumoso. O BLEEX 1, Figura 1.1(b), com uma estrutura mais enxuta, foi exposto pela primeira vez com um usuário que caminhava a uma velocidade de até 3 km/h, carregando uma carga de aproximadamente 35 kg.

Com o avanço no desenvolvimento do BLEEX 1, uma nova versão foi construída e melhorias foram realizadas, principalmente em relação ao tamanho do equipamento. O BLEEX2, Figura

(a) LEE

(b) BLEEX 1

(c) BLEEX 2

Figura 1.1: Evolução do BLEEX - Berkeley University.

1.1(c), foi apresentado com uma estrutura pesando apenas 14 kg, sem a exposição de cabos, circuitos e atuadores, (GUIZZO e GOLDSTEIN, 2005). O BLEEX2 utiliza atuadores hidráulicos acionados por uma bomba ligada a um pequeno motor a gasolina, disposto em uma mochila que o próprio usuário carrega nas costas. Cerca de 40 sensores disponibilizam constantemente informações a um computador central que calcula o que é necessário fazer para distribuir o peso de tal forma que o usuário não sinta o peso excedente, (ZOSS et al., 2005, 2006; KAZEROONI, 2005; KIM et al., 2004).

O BLEEX2 foi concluído em 2005 e é capaz de transportar aproximadamente 70 kg, além do peso do usuário, a uma velocidade de 8 km/h. Sua arquitetura o faz ser altamente manobrável, robusto, leve e durável, podendo transcender as limitações humanas. O controlador apresenta grande sensibilidade ao controle de força e torque, permitindo ao usuário diversas manobras.

Com bons resultados no desenvolvimento do BLEEX, os cientistas envolvidos nesse projeto fundaram a *Berkeley Bionics*, empresa que tem desenvolvido vários outros exoesqueletos baseados no BLEEX para fins militares. O ExoHiker, Figura 1.2(a), o ExoClimber, Figura 1.2(b) e o Human Universal Load Carrier (HULC), Figura 1.2(c) têm tamanhos e pesos reduzidos em relação ao BLEEX, e todos possuem a mesma tecnologia, e com as características de suportar pesos acima do padrão. Em 2007, no entanto, a Berkeley Bionics iniciou um trabalho para construir exoesqueletos com a finalidade de beneficiar pessoas com deficiência nos membros inferiores.

Já o grupo de pesquisa liderado pelo roboticista Stephen Jacobsen, da Sarcos Research

Figura 1.2: Exoesqueletos baseados no BLEEX - Berkeley Bionics.

Corporation, desenvolveu o Raytheon, Figuras 1.3(a) e 1.3(b), um exoesqueleto que também tem a finalidade de prover mais força e potência aos soldados.

Figura 1.3: Exoesqueleto Raytheon - Sarcos Research Corporation.

O Raytheon é considerado o mais poderoso exoesqueleto construído até o momento. Em uma apresentação realizada em 2005, um usuário conseguiu carregar 85 kg sem sentir a carga total. Assim como o BLEEX, sensores de força, nesse caso especialmente desenvolvidos pela Sarcos, instalados em pontos específicos e em contato com o corpo do usuário fornecem informações para que o controle do exoesqueleto seja feito de forma harmônica com o movimento do usuário.

Também como no BLEEX, a unidade de potência deste sistema é um motor de combustão interna, localizado na parte traseira do robô. Um dos aspectos que torna o Raytheon mais eficiente em relação ao BLEEX é a presença de membros superiores, permitindo que o usuário tenha um exoesqueleto completo atuando junto ao próprio corpo.

Nessa mesma linha de pesquisa, mas focando no auxílio de pessoas com deficiência motora, destaca-se o desenvolvimento do HAL-5 (*Hybrid Assistive Limb*), outro projeto de grande relevância, supervisionado pelo Prof. Yoshiyuki Sankai da Universidade de Tsukuba, Japão. O HAL-5 foi apresentado primeiramente na 2005 World Expo e atualmente encontra-se na quinta geração de exoesqueletos projetados para auxiliar pessoas com deficiência, Figuras 1.4(a) e 1.4(b).

(a) Usuário suportando peso com o HAL-5

(b) Usuário com o HAL-5

Figura 1.4: Exoesqueleto HAL-5 - Tsukuba University.

O HAL-5 possui uma estrutura composta de ligas níquel molibdênio e duralumínio e seu mecanismo é protegido por uma envoltória de plástico. O equipamento é fixado na parte externa do usuário, permitindo que ele permaneça em pé, caminhe e suba escadas. Assim como no Raytheon, o HAL-5 também possui atuação nos membros superiores. Além disso, o exoesqueleto possibilita que o usuário carregue até 40 kg a mais que normalmente conseguiria, por até 10 minutos. Ao contrário do BLEEX e do Raytheon, a fonte de energia utilizada para acionar os atuadores, que são pequenos motores DC nos ombros, cotovelos, joelhos e quadril, provém de baterias híbridas níquel-metal e de lítio que podem funcionar por até 2 horas e 40 minutos. O computador e os sistemas de comunicação do tipo sem fio são dispostos em um pequeno compartimento anexado ao cinto, excluindo-se a necessidade do usuário carregar uma mochila com

os equipamentos de controle. Dois sistemas de controle são utilizados para acionar o exoesqueleto: o primeiro, chamado sistema *biocibernético* (termo cunhado pelo Prof. Sankai), monitora as correntes elétricas do corpo do usuário, conhecidas como sinais mio elétricos, e acionam os motores quando o usuário pretende realizar algum movimento, através de um controle assistido de potência (HAYASHI et al. , 2005; LEE e SANKAI , 2002); o segundo faz com que o usuário e o exoesqueleto movam-se de forma suave e sincronizada, denominado controle por sequência de fases, que gera uma sequência de movimento assistidos pela transição de movimentos fundamentais chamados de fase, (KIM e SANKAI , 2005b) e (KIM e SANKAI , 2005). A calibração desse sistema pode levar até dois meses para cada pessoa e o custo total do equipamento chega a US\$14.000,00.

Figura 1.5: Exoesqueleto ReWalk - Argo Medical Technologies.

Em Israel, a empresa *Argo Medical Technologies*, especializada em tecnologia aplicada em medicina, desenvolveu um exoesqueleto que ajuda na locomoção de paraplégicos, auxiliando-os a caminhar, subir escadas, sentar, descer terrenos íngremes e até mesmo dirigir. O dispositivo leva o nome de ReWalk, Figura 1.5 e foi desenvolvido pelo Dr. Amit Goffer, diretor da *Argo*. Foram investidos neste projeto cerca de US\$30 milhões. O exoesqueleto traz um leve suporte para os braços, integrado a uma armadura que conta com motores de corrente contínua, baterias recarregáveis, uma série de sensores e um sistema de controle que se comunica com um computador. O exoesqueleto ReWalk detecta os movimentos da parte superior do corpo e, a partir daí, o processo de caminhada é iniciado. Para auxiliar na estabilidade e segurança do procedimento, o paciente faz uso de muletas, sendo que o exoesqueleto pode ser usado durante o dia inteiro, além de substituir outros aparelhos dentro de casa ou no centro de reabilitação.

Muito se tem trabalhado, também na busca de soluções para a reabilitação de pessoas com deficiência, com as chamadas *órteses*. Em geral, uma órtese fornece sustentabilidade ao corpo do usuário permitindo que este permaneça em pé sem precisar do auxílio de cadeira de rodas. Na área médica, o termo órtese refere-se aos aparelhos ortopédicos com a finalidade de estabilizar um segmento ou articulação, possibilitando dissipar parte da energia mecânica imposta a este. O uso de órteses possibilita que o paciente possa utilizar parte da capacidade funcional dos membros sem que esta utilização atrapalhe no processo de cicatrização, ajudando assim no processo de recuperação do paciente. Um tipo de órtese muito utilizado na reabilitação é a órtese RGO (*Reciprocating Gait Orthosis*), Figuras 1.6(a) e 1.6(b). As órteses do tipo RGO são órteses para pessoas com paraplegia que possibilitam o caminhar através de um mecanismo que utiliza o movimento de uma perna para provocar o movimento da outra perna.

(a) Órtese RGO

(b) Usuário utilizando uma órtese RGO

Figura 1.6: Modelos de órteses RGO.

Apesar de, originalmente, uma órtese não possuir atuação eletromecânica, e sim mecânica, algumas pesquisas têm proposto o uso de órteses híbridas, as quais possuem eletrodos que permitem a estimulação elétrica funcional dos músculos do paciente para reduzir os esforços necessários durante o caminhar (JASPERS et al. , 1996; PÉREZ-ORIVE e MAYAGOITIA , 1994; TO et al. , 2005). Em TO et al. (2005), é descrito um sistema de controle que coordena o travamento das juntas do joelho e do calcanhar de uma órtese, enquanto a estimulação neuromuscular funcional é utilizada para controlar as juntas desbloqueadas. As órteses híbridas possuem algumas desvantagens em comparação com órteses mecânicas simples, pois a colocação e retirada dos eletrodos é uma operação demorada e de custo elevado e o usuário deve pressionar um botão a cada passo a ser realizado, (JASPERS et al. , 1996). Além disso, a fadiga dos

músculos do paciente geralmente aparece rapidamente devido ao esforço realizado, fazendo com que a força de atuação fornecida para locomover a órtese sofra redução significativa, (TO et al. , 2005).

Figura 1.7: Gravity Balancing Leg Orthosis - Delaware University.

BANALA e AGRAWAL (2005), da Universidade de Delaware, propõem uma órtese com atuação eletromecânica que usa o balanço da gravidade para reduzir os torques requeridos pelos atuadores durante o movimento. O protótipo de apenas uma perna, Figuras 1.7(a) e 1.7(b), faz com que se reduza o torque que o paciente deve aplicar nas juntas da perna para realizar um passo, pois em baixas velocidades, a força da gravidade é que tem maior influência na dinâmica do sistema.

Em JEZERNIK et al. (2003), uma órtese robótica denominada Lokomat, Figura 1.8(a) e 1.8(b), é utilizada na reabilitação de pacientes com lesão medular ou que sofreram um AVC (*Acidente Vascular Cerebral*). O equipamento foi construído sob a supervisão do Prof. Morari, do *Automatic Control Laboratory do Swiss Federal Institute of Technology (ETHZ)*, em Zurique na Suíça. O dispositivo é instalado em uma esteira rolante e o paciente se locomove sobre a esteira utilizando um compensador de peso. O treinamento regular de pacientes é realizado impondo um padrão de marcha fixo, obtido pela análise do caminhar de uma pessoa considerada saudável, e um controle de posição das juntas da órtese robótica. Porém, é importante garantir que o paciente esteja efetivamente andando, e não apenas tendo a sua perna movida passivamente pelo aparelho de locomoção. Esta ideia levou ao desenvolvimento de três algoritmos de adaptação do padrão de marcha, (JEZERNIK et al. , 2004; RIENER et al. , 2005). Esses algoritmos

(a) Órtese Lokomat.

(b) Paciente utilizando o Lokomat.

Figura 1.8: Órtese Lokomat - Swiss Federal Institute of Technology.

possibilitam ao paciente ter a capacidade de alterar o padrão de marcha conforme o seu grau de locomoção voluntária. O primeiro deles produz uma adaptação do padrão de marcha estimando primeiramente os torques de interação paciente-órtese e então adaptando as trajetórias angulares de modo a reduzir tais torques de interação. O segundo algoritmo relaciona o torque de interação homem-robô com a mudança necessária de aceleração das trajetórias angulares. No terceiro algoritmo, um controle de impedância gera uma relação entre os torques de interação e os desvios de posição permitidos. Este procedimento permite adaptação direta das trajetórias angulares através do desvio de posição medido ou dos torques de interação estimados.

Os algoritmos de adaptação de marcha citados acima foram definidos em RIENER et al. (2005) considerando a órtese robótica Lokomat, que apresenta uma base fixa ao solo. Mas, no caso de um exoesqueleto que se moverá livremente no espaço, a geração de trajetórias estáveis deve ser considerada. Em HUANG et al. (2001) é apresentada uma metodologia de geração de trajetórias estáveis considerando o ZMP (*Zero Moment Point*). O ZMP é o ponto no solo onde a resultante das forças de reação do solo age no robô (VUKOBRATOVIC e JURICIC, 1969). Para que o sistema seja estável, é necessário que o ZMP esteja dentro da área de suporte do robô, região de contato do robô com o solo. Em MOUSAVI et al. (2008), a mesma metodologia é utilizada para gerar trajetórias combinadas estáveis, realizando simulações em plano inclinado com o robô bípede subindo e descendo escadas, obtendo resultados satisfatórios com relação a estabilidade. Além disso, para determinar a trajetória dos pés e da cintura é utilizado o método de interpolação por *splines* cúbicas, pois este tipo de interpolação reduz a possibilidade

de oscilação da trajetória em comparação com a interpolação polinomial além de também gerar uma curva derivável até segunda ordem. Porém, não é feita referência a metodologia utilizado para determinação dos parâmetros que compõem uma trajetória estável.

Outros inúmeros trabalhos citam outras abordagens para o desenvolvimento e geração de trajetórias estáveis para robôs bípedes. Em VANDERBORGHT et al. (2008) é proposto um gerador de trajetórias baseado em um pêndulo invertido para o robô bípede *Lucy*, Figura 1.9. O robô bípede *Lucy* é desenvolvido na Vrije Universiteit Brussel, na Bélgica, e seu sistema segue uma estratégia de geração de trajetórias em tempo real, utilizando a equação de momento angular para a dinâmica natural dos membros superiores, e com isso determinar o comportamento dos membros inferiores. KITAMURA et al. (1990) propõe um gerador de trajetórias no qual o movimento do centro de gravidade do robô é simulado por um sistema de pêndulo invertido, e a cinemática inversa do robô é usada para calcular as posições do centro de gravidade, e da extremidade do pé do robô.

Figura 1.9: Robô bípede Lucy - Vrije Universiteit Brussel.

FUKUDA et al. (1997) utiliza uma rede neural auto-adaptativa com algoritmos genéticos para encontrar o melhor ponto de estabilidade ZMP, gerando assim uma trajetória estável que permite que o robô caminhe sobre superfícies de diferentes inclinações.

No Laboratório de Mecatrônica da USP-São Carlos está sendo desenvolvido um exoesqueleto para membros inferiores para auxílio de pessoas deficientes. O exoesqueleto em questão é

baseado em uma órtese de reciprocção, RGO, Figura 1.10(a), e corresponde a uma órtese de marcha recíproca LSU (*Lousiana State University*), desenvolvida no *Ontário Crippled Children's Centre*, em Toronto, no início da década de 1970. Indicada para pacientes paraplégicos portadores de lesões medulares, a órtese é composta de duas KAFOs (*Knee-Ankle-Foot-Orthosis*), conhecidas como goteiras, confeccionadas em termoplástico unidas a uma banda pélvica metálica por barras também metálicas. Dois cabos de reciprocção conectam as articulações pélvicas, permitindo movimentos alternados de flexão e extensão do quadril, (GUIZZO e GOLDSTEIN, 2005). Atuadores elásticos em série são utilizados para a movimentação das juntas do exoesqueleto, (JARDIM e SIQUEIRA, 2008). Tal dispositivo, Figura 1.10(b), apresenta características ideais para a utilização neste tipo de equipamento: controle de força, impedância controlável (possibilidade de impedância baixa), absorção de impactos, baixo atrito e largura de banda que se aproxima da movimentação muscular. Sensores de força do tipo FRS (*Force Resistor Sensor*) são instalados em pontos de contato com o pé para se determinar os instantes de contato com o solo.

Figura 1.10: Órtese RGO e atuador elástico (SEA) - Universidade de São Paulo.

1.3 Descrição do trabalho

Este trabalho é parte integrante de uma linha de pesquisa desenvolvida no Laboratório de Mecatrônica da EESC-USP que tem como principal objetivo desenvolver um exoesqueleto para membros inferiores destinado a reabilitação de pessoas com deficiências no caminhar. Em parti-

cular, este trabalho visa a descrição e implementação computacional de algoritmos de adaptação do padrão de marcha baseados em redes neurais artificiais, com garantia de estabilidade baseada no critério do ZMP.

O trabalho está dividido em três partes: desenvolvimento e simulação de um gerador de trajetórias estáveis com critério de estabilidade, desenvolvimento de algoritmos de adaptação do padrão de marcha, e desenvolvimento e implementação do sistema de adaptação do padrão de marcha com a utilização de redes neurais artificiais visando melhorar a performance do sistema.

A primeira parte trata do desenvolvimento e simulação de um gerador de trajetórias estáveis, considerando o critério de estabilidade ZMP. As trajetórias dos pés e quadril são geradas a partir do método de interpolação por *splines* cúbicas considerando pontos chaves obtidos da análise do caminhar padrão de uma pessoa sem deficiência.

A segunda parte mostra a implementação e simulação dos algoritmos de adaptação baseados em JEZERNIK et al. (2004) e RIENER et al. (2005). Neste trabalho são considerados dois algoritmos, sendo o primeiro baseado na dinâmica inversa do robô e o segundo baseado na dinâmica direta. São mostrados resultados simulados com esses algoritmos, considerando o gerador de trajetória proposto na primeira parte.

A principal contribuição deste trabalho é apresentada na terceira parte, onde é proposta a utilização de redes neurais artificiais no sistema de adaptação da trajetória. Também é apresentada uma rede neural que reproduz o gerador de trajetórias estáveis, citado na primeira parte.

1.4 Disposição dos capítulos

No **Capítulo 2** é apresentado o gerador de trajetórias estáveis com critério *ZMP*. Simulações foram realizadas em Matlab e seus resultados são apresentados com algumas considerações sobre a convergência do ZMP.

O **Capítulo 3** descreve os algoritmos de adaptação do padrão de marcha e os processos de otimização dos funcionais que permitem a adaptação do padrão de marcha. Alguns resultados simulados também são mostrados.

O sistema de adaptação proposto, com o uso de redes neurais, é apresentado no **Capítulo 4**, com algumas considerações sobre redes neurais com arquitetura MLP.

No **Capítulo 5** são apresentados os resultados de testes simulados obtidos no trabalho.

Por fim, no **Capítulo 6** são apresentadas as considerações e conclusões do trabalho, e propostas para trabalhos futuros.

Capítulo 2

Gerador de trajetórias com o critério de estabilidade ZMP

Robôs bípedes em geral devem ter um alto grau de mobilidade, especialmente quando se movem em terrenos irregulares, sobem escadas, e em ambientes que possuam obstáculos. Muitas questões como o critério de estabilidade, análise dinâmica e a própria aplicação de um robô bípede precisam ser consideradas no desenvolvimento desse tipo de robô, em particular, na geração das trajetórias para as juntas que o constitui.

A geração de trajetórias através de métodos simples tem sido focada em muitos trabalhos relacionados a robôs bípedes. Em HUANG et al. (2001) é proposto um método para produzir trajetórias dos pés e do quadril, através do uso do método de interpolação por *splines* cúbicas, incluindo condições de terreno, restrições da condição de estabilidade e relação entre o caminhar padrão e as especificações dos atuadores. Esse método é bastante simples e capaz de obter curvas adequadas em relação a pontos chave da trajetória de uma determinada junta.

Neste capítulo é apresentado um gerador de trajetórias baseados na proposta de HUANG et al. (2001) considerando o critério de estabilidade ZMP e com os parâmetros utilizados na implementação do nosso exoesqueleto.

2.1 Modelagem cinemática do exoesqueleto

O modelo cinemático do robô possui sete juntas, sendo três para cada perna, referentes aos ângulos absolutos dos pés, q_f e q_b , tíbia e fêmur, e uma junta relativa ao movimento do quadril.

O modelo pode ser visto na Figura 2.1, sendo que uma junta no eixo YZ , q_s , referente ao ângulo de balanço lateral pode ser acrescentada para efeitos de uma análise tridimensional do sistema. Neste trabalho, a análise do exoesqueleto foi realizada considerando apenas o *plano sagital* (XZ), como forma de simplificar o modelo.

Figura 2.1: Modelo do robô utilizado neste trabalho (ângulos absolutos).

As posições angulares desejadas de cada junta são derivadas a partir do gerador de trajetórias, apresentado em detalhes na Seção 2.2.

2.2 Metodologia para obtenção das trajetórias

2.2.1 Padrão de caminhada

Considera-se o modelo de exoesqueleto como sendo um robô bípode com tronco (HUANG et al. , 2001). Para que um robô bípode seja capaz de caminhar em várias condições de terrenos, ele deve ser capaz de realizar vários tipos de movimentos nos pés. Observando que o fenômeno de locomoção bípode humana é um fenômeno periódico, um ciclo completo pode ser dividido em duas fases, sendo uma fase de apoio duplo e uma fase de apoio simples. A primeira fase começa quando o calcanhar do pé a frente toca o chão e termina quando a ponta do pé de trás deixa o solo. A segunda fase ocorre quando apenas um pé está fixo no chão e o outro encontra-se no ar, em fase de balanço. Aqui consideramos a fase de suporte duplo como sendo 20% do ciclo total do caminhar, (HUANG et al. , 2001). A Figura 2.2 mostra dois ciclos completos, considerando

dois passos, um para cada perna.

Figura 2.2: Fases do caminhar de uma pessoa sem deficiência.

Através de cálculos de cinemática inversa, podemos determinar as trajetórias de todas as outras juntas se conhecermos as trajetórias do tornozelo e do quadril.

Considerando um sistema cartesiano inercial XZ , baseado no plano sagital, pode-se parametrizar a trajetória do pé e do quadril da seguinte forma:

- (a) *Trajetória do pé:* durante o ciclo do caminhar, os deslocamentos vertical e horizontal do tornozelo são definidos por um vetor $X_a = [x_a(t), z_a(t), \theta_a(t)]^T$, sendo $(x_a(t), z_a(t))$ a posição do tornozelo, e $\theta_a(t)$ o ângulo formado entre o pé e o plano horizontal.
- (b) *Trajetória do quadril:* é definida pelo vetor $X_h = [x_h(t), z_h(t), \theta_h(t)]^T$, sendo $(x_h(t), z_h(t))$ a posição do quadril, e $\theta_h(t)$ o ângulo formado entre o tronco e o plano horizontal.

A distância entre o quadril e o eixo Z do sistema de coordenadas fixo é denotado para instantes do começo e fim da fase de duplo suporte, por x_{ed} e x_{sd} , respectivamente, mostrados na Figura 2.3. O parâmetro T_c representa o tempo de duração de cada passo, em segundos [s], incluindo as fases de simples suporte e duplo suporte. O parâmetro T_d é o tempo de duração da fase de duplo suporte. Neste trabalho, o valor de T_d foi fixado em 20% de T_c , (HUANG et al. , 2001).

Figura 2.3: Ciclo de caminhada considerando os parâmetros x_{ed} e x_{sd} .

A seguir é mostrada a modelagem completa para cada uma das juntas citadas acima.

2.2.2 Trajetória do pé

Considerando um período de tempo de um passo igual a T_c , temos que o k -ésimo passo ocorre entre o instante kT_c e o instante $(k+1)T_c$, com $k = 0, 1, 2, \dots, K$, sendo K o número total de passos. Define-se o início do k -ésimo passo como sendo o instante em que o calcanhar de um pé deixa o solo, terminando quando este toca o solo novamente. Define-se também q_b como sendo o ângulo em que o pé deixa o solo e q_f como o ângulo que ele chega ao solo.

Assumindo que a sola do pé direito está completamente em contato com o solo entre os instantes kT_c e $(k+1)T_c + T_d$, pode-se admitir as seguintes condições angulares mostradas em (2.1):

$$\theta_a = \begin{cases} q_{gs}(k), & t = kT_c \\ q_b, & t = kT_c + T_d \\ -q_f, & t = (k+1)T_c \\ -q_{ge}(k), & t = (k+1)T_c + T_d, \end{cases} \quad (2.1)$$

sendo $q_{gs}(k)$ e $q_{ge}(k)$ os ângulos de inclinação do solo em contato com o pé de apoio, particularmente iguais a zero para uma superfície plana, como vistos na Figura 2.4.

Figura 2.4: Modelo do robô com as varáveis relacionadas à trajetória do pé.

Para o deslocamento horizontal, x_a , e vertical do pé, z_a , admitem-se as condições abaixo:

$$x_a = \begin{cases} kD_s, & t = kT_c \\ kD_s + l_{an} \sin q_b + l_{af}(1 - \cos q_b), & t = kT_c + T_d \\ kD_s + L_{ao}, & t = kT_c + T_m \\ (k+2)D_s - l_{an} \sin q_f - l_{ab}(1 - \cos q_f), & t = (k+1)T_c \\ (k+2)D_s, & t = (k+1)T_c + T_d, \end{cases} \quad (2.2)$$

$$z_a = \begin{cases} h_{gs}(k) + l_{an}, & t = kT_c \\ h_{gs} + l_{af} \sin q_b + l_{an} \cos q_b, & t = kT_c + T_d \\ H_{ao}, & t = kT_c + T_m \\ h_{ge} + l_{ab} \sin q_f + l_{an} \cos q_f, & t = (k+1)T_c \\ h_{ge}(k) + l_{an}, & t = (k+1)T_c + T_d \end{cases} \quad (2.3)$$

sendo (L_{ao}, H_{ao}) a posição mais alta do pé durante o passo, D_s a metade do comprimento de cada passo, $kT_c + T_m$ o instante de tempo em que o pé ocupa a posição mais alta, l_{an} a altura do pé, l_{ab} a distância no eixo X entre a junta do tornozelo e o calcanhar, e l_{af} a distância no eixo X entre a junta do tornozelo e a ponta do pé, Figura 2.5. Os parâmetros h_{ge} e h_{gs} referem-se à altura entre o solo e o pé de apoio, no início e no fim do passo, respectivamente.

Figura 2.5: Modelo do pé do robô.

Similar ao processo do caminhar humano, a velocidade angular do pé, $\dot{\theta}_a$, é igual a zero em alguns instantes, incluindo no começo e fim da sua trajetória, e da mesma forma, $\dot{x}_a = 0$ e $\dot{z}_a = 0$. Desse modo, temos:

$$\begin{cases} \dot{\theta}_a(kT_c) = 0 \\ \dot{\theta}_a((k+1)T_c + T_d) = 0, \end{cases} \quad (2.4)$$

$$\begin{cases} \dot{x}_a(kT_c) = 0 \\ \dot{x}_a((k+1)T_c + T_d) = 0, \end{cases} \quad (2.5)$$

$$\begin{cases} \dot{z}_a(kT_c) = 0 \\ \dot{z}_a((k+1)T_c + T_d) = 0. \end{cases} \quad (2.6)$$

Utilizando-se dessas condições é possível gerar uma trajetória suave através do método de interpolação por *splines* cúbicas, que gera uma trajetória derivável até segunda ordem para todo intervalo de tempo. Nesse método, em vez de um só polinômio, utilizam-se vários polinômios, um para cada par de abscissas consecutivas, ajustando-se os coeficientes de forma a permitir uma transição suave de um polinômio para outro, garantindo continuidade sempre que possível, (ver Apêndice B). Portanto, considerando valores iniciais de D_s e T_c , e variando os parâmetros q_{gs} , q_{ge} , h_{gs} , h_{ge} , q_b , q_f , H_{ao} e L_{ao} é possível gerar diferentes trajetórias para os pés.

2.2.3 Trajetória do quadril

A trajetória do quadril é de extrema importância para a geração de trajetórias com critério ZMP.

O movimento do quadril na direção do eixo Z dificilmente afeta a posição do ZMP, e podemos assumir este valor como sendo uma pequena variação entre o ponto mais alto e o ponto mais baixo sendo que o primeiro ocorre na metade da fase de apoio simples e o segundo na metade

da fase de apoio duplo. Para facilitar a análise da trajetória do quadril, o parâmetro $\theta_h(t)$ foi considerado como constante. Desta forma z_h pode ser dado por:

$$z_h = \begin{cases} H_{hmin}, & t = kT_c + 0,5T_d \\ H_{hmax}, & t = kT_c + 0,5(T_c - T_d) \\ H_{hmin}, & t = (k+1)T_c + 0,5T_d. \end{cases} \quad (2.7)$$

O movimento do quadril na direção do eixo X é o que mais afeta a estabilidade do caminhar de um robô bípede no plano sagital. Em HUANG et al. (2001) é proposta a escolha de uma trajetória que possua a maior margem de estabilidade, considerando o cálculo do ZMP de várias trajetórias para o quadril na direção do eixo X . Como visto anteriormente, um processo completo de caminhada consiste em uma fase inicial partindo do repouso, uma fase permanente com velocidade constante e a fase final que consiste no retorno ao repouso. Todas essas fases podem ser obtidas pela variação das constantes que determinam a trajetória tanto do pé quanto do quadril. Para se obter a trajetória do quadril na direção do eixo X na fase permanente temos que assumir as seguintes condições:

$$x_h = \begin{cases} kD_s + x_{ed}, & t = kT_c \\ (k+1)D_s - x_{sd}, & t = kT_c + T_d \\ (k+1)D_s + x_{ed}, & t = (k+1)T_c. \end{cases} \quad (2.8)$$

Da definição de interpolação por *splines* cúbicas, as seguintes condições para as derivadas devem ser satisfeitas:

$$\begin{cases} \dot{x}_h(kT_c) = \dot{x}_h(kT_c + T_c) \\ \ddot{x}_h(kT_c) = \ddot{x}_h(kT_c + T_c). \end{cases} \quad (2.9)$$

Utilizando esse tipo de interpolação e variando x_{sd} e x_{ed} em um intervalo coerente podemos obter uma série de trajetórias e escolher a de maior margem de estabilidade definida da seguinte forma:

$$\begin{aligned} \min \quad & d_{zmp}(x_{sd}, x_{ed}) \\ & x_{sd} \in (0; 0,5D_s), x_{ed} \in (0; 0,5D_s), \end{aligned} \quad (2.10)$$

sendo d_{zmp} a distância mínima entre o ZMP e o centro da região de estabilidade formada pelo polígono convexo dos pontos de contato (ver Apêndice A). No caso deste trabalho, a região de estabilidade é dada por um segmento de reta, visto que estamos trabalhando apenas com o

plano sagital. Como existem apenas dois parâmetros x_{sd} e x_{ed} é possível obter soluções para a Eq. 2.10 através de algoritmos de busca. O método utilizado neste trabalho para obter essas soluções é descrito na Seção 2.3.

2.2.4 Implementação computacional

Utilizando a metodologia descrita acima, foi desenvolvido um programa em Matlab para gerar as trajetórias θ_a , x_a , z_a , z_h e x_h . A trajetória x_h depende dos parâmetros x_{ed} e x_{sd} , selecionados pelo método de otimização em relação ao ZMP descrito na Seção 2.3 .

As curvas geradas para o pé são válidas apenas no intervalo em que o pé não está parado, sendo necessária a redefinição do seguimento da trajetória no qual o pé fica parado com todas as suas variáveis constantes. Isso acarreta em uma curva descontínua para a aceleração dos parâmetros referentes aos pés, o que por consequência gera descontinuidade do ZMP. Esse problema foi resolvido com a utilização de um polinômio de 5º grau, como na Eq. 2.11, para redefinir o início da trajetória (logo depois da fase de apoio simples do pé) e o fim da trajetória (logo antes da fase de apoio simples) gerada pela *spline* . Com esse polinômio, é possível garantir que a curva das acelerações seja contínua para todo o tempo.

$$P(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5. \quad (2.11)$$

A partir das trajetórias geradas, aplica-se a cinemática inversa para encontrar os ângulos absolutos da tíbia, q_{tibia} , e do fêmur, q_{femur} , dados por:

$$\begin{cases} \cos(\theta_2) &= (dz^2 + dx^2 - L_{th}^2 - L_{sh}^2)/(2L_{th}L_{sh}) \\ \sin(\theta_2) &= \sqrt{1 - \cos(\theta_2)^2} \\ \theta_2 &= \text{atan2}(\sin(\theta_2), \cos(\theta_2)), \end{cases} \quad (2.12)$$

$$\begin{cases} \sin(q_{tibia}) &= [dz[L_{sh} + L_{th} \cos(\theta_2)] - dxL_{th} \sin(\theta_2)]/[L_{th}^2 \sin(\theta_2)^2 + (L_{sh} + L_{th} \cos(\theta_2))^2] \\ \cos(q_{tibia}) &= [dx + L_{th} \sin(q_{tibia}) \sin(\theta_2)]/[L_{sh} + L_{th} \cos(\theta_2)] \\ q_{tibia} &= \text{atan2}(\sin(q_{tibia}), \cos(q_{tibia})) \\ q_{femur} &= q_{tibia} + \theta_2, \end{cases} \quad (2.13)$$

sendo θ_2 é o ângulo do fêmur relativo à tíbia. Os parâmetros d_x e d_z são distâncias medidas em

Figura 2.6: Modelo do robô com ângulos relativos.

relação aos eixos do sistema de coordenadas para efeito de resolver as equações da cinemática inversa, Figura 2.2.4.

Para que as equações em 2.13 tenham soluções é necessário que a distância entre o tornozelo e a cintura seja menor do que a soma vetorial dos comprimentos da tíbia e do fêmur, ou seja, $|\overline{AH}| \leq L_{sh} + L_{th}$, sendo A o índice relativo ao tornozelo e H à cintura, L_{sh} o comprimento da tíbia e L_{th} o comprimento do fêmur.

As Figuras de 2.7(a) a 2.7(d) apresentam as trajetórias geradas pelo procedimento descrito acima, considerando os parâmetros da trajetória mostrados na Tabela 2.1. O ângulo do torso foi considerado constante igual a 81° , visto que esse valor permitiu melhores resultados em termos de estabilidade do sistema. Em HUANG et al. (2001), o ângulo sugerido para o torso é de 90° .

Tabela 2.1: Parâmetros utilizados para gerar as trajetórias.

q_{gs} (rad)	q_{ge} (rad)	h_{gs} (m)	h_{ge} (m)	q_b (rad)	q_f (rad)	H_{ao} (m)	x_{ed} (m)
0	0	0	0	-0,20	0,20	0,16	0,49
L_{ao} (m)	T_d (s)	T_c (s)	T_m (s)	H_{hmax} (m)	H_{hmin} (m)	D_s (m)	x_{sd} (m)
0,25	0,18	0,90	0,45	0,895	0,875	0,50	0,25

(a) Trajetória do pé.

(b) Trajetória da tíbia.

(c) Trajetória do fêmur.

(d) Trajetória do torso.

Figura 2.7: Trajetórias geradas a partir da metodologia baseada no ZMP.

Para o cálculo do ZMP são necessárias também as velocidades e acelerações angulares de todos as juntas, obtidas a partir da derivação da equação vetorial fechada do sistema, obtendo-se as seguintes equações:

$$\left\{ \begin{array}{l} \dot{q}_{femur} = [-(\dot{x}_a - \dot{x}_h) \cos(q_{tibia}) - (\dot{z}_a - \dot{z}_h \sin(q_{tibia}))]/[L_{th} \sin(q_{tibia} - q_{femur})] \\ \dot{q}_{tibia} = [(\dot{x}_a - \dot{x}_h) - L_{th} \dot{q}_{femur} \sin(q_{femur})]/[L_{sh} \sin(q_{tibia})] \\ \ddot{q}_{femur} = [L_{th}(\dot{q}_{femur}^2) \cos(q_{tibia} - q_{femur}) + L_{sh}(\dot{q}_{tibia}^2) - (\ddot{x}_a - \ddot{x}_h) \\ \cos(q_{tibia}) - (\ddot{z}_a - \ddot{z}_h \sin(q_{tibia}))]/[L_{th} \sin(q_{tibia} - q_{femur})] \\ \ddot{q}_{tibia} = [-L_{th}(\ddot{q}_{femur} \sin(q_{femur}) - (\dot{q}_{femur}^2) \cos(q_{femur})) - L_{sh}(\dot{q}_{tibia}^2) \\ \cos(q_{tibia}) + (\ddot{x}_a - \ddot{x}_h)]/[L_{sh} \sin(q_{tibia})], \end{array} \right. \quad (2.14)$$

Com a finalidade de calcular o ZMP o programa foi modificado utilizando a ferramenta simbólica do Matlab. A partir dos ângulos de cada junta do modelo e suas respectivas velocidades e acelerações, obtidas como descrito acima, este programa calcula o ZMP. A Figura 2.8 mostra o valor do ZMP e os valores máximos e mínimos permitidos. Note que o ZMP permanece dentro da faixa aceitável durante as fases de suporte simples e duplo. Entretanto, no momento de transição do suporte de um pé para o outro, o ZMP deixa a faixa por um curto período de tempo. Um processo de otimização do ZMP foi implementado para melhorar a procura pelo ZMP ótimo, e é apresentado na próxima seção. Com este procedimento o ZMP permanece na faixa aceitável a maior parte possível do tempo do passo.

2.3 Otimização da trajetória em relação ao ZMP

Nesta seção é apresentada a otimização da trajetória com relação ao ZMP considerando a variação nos parâmetros x_{ed} e x_{sd} . Para que o ZMP permaneça a maior parte do tempo próximo ao centro do polígono de apoio, foi definido um funcional, J_{zmp} , que é a somatória das distâncias, ao quadrado, do ZMP ao centro do polígono de apoio, ao longo do tempo do passo, Eq. 2.15.

$$J_{zmp}(x_{ed}, x_{sd}) = \frac{\sum_{n=1}^p d_{zmp}^2}{p}, \quad (2.15)$$

sendo p o número de pontos ao longo da trajetória nos quais o d_{zmp} é calculado. Nesse caso, temos uma reta de apoio, que é definida como sendo o pé de apoio na fase de apoio simples e a reta formada entre a ponta de um pé e o calcanhar do outro na fase de apoio duplo.

Figura 2.8: Posição no eixo X do ZMP para as trajetórias geradas.

Os parâmetros x_{ed} e x_{sd} são considerados na minimização do funcional da Eq. 2.15, pois estes são os parâmetros que mais influenciam na estabilidade do sistema. Dessa maneira, quando o funcional J_{zmp} é minimizado em relação a x_{ed} e x_{sd} encontramos a trajetória mais estável possível para determinados parâmetros definidos na Seção 2.2. Caso esse procedimento não fosse realizado, o tempo necessário de processamento para verificar todas as possibilidades seria extremamente dispendioso.

2.3.1 Método utilizado na minimização

O método escolhido para minimização foi o método do *gradiente descendente*, devido a sua boa convergência depois de ajustar alguns de seus parâmetros e implementar algumas modificações, além do menor número de iterações para o cálculo de J_{zmp} , processo relativamente demorado. O método é definido da seguinte maneira:

$$\bar{X}_{n+1} = \bar{X}_n + \eta \nabla J_{zmp}(x_{ed}, x_{sd}), \quad (2.16)$$

sendo $\bar{X}_n$ o vetor contendo os valores de x_{ed} e x_{sd} no passo n , η a taxa de otimização e $\nabla J_{zmp}(x_{ed}, x_{sd})$ o gradiente do funcional com relação a x_{ed} e x_{sd} .

Figura 2.9: Convergência do método para um valor inicial próximo ao valor mínimo.

É possível notar na Figura 2.9 que o método converge satisfatoriamente na região do ponto inicial, se a variação de J_{zmp} for pequena.

É necessário observar que os valores de x_{ed} e x_{sd} estão normalizados através da constante $\frac{100}{D_s}$, ou seja, no gráfico $x_{ed,real} = x_{ed}D_s/100$ e $x_{sd,real} = x_{sd}D_s/100$.

2.3.2 Restrições impostas à minimização

Como mostrado na Seção 2.2, J_{zmp} é restrito a $x_{sd} \in (0; 0,5D_s)$, $x_{ed} \in (0; 0,5D_s)$, Eq. 2.10. Portanto, é necessário restringir o método a esse polígono de valores para não se obter valores inconsistentes de x_{ed} e x_{sd} .

Um grande inconveniente da aplicação do método citado acima é que existem regiões pertencentes ao polígono de valores válidos em que não é possível calcular J_{zmp} . Isso ocorre em decorrência de não haver solução de q_{tibia} e q_{femur} para alguns valores de x_{ed} e x_{sd} .

Se o valor resultante da otimização pertencer a uma região fora do domínio da função, a otimização não tem continuidade, pois $\nabla J_{zmp}(x_{ed}, x_{sd})$ é zero devido a definição de J_{zmp} constante igual a zero quando não é possível calcular J_{zmp} . Esse fato não é raro, dependendo do ponto inicial escolhido, pois ∇J_{zmp} pode ser muito grande.

Para solucionar esse problema foi definida a seguinte estratégia: se o resultado da otimização

for tal que $J_{zmp} = 0$ e $\nabla J_{zmp} = 0$, o método tenta utilizar, como alternativa e para que o método não fique restrito a um quadrado, um dos versores normais ao último $\nabla J_{zmp} \neq 0$, multiplicado pelo módulo deste, até que o resultado pertença ao domínio do funcional.

Figura 2.10: Convergência do método para um valor inicial distante do valor mínimo, passando por fora do domínio.

A Figura 2.10 mostra que a estratégia utilizada para resolver o problema da minimização com restrição de domínio é satisfatória, fazendo com que o gradiente retorne ao domínio do funcional. Importante ressaltar que deve-se analisar a convergência do método caso as variáveis recebam valores diferentes dos valores utilizados neste trabalho.

Outra implementação feita no método de minimização foi definir um vetor com direção padrão $\nabla J_{zmp} = (0, 5; 0, 5)$ para o caso de o ponto inicial cair fora do domínio de J_{zmp} . Da observação das superfícies definidas por J_{zmp} para diferentes parâmetros de trajetória, o domínio de J_{zmp} situa-se predominantemente no canto superior direito do polígono de valores válidos para x_{ed} e x_{sd} . Esse procedimento empírico possibilita uma enorme chance de convergência do método para valores iniciais fora do domínio.

A Figura 2.11 mostra a convergência do método após as modificações citadas anteriormente. Podemos observar que o método converge em poucas iterações, o que não implica em pouco tempo. Mesmo assim, temos uma evolução, pois como citado em HUANG et al. (2001), para o processo de otimização era necessário um “exaustivo trabalho computacional”.

Figura 2.11: Análise de convergência do método para diferentes valores iniciais.

2.3.3 Análise da convergência em relação a D_s e T_c

Essa análise é essencial pois os parâmetros D_s e T_c determinam uma característica de extremo interesse, a velocidade do caminhar. Variando esses parâmetros podemos oferecer uma gama de padrões de marcha ao paciente.

A Figura 2.12(a) mostra a superfície de J_{zmp} para os mesmos parâmetros da Tabela 2.1, ou seja, $D_s = 0,5$ e $T_c = 0,9$, enquanto a Figura 2.12(b) considera o parâmetro $T_c = 0,8$. Pode-se concluir que nem a forma nem o domínio de J_{zmp} variam muito, e é possível constatar apenas que J_{zmp} apresenta valores mais altos para pontos distantes do valor ideal.

Em contrapartida, nas Figuras 2.13(a) e 2.13(b) observa-se que o domínio de J_{zmp} reduz-se muito com a variação de D_s , tornando inviável a minimização de J_{zmp} . Isso ocorre devido a não existência de solução para q_{tibia} e q_{femur} em uma região muito maior. Em outras palavras, existem poucas trajetórias que mantêm a cintura na mesma altura e desenvolva um passo de comprimento de 0,6 m, Figura 2.13(b).

Para solucionar esse problema para que, dados valores de D_s e T_c definidos a partir de uma entrada do usuário, seja possível gerar as trajetórias e minimizá-las com relação ao ZMP, uma nova análise da metodologia é necessária. Da análise do padrão de caminhar bípede e de testes efetuados é possível perceber que o domínio de J_{zmp} está intimamente ligado ao parâmetro

(a) Superfície de J_{zmp} para $D_s = 0,5$ m e $T_c = 0,9$ s. (b) Superfície de J_{zmp} para $D_s = 0,5$ m e $T_c = 0,8$ s.

Figura 2.12: Variação da superfície de J_{zmp} em relação a T_c .

(a) Superfície de J_{zmp} para $D_s = 0,5$ m e $T_c = 0,9$ s. (b) Superfície de J_{zmp} para $D_s = 0,6$ m e $T_c = 0,9$ s.

Figura 2.13: Variação da superfície de J_{zmp} em relação a D_s .

H_{hmax} (altura máxima da cintura definido na Seção 2.2). Quanto maior o comprimento do passo desenvolvido pelo bípede, menor é a altura máxima que a cintura pode atingir. Uma observação importante a ser feita é que não é viável fazer uma análise da superfície de J_{zmp} para cada conjunto de parâmetros, pois para gerar a superfície completa leva-se um período de tempo bastante razoável.

Portanto, é necessário determinar uma estimativa para o parâmetro H_{hmax} em função de D_s . Uma estratégia é definir a altura máxima da cintura igual à altura do triângulo isósceles de lado $L_{th} + L_{sh}$ e base D_s mais a altura do tornozelo, Figura 2.14. A Eq. 2.17 define a altura H_{hmax} com essa nova abordagem.

$$H_{hmax} = \left(\sqrt{(L_{sh} + L_{th})^2 - \left(\frac{D_s}{2}\right)^2} + l_{an} \right) \alpha. \quad (2.17)$$

sendo $\alpha = 1 - \left(\frac{D_s - 0,5}{0,5}\right)^2$, um parâmetro utilizado para corrigir o valor de H_{hmax} quando é realizada a otimização do ZMP.

Figura 2.14: Redefinição da altura máxima do quadril (H_{hmax}).

As Figuras 2.15(a), 2.15(b) e 2.15(c) mostram as superfícies de J_{zmp} obtidas variando-se os valores de D_s e considerando o domínio corrigido pela correlação empírica de H_{hmax} . Note que mesmo alterando-se o valor de D_s , o domínio permanece dentro de uma faixa aceitável para a minimização.

(a) Superfície de J_{zmp} para $D_s = 0,6$ m e $T_c = 0,9$ s.

(b) Superfície de J_{zmp} para $D_s = 0,55$ m e $T_c = 0,9$ s.

(c) Superfície de J_{zmp} para $D_s = 0,45$ m e $T_c = 0,9$ s.

Figura 2.15: Superfícies de J_{zmp} com o domínio corrigido pela correlação empírica de H_{hmax} .

2.4 Simulação considerando o gerador de trajetórias com critério ZMP

Para avaliar na prática a eficiência do gerador de trajetórias descrito anteriormente, foi desenvolvido um simulador para o conjunto paciente-órtese utilizando o software Matlab. As matrizes dinâmicas do sistema são geradas através da ferramenta de cálculo simbólico do Matlab (*Symbolic Toolbox*), utilizando-se o método de modelagem de Lagrange. Os parâmetros dinâmicos da órtese, mostrados na Tabela 2.2, foram obtidos através do modelo desenhado em software *Solid Edge*. Também são apresentados os parâmetros do paciente considerados na simulação, obtidos de SIQUEIRA et al. (2008), considerando um indivíduo de 85 kg, e altura de 1,74 m.

Tabela 2.2: Parâmetros dinâmicos da órtese e do paciente.

Órtese		Paciente	
$M_{total,ort}$ (kg)	4,8	$M_{total,pac}$ (kg)	85
$L_{total,ort}$ (m)	1,0	$L_{total,pac}$ (m)	1,74
Massa do Membro (kg)			
$M_{coxa,ort}$	0,95	$M_{coxa,pac}$	8,5
$M_{perna+pe,ort}$	0,72	$M_{perna+pe,pac}$	5,2
$M_{tronco,ort}$	1,49	$M_{tronco,pac}$	57,6
Comprimento de Membro (m) - direção z			
$L_{coxa,ort}$	0,39	$L_{coxa,pac}$	0,39
$L_{perna+pe,ort}$	0,49	$L_{perna+pe,pac}$	0,49
$L_{tronco,ort}$	0,12	$L_{tronco,pac}$	0,87
Centro de Massa do Membro (m) - direção z			
$CM_{coxa,ort}$	0,18	$CM_{coxa,pac}$	0,17
$CM_{perna+pe,ort}$	0,17	$CM_{perna+pe,pac}$	0,30
$CM_{tronco,ort}$	0,11	$CM_{tronco,pac}$	0,33
Momento de Inércia do Membro (kgm^2) - plano sagital			
$I_{coxa,ort}$	0,03	$I_{coxa,pac}$	0,14
$I_{perna+pe,ort}$	0,02	$I_{perna+pe,pac}$	0,22
$I_{tronco,ort}$	0,06	$I_{tronco,pac}$	10,73

A interface gráfica do simulador é mostrada na Figura 2.16. Note que esta interface foi produzida utilizando os comandos básicos de geração de retas e pontos, resultando em interface simples. De acordo com a modelagem considerada, o paciente acompanha a órtese, ou seja, as trajetórias das juntas da órtese e do paciente são iguais.

A Figura 2.17 mostra uma animação do movimento do sistema paciente-órtese, para uma simulação de dois passos, considerando $D_s = 0,45$ e $T_c = 0,9$ para o passo inicial e no passo seguinte esses valores são $D_s = 0,57$ e $T_c = 0,85$, aumentando assim a velocidade do padrão

de caminhada em aproximadamente 34%. Somente a órtese é representada, pois a dinâmica do paciente é incorporada na dinâmica da órtese.

Figura 2.16: Interface gráfica do simulador em Matlab (verde = perna direita; vermelho = perna esquerda; azul = tronco).

Figura 2.17: Animação do movimento com sobreposição.

As Figuras 2.18(a) a 2.18(d) apresentam os resultados de posição das juntas do tornozelo, joelho e quadril da perna direita e posição absoluta do tronco. Também são mostradas as trajetórias desejadas de cada junta. Entretanto, a posição do tronco, q_{torso} , sofre uma variação com relação à trajetória desejada no início do segundo passo, gerada pelas condições de impacto e variações no cálculo das forças de contato, Figura 2.18(d). Estas variações refletem diretamente

na posição do tronco, pois esta é medida de forma absoluta. Ao longo do passo a trajetória converge para a trajetória desejada através da aplicação do controle de posição. Pode-se notar também que a trajetória desejada é descontínua, pois seus parâmetros foram modificados, mas o controle rapidamente busca a nova trajetória de forma suave sem causar grandes perdas a essa.

Figura 2.18: Trajetórias absolutas, simuladas e desejadas do pé, tíbia, fêmur e torso.

2.4.1 Resultados da posição do ZMP utilizando o Matlab

Nesta seção são apresentados os resultados da posição do ZMP calculada utilizando o simulador. Nos gráficos referentes ao ZMP mostrados abaixo a origem do sistema de coordenadas é transferida para zero assim que um dos pés toca o solo e o outro o deixa.

Como podemos observar na Figura 2.19, a trajetória desejada do ZMP fica a maior parte do tempo dentro da região estável. Entretanto, a trajetória simulada, mesmo desconsiderando o

Figura 2.19: Posição do ZMP simulado.

pico negativo, gerado pela maneira com que as forças de contato são calculadas, fica um tempo maior fora da região estável e mais afastada do seu centro.

O controle proposto para melhorar a posição do ZMP é um controle PD, Eq. 2.18, aplicado na junta entre o tronco e a coxa da perna de suporte, e tendo como referência o centro da região estável. Este ângulo influencia diretamente a posição da perna em balanço e o tronco, modificando assim significativamente a posição do ZMP. Mas, como os sistemas da fase de apoio simples e de apoio duplo são diferentes, as constantes do controle, K_P e K_D , para cada fase são diferentes. Após o ajuste das constantes foram obtidos os resultados apresentados na Figura 2.20.

$$\tau_4 = K_P d_{zmp} + K_D \dot{d}_{zmp}. \quad (2.18)$$

O controle mostra eficácia ao realizar a aproximação do ZMP ao centro da região estável na fase de apoio duplo e faz com que após o pico negativo, o ZMP chegue mais rápido a região de estabilidade, na fase de apoio simples. Dessa forma esse controle se mostra eficaz em melhorar a estabilidade do sistema.

Figura 2.20: Posição do ZMP simulado com controle na junta do quadril.

2.5 Conclusão sobre o capítulo

Com base nos resultados obtidos das simulações pode-se afirmar que o método utilizado para geração de trajetória, descrito na Seção 2.2, mostra-se satisfatório para a aplicação no exoesqueleto em construção. Porém, tal método apresenta um grande inconveniente: o tempo necessário para a otimização pelo critério do ZMP, impedindo que o mesmo seja aplicado em tempo real. Uma maneira de diminuir esse tempo é com a utilização de redes neurais artificiais, para agilizar os cálculos necessários durante o processo de otimização do ZMP.

Importante ressaltar que as modificações propostas nas Seções 2.3.2 e 2.3.3 são apenas alternativas e podem ser inseridas em um processo de otimização que considere as restrições, ou seja, utilizando métodos de otimização com restrições. Optamos por contornar os problemas de restrições com as modificações propostas apenas para não sair do foco do trabalho, que são os algoritmos de adaptação e o uso de redes neurais.

O método de geração de trajetória utilizando o ZMP mostra boa serventia também para ser utilizado em algoritmos de adaptação do padrão de marcha, considerando os torques de interação entre o paciente e a órtese em todas as juntas. A utilização de algoritmos de adaptação do padrão de marcha é explorada nos próximos capítulos, considerando como parâmetros de adaptação os valores de T_c e D_s , pois ambos interferem diretamente na velocidade do caminhar, característica

bastante relevante para o caminhar. A utilização de redes neurais nesse processo também é explorada como alternativa para diminuir o tempo gasto durante os cálculos necessários no processo como um todo.

Capítulo 3

Algoritmos de adaptação do padrão de marcha

Os algoritmos utilizados nesse trabalho foram definidos em RIENER et al. (2005) e implementados também em JEZERNIK et al. (2004) na órtese Lokomat. O Lokomat é uma órtese com atuação nos joelhos e quadril, e emprega a técnica *treadmill training*, (HESSE et al. , 1995; DIETZ et al. , 1995), na qual o paciente está em pé sobre uma esteira e o peso do seu corpo é reduzido por um sistema de suspensão. Nesse caso, as trajetórias de referências dos joelhos e quadril são geradas a partir da trajetória de referência baseada no caminhar de uma pessoa sem deficiência, e um mecanismo do tipo quatro barras é usado para garantir a estabilidade lateral do sistema. Os algoritmos fazem a adaptação do padrão de marcha dos pacientes que caminham sobre a esteira. Esses algoritmos possibilitam ao paciente ter a capacidade de alterar o padrão de marcha conforme o seu grau de locomoção voluntária. JEZERNIK et al. (2004) desenvolveu três tipos de algoritmos, sendo o primeiro baseado na dinâmica inversa da órtese no qual é feita uma minimização *on-line* do torque de interação paciente-órtese, o segundo baseado na dinâmica direta no qual é feita uma estimação da variação da aceleração desejada do caminhar padrão, e o terceiro baseado em um controle de impedância e adaptação direta das trajetórias angulares das juntas.

Neste trabalho foram realizados testes e simulações utilizando os algoritmos baseados na dinâmica direta e inversa do modelo. Os algoritmos foram modificados de acordo com o modelo de exoesqueleto descrito no Capítulo 3, que tem por finalidade assistir pessoas em um ambiente livre para caminhar, ou seja, sem o uso de esteira e sem o uso de um suporte para o tronco do

paciente.

A seguir, é apresentada a modelagem dinâmica do exoesqueleto, e baseado nessa modelagem, posteriormente são mostrados os algoritmos de adaptação do padrão de marcha baseados na dinâmica direta e inversa do modelo.

3.1 Modelagem dinâmica do exoesqueleto

Para o desenvolvimento e a aplicação dos algoritmos de adaptação do padrão de marcha, a dinâmica do exoesqueleto foi modelada usando a equação básica da robótica, derivada pelo Princípio de Lagrange, (FU et al. , 1987):

$$\frac{d}{dt} \left(\frac{\delta L}{\delta \dot{q}_i} \right) - \frac{\delta L}{\delta q_i} = \tau_i, \quad \forall i = 1, 2, \dots, n, \quad (3.1)$$

onde L é a função lagrangiana ($L = T - U$), sendo T o total de energia cinética do robô, U o total de energia potencial do sistema, q_i o vetor de coordenadas generalizadas do robô, $\dot{q}_i$ o vetor de velocidade das juntas do robô, τ_i a força aplicada ao sistema para mover cada junta i , e n é o número de juntas do robô. A partir da Eq. 3.1, tem-se a equação dinâmica do exoesqueleto, dada por:

$$M_{ort}(q)\ddot{q} + C_{ort}(q, \dot{q}) + G_{ort}(q) = \tau_a + \tau_{pac} - \tau_{att}, \quad (3.2)$$

sendo $q \in \mathbb{R}^7$ o vetor das posições generalizadas compreendendo as juntas dos pés, joelhos, quadril e do tronco, $M \in \mathbb{R}^{7 \times 7}$ a matriz de inércia simétrica positiva definida, $C \in \mathbb{R}^7$ o vetor dos torques centrífugos e de Coriolis, e $G \in \mathbb{R}^7$ é o vetor dos torques gravitacionais. Os termos $\tau \in \mathbb{R}^7$ na Eq. 3.2 são os torques agindo na órtese: τ_a é o torque fornecido pelos atuadores, τ_{pac} é o torque gerado pela interação paciente-órtese e τ_{att} é o torque gerado pelo atrito nas juntas.

O torque de interação paciente-órtese, τ_{pac} , pode ser dividido em componentes ativo e passivo. O torque paciente-passivo, $\tau_{pac,pas}$, é o torque necessário para deslocar o paciente quando ele está se movendo de forma passiva, ou seja, não está exercendo nenhuma força que interfira na trajetória de referência. Caso o paciente influencie no movimento da órtese, ele produzirá o torque paciente-ativo, $\tau_{pac,at}$, e tenderá a mudar a trajetória de referência. Portanto, a Eq. 3.2 pode ser reescrita, considerando agora, as dinâmicas do paciente e da órtese:

$$M_{pac,ort}(q)\ddot{q} + C_{pac,ort}(q, \dot{q}) + G_{pac,ort}(q) = \tau_a + \tau_{pac,at} - \tau_{att}, \quad (3.3)$$

sendo $M_{pac,ort}(q)\ddot{q}$, $C_{pac,ort}(q, \dot{q})$ e $G_{pac,ort}(q)$ valores correspondentes a soma das dinâmicas do paciente e da órtese, ou seja, o termo $\tau_{pac,pas}$ é incorporado no sistema paciente-órtese.

O torque a ser aplicado pelos atuadores, τ_a , é obtido neste trabalho utilizando o controle torque calculado, (LEWIS et al. , 1993). Neste método, dada uma trajetória desejada, com suas acelerações, $\ddot{q}^d$, velocidades, $\dot{q}^d$, e posições, q^d , o torque necessário para o sistema se mover conforme esta trajetória é calculado pela equação dinâmica do sistema, ou seja:

$$\tau_a = M_{pac,ort}(q)(\ddot{q}^d) + C_{pac,ort}(q, \dot{q})\dot{q}^d + G_{pac,ort}(q). \quad (3.4)$$

Para assegurar que a trajetória real seguirá perfeitamente a trajetória desejada, foi acrescentado um controle PD. Assim, a Eq. 3.4 pode ser reescrita como:

$$\tau_a = M_{pac,ort}(q) \left(\ddot{q}^d + K_P e + K_D \dot{e} \right) + C_{pac,ort}(q, \dot{q})\dot{q}^d + G_{pac,ort}(q), \quad (3.5)$$

sendo e e $\dot{e}$ os erros de posição e velocidades das juntas, respectivamente, e $\ddot{e} = -K_P \dot{e} + K_D \ddot{e}$. Os escalares K_P e K_D podem ser escolhidos empiricamente.

Com essa modelagem, pode-se realizar uma abordagem utilizando a dinâmica direta ou a dinâmica inversa do modelo. O problema da dinâmica direta consiste em determinar a evolução temporal das posições e velocidades das juntas em função dos torques e forças aplicados pelos atuadores. O mais comum, entretanto, é o cálculo da dinâmica inversa, que consiste em determinar os torques nos atuadores, dadas as acelerações, velocidades e posições das juntas correspondentes. Em ambos os casos, para se calcular a dinâmica do modelo, faz-se necessário levar em consideração efeitos de inércia, forças centrífugas, atrito e viscosidade das juntas e a gravidade, mostrados na Eq. 3.3, tornando este um problema complexo de ser resolvido.

3.2 Algoritmo de adaptação baseado na dinâmica inversa

O algoritmo de adaptação baseado na dinâmica inversa é desenvolvido através da análise de interação paciente-órtese. A hipótese é que, no caso de um movimento sincronizado entre o paciente e a órtese, deverá existir uma mínima interação entre esses dois sistemas acoplados. No

caso ideal, a interação deve ser zero se o paciente seguir exatamente a trajetória de referência da órtese. Caso haja dessincronização, a interação tende a aumentar.

Esse algoritmo de adaptação é baseado na representação do problema da dinâmica inversa e trabalha via minimização dos torques de interação. No primeiro passo, os torques de cada junta são medidos e os torques de interação paciente-órtese são obtidos através de sensores de força instalados em pontos estratégicos da órtese. Para efeito de simulações, os torques de interação entre a órtese e o paciente são calculados a partir da comparação entre a trajetória desejada pelo paciente, q_{pac}^d , e a trajetória de referência, q_r . Assume-se que o torque de interação resulta de um acoplamento virtual do tipo mola, ou seja:

$$\tau_{pac,at} = K_v(q_{pac}^d - q_r). \quad (3.6)$$

O parâmetro K_v deve ser ajustado de modo a levar a magnitudes realistas do torque ativo do paciente. Aqui, o valor de K_v foi ajustado empiricamente, de acordo com os valores da trajetória desejada pelo paciente.

Depois de obter a estimação dos torques do paciente, a variação na trajetória de referência q_r é calculada de modo que esta variação resulte na redução do torque ativo produzido pelo paciente. Para que isso ocorra, é necessário realizar a otimização de um funcional, J_{inv} , que é calculado durante a trajetória:

$$J_{inv}(\delta q_r, F) = \sum_k \left\| -\tau_{pac,at}(F)_{(k)} - \delta\tau(\delta q_r)_{(k)} \right\|_2^2, \quad (3.7)$$

sendo que F representa as forças de interação entre o paciente e a órtese, e δq_r a variação na trajetória de referência. Assim, o problema é encontrar um valor de δq_r que minimize os torques de interação entre o paciente e a órtese.

A Eq. 3.7 mostra que a variação no torque $\delta\tau$ produzido por uma variação nas trajetórias de referência resulta na diminuição do torque ativo do paciente, $\tau_{pac,at}$. Essa variação de torque, $\delta\tau$, é calculada a partir da dinâmica inversa do modelo, ou seja:

$$\begin{aligned} \delta\tau(\delta q_r) = & M_{pac,ort}(q_r + \delta q_r) \left(\ddot{q}_r + \delta\ddot{q}_r \right) + C_{pac,ort} \left(q_r + \delta q_r, \dot{q}_r + \delta\dot{q}_r \right) \\ & + G_{pac,ort}(q_r + \delta q_r) - M_{pac,ort}(q_r) \ddot{q}_r - C_{pac,ort}(q_r, \dot{q}_r) - G_{pac,ort}(q_r). \end{aligned} \quad (3.8)$$

A Figura 3.1 ilustra o diagrama do sistema de adaptação baseado na dinâmica inversa.

Figura 3.1: Diagrama do sistema de adaptação baseado na dinâmica inversa.

Como a adaptação do padrão de marcha deve ocorrer em tempo real, um algoritmo de minimização rápido e simples deve ser usado. O método do gradiente descendente (o mesmo descrito na Seção 2.3.1), WIBERG (1976), que considera a primeira derivada, converge lentamente nas proximidades do mínimo, mas é poderoso para configurações distantes de um mínimo. Assim, o mesmo foi escolhido para calcular os parâmetros que minimizam o funcional J_{inv} .

A adaptação do padrão de marcha deve ser realizada sobre algum parâmetro do gerador de trajetórias. Considerando o gerador descrito no Capítulo 2, os parâmetros escolhidos para realizar a adaptação foram o tempo de duração de um passo, T_c , e o tamanho do passo, D_s , vistos que esses parâmetros interferem diretamente na velocidade do caminhar. Então, a adaptação utilizando esses parâmetros e considerando o método do gradiente descendente é feita pelas Eq. 3.9 e 3.10.

$$T_{c(k+1)} = T_{c(k)} - \eta \frac{\partial J_{inv}}{\partial T_c}, \quad (3.9)$$

$$D_{s(k+1)} = D_{s(k)} - \eta \frac{\partial J_{inv}}{\partial D_s}, \quad (3.10)$$

onde $\eta < 1$ é um escalar escolhido empiricamente, e $k = 1, \dots, n$ o número de iterações do processo de otimização.

3.3 Algoritmo de adaptação baseado na dinâmica direta

Para reduzir a dependência do modelo dinâmico no algoritmo de adaptação da Seção 3.2, uma variação do algoritmo foi desenvolvida a partir da dinâmica direta do sistema órtese-paciente:

$$\ddot{q} = M_{pac,ort}^{-1}(q) \{ \tau_a - \tau_{att} - C_{pac,ort}(q, \dot{q}) - G_{pac,ort}(q) \} + M_{pac,ort}^{-1}(q) \tau_{pac,at}. \quad (3.11)$$

Examinando a Eq. 3.11, pode-se observar que o último termo representa a variação na aceleração da trajetória de referência imposta pelo paciente. Essa hipótese forma a base do algoritmo de adaptação baseado na dinâmica direta. A ideia é estimar novamente o termo $\tau_{pac,at}$ via medição de forças e calcular a variação na aceleração. Adicionando a variação medida à aceleração nominal, obtém-se a aceleração desejada pelo paciente, ou seja:

$$\ddot{\delta}q = -M_{pac,ort}^{-1}(q) \delta\tau_{pac,at}, \quad (3.12)$$

$$\ddot{q}_{des,pac} = \ddot{q}_r + w\ddot{\delta}q, \quad (3.13)$$

sendo w uma constante de adaptação, $q_{des,pac}$ a trajetória desejada pelo paciente, e $\ddot{q}_r$ a aceleração considerando a trajetória de referência.

Assim como no método anterior, a adaptação dos parâmetros da trajetória é realizada via minimização de um funcional, nesse caso dado pela Eq. 3.14.

$$J_{dir}(\delta q_r, F) = \sum_k \left\| \ddot{q}_{des,pac}(\tau_{pac,at})_{(k)} - \ddot{q}_{adap}(\delta q_r)_{(k)} \right\|_2^2. \quad (3.14)$$

Uma importante vantagem da dinâmica direta em relação à inversa é a menor dependência do modelo, sendo que apenas o conhecimento da matriz de inércia $M_{pac,ort}(q)$ é necessário, ao contrário do método indireto que precisa do conhecimento das matrizes $M_{pac,ort}(q)$, $C_{pac,ort}(q, \dot{q})$ e $G_{pac,ort}(q)$.

A adaptação dos parâmetros é feita da mesma forma que o método anterior, ou seja, através dos cálculos dos novos valores de T_c e D_s , Eq. 3.9 e 3.10. O diagrama mostrado na Figura 3.1 também pode ser usado para representar o algoritmo de adaptação baseado na dinâmica direta, visto que a diferença está no cálculo do funcional J_{dir} e no uso da equação dinâmica do modelo.

3.4 Simulações dos algoritmos

Os algoritmos foram testados em simulações utilizando o Matlab, e considerando o gerador de trajetórias descrito no Capítulo 3. O tempo total de simulação foi fixado em 2,57 s, o que inclui dois processos de otimização dos parâmetros D_s e T_c , visto que a adaptação ocorre sempre no final de cada passo, ou seja, a primeira adaptação ocorre somente depois de realizado um passo. As simulações foram realizadas utilizando um computador pessoal com processador Core 2 Duo 2.0GHz 2MB Cache e 3GB de memória RAM, rodando o Sistema Operacional Microsoft Windows Vista, e os tempos mostrados como resultados dessas simulações foram obtidos baseados nesse sistema.

3.4.1 Resultados com algoritmo baseado na dinâmica inversa

O algoritmo baseado na dinâmica inversa foi implementado e simulado com valores iniciais de $D_s = 0,50$ e $T_c = 0,90$, o que corresponde a uma velocidade inicial de aproximadamente 1,1 m/s. Os valores desejados para esses parâmetros foram fixados em $D_s = 0,54$ e $T_c = 0,86$, correspondendo a um aumento de cerca de 12% na velocidade do caminhar. O processo de otimização dos parâmetros é realizado com cinco iterações do método do gradiente.

Os gráficos mostrados nas Figuras de 3.2 a 3.5 mostram as trajetórias angulares de cada junta do exoesqueleto, todas em valores absolutos, geradas após a simulação do sistema.

Figura 3.2: Trajetórias dos pés - dinâmica inversa.

(a) Trajetória da tíbia esquerda.

(b) Trajetória da tíbia direita.

Figura 3.3: Trajetórias das tíbias - dinâmica inversa.

(a) Trajetória do fêmur esquerdo.

(b) Trajetória do fêmur direito.

Figura 3.4: Trajetórias dos fêmures - dinâmica inversa.

Figura 3.5: Trajetórias do torso - dinâmica inversa.

Pode-se notar que já na primeira otimização dos parâmetros, os resultados já se mostram satisfatórios e a adaptação da trajetória é realizada para os valores adaptados. Os valores obtidos na primeira adaptação foram $D_s = 0,539$ e $T_c = 0,864$, e na segunda $D_s = 0,539$ e $T_c = 0,859$. Por ser dependente do modelo dinâmico no processo de otimização, o tempo de cálculo se mostrou elevado, sendo esse tempo de 42,8 segundos e 41,7 segundos, respectivamente para a primeira e segunda adaptação.

Para o torso, Figura 3.5, nota-se que as curvas da trajetória nominal, desejada e adaptada ficam sobrepostas já que não há alteração desses valores. No entanto, a trajetória atual sofre uma pequena variação devido a atuação do controle implementado, mas que não prejudica as trajetórias das outras juntas do exoesqueleto.

É importante observar que os tempos de cálculos mostrados aqui dependem muito das configurações do computador que está sendo usado para realizar as simulações. Neste trabalho, o mesmo computador foi utilizado para realizar as simulações.

3.4.2 Resultados com algoritmo baseado na dinâmica direta

O algoritmo baseado na dinâmica direta foi simulado com os mesmo valores iniciais e desejados de D_s e T_c utilizados no método anterior. Nesse caso, o tempo de cálculo dos processos de adaptação dos parâmetros foi de aproximadamente 17,1 segundos na primeira adaptação, e 20,9 segundos na segunda adaptação. Comparando com o método anterior, nota-se uma diminuição de cerca de 50% no tempo consumido para realizar esse processo.

Os gráficos mostrados nas Figuras de 3.6 a 3.9 mostram as trajetórias angulares de cada junta do exoesqueleto, todas em valores absolutos, geradas após a simulação do sistema baseado na dinâmica direta.

Figura 3.6: Trajetórias dos pés - dinâmica direta.

Figura 3.7: Trajetórias das tíbias - dinâmica direta.

Assim como no método da dinâmica inversa, os valores otimizados dos parâmetros D_s e T_c foram satisfatórios já na primeira adaptação. Os valores obtidos na primeira adaptação foram $D_s = 0,536$ e $T_c = 0,864$, e na segunda $D_s = 0,539$ e $T_c = 0,856$.

(a) Trajetória do fêmur esquerdo.

(b) Trajetória do fêmur esquerdo.

Figura 3.8: Trajetórias dos fêmures - dinâmica direta.

Figura 3.9: Trajetórias do torso - dinâmica direta.

3.5 Conclusão sobre o capítulo

Ambos os algoritmos mostraram resultados satisfatórios na adaptação do padrão de marcha. Por ser mais dependente do modelo, o algoritmo baseado na dinâmica inversa requer um tempo maior de cálculo no processo de otimização dos parâmetros. Os resultados mostraram que é viável a utilização dos dois algoritmos no sistema de adaptação para o exoesqueleto que está sendo desenvolvido em nosso laboratório.

Capítulo 4

Adaptação do padrão de marcha utilizando redes neurais

Muitos trabalhos têm feito uso de redes neurais artificiais para representar sistemas dinâmicos não-lineares, principalmente utilizando redes multicamadas e recorrentes. As redes multicamadas em geral apresentam grande robustez, sendo capazes de realizar adaptações em seus pesos sinápticos de acordo com as modificações do ambiente. Entre as redes com essa arquitetura, as mais utilizadas são as redes *Perceptron Multicamadas* (*MultiLayer Perceptron - MLP*). As redes MLP utilizam métodos simples de convergência durante a adaptação de seus pesos, geralmente baseadas no método do gradiente descendente, e são capazes de generalizar suas saídas com pequena margem de erro. Nesse capítulo é apresentada uma introdução teórica à arquitetura de redes MLP, apenas como forma de elucidar o funcionamento da rede no sistema, e posteriormente é apresentado o sistema de adaptação do padrão de marcha utilizando redes neurais em seu processo de otimização e geração de dados da trajetória.

4.1 Redes Perceptron de camada única

O Perceptron é a forma mais simples de uma rede neural artificial, sendo muito utilizado para classificação de padrões, desde que esses padrões sejam linearmente separáveis. Criado por Rosenblatt em 1958 (ROSENBLATT, 1958), o perceptron basicamente se constitui por um único neurônio, Figura 4.1, com pesos sinápticos, w_i , com i referente ao terminal de entrada da sinapse à qual o peso se refere, e bias ajustáveis (threshold), x_0 , que geralmente assumem valor $+1$ ou -1 . O bias tem o efeito de aumentar ou diminuir o potencial de ativação do neurônio

correspondente.

Figura 4.1: Modelo de um perceptron.

As entradas da rede, representadas pelo vetor $x(n) = [x_0(n), x_1(n), x_2(n), \dots, x_N(n)]^T$, com N = número de entradas, e n padrões de treinamento, são números reais enquanto a saída $y(n)$ é binária, sendo determinada por uma função de ativação do tipo sinal, $y = \{g(u) \mid g(u) = 1, \text{ se } u \geq 0, \text{ ou } g(u) = -1, \text{ se } u < 0\}$, ou $y = \{g(u) \mid g(u) = 1, \text{ se } u \geq 0, \text{ ou } g(u) = 0 \text{ se } u < 0\}$. A função de ativação restringe a amplitude de saída de um neurônio. O valor de u é obtido pela somatória dos produtos entre as entradas e seus respectivos pesos sinápticos, $u = \sum_{i=0}^N w_j x_i$. O treinamento do perceptron deve ser supervisionado, ou seja, deve-se saber a saída desejada para que a rede convirja, obtendo a classificação final da sua saída.

4.1.1 Análise matemática do perceptron

Partindo do princípio de que a saída do perceptron é fornecida por uma função sinal do tipo logística, e considerando um neurônio constituído de duas entradas, tem-se:

$$y = \begin{cases} 1, & \text{se } w_1 x_1 + w_2 x_2 - x_0 \geq 0 \\ -1, & \text{se } w_1 x_1 + w_2 x_2 - x_0 < 0. \end{cases} \quad (4.1)$$

O valor do bias pode ser multiplicado por um peso w_0 , ou seja $x_0 w_0$, agindo como uma entrada a mais. Se as duas classes as quais se referem os dados de saída da rede puderem ser separadas por uma única reta, então o perceptron pode ser aplicado. Como os valores iniciados para os pesos são aleatórios, a reta pode iniciar em qualquer posição, e o perceptron reposicionará

a reta de forma que a mesma fique na fronteira de separação dos grupos. O valor de x_0 na Eq. 4.1 define o coeficiente linear da reta (onde a reta corta o eixo y). Se for definido $x_0 = 0$, por exemplo, a reta passará pela origem. Os pesos w_i representam os coeficientes angulares das retas.

Graficamente, obtém-se:

Figura 4.2: Análise gráfica do Perceptron.

A adaptação dos pesos é representada na Eq. 4.2 na forma da *regra de aprendizagem por correção de erro*. Em particular, para um processo de minimização do erro, essa regra é comumente chamada de *regra delta de Widrow-Hoff*, pois foi proposta por WIDROW e HOFF (1960).

$$w_{ij}(n+1) = w_{ij}(n) + \eta[d(n) - y(n)]x_i(n), \quad (4.2)$$

onde η é a taxa de aprendizagem, sendo restrita ao intervalo $0 < \eta \leq 1$, e $d(n) - y(n)$ é o sinal de erro na saída da rede.

4.2 Redes Perceptron Multicamadas (MLP)

Como o uso de um único neurônio, ou mesmo uma única camada de neurônios perceptron é restrito a algumas aplicações de classificação de dois ou mais padrões linearmente separáveis, problemas mais complexos e que envolvam funções não lineares têm sido resolvidos com o uso de vários neurônios dispostos em mais de uma camada. Nessa arquitetura, incluem-se os

Perceptrons Multicamadas.

A teoria das redes MLP foi criada em 1969 por Minsky e Papert, e posteriormente aprimoradas por WERBOS (1974) com o desenvolvimento da teoria de aprendizagem do algoritmo *backpropagation*, ou *algoritmo de retropropagação do erro*. A tese de doutorado de Werbos, defendida em 1974, foi a primeira descrição documentada da computação eficiente do gradiente em modo reverso aplicada a modelos gerais de redes, sendo as redes neurais um caso particular. Tal algoritmo baseia-se na regra de aprendizagem por correção de erro e pode ser visto como uma generalização do algoritmo dos *Mínimos Quadrados (LMS)*.

A aprendizagem por retropropagação de erro consiste de dois passos através das diferentes camadas da rede: um passo para frente, a propagação forward, e um passo para trás, a retropropagação do erro backward. No passo forward, um padrão de entrada é apresentado e se propaga através da rede, camada por camada, obedecendo as funções somatórias e de ativação de cada neurônio, e um conjunto de saídas é produzido como resposta real da rede. Durante a propagação, os pesos sinápticos da rede não se alteram. Por outro lado, durante o passo backward, os pesos sinápticos são todos ajustados de acordo com uma regra de correção de erro. Desse modo, temos que a resposta real da rede é subtraída de uma resposta desejada para produzir um sinal de erro, que é então propagado para trás através da rede, contra a direção das conexões sinápticas. Os pesos sinápticos são ajustados para fazer com que a resposta real da rede se aproxime da resposta desejada. Cada processo como o descrito acima corresponde a uma época de treinamento da rede. Esse processo é repetido várias vezes até que o erro produzido na camada de saída atinja um valor esperado, ou que um número de épocas pré-determinado seja atingido.

Na Seção 4.2.1 é apresentada em detalhes a derivação do algoritmo backpropagation.

4.2.1 Algoritmo de treinamento Backpropagation

Para a derivação do algoritmo, considera-se uma rede com N entradas, duas camadas escondidas contendo $N1$ e $N2$ neurônios, respectivamente, e uma camada com $N3$ saídas, (HAYKIN, 2001), como visto na Figura 4.3. Assumindo essa arquitetura, temos as seguintes notações:

- $x(n) = [x_0(n), x_1(n), x_2(n), \dots, x_N(n)]^T$ é o vetor de entradas da rede, onde n é o número de padrões disponíveis para treinamento da rede;
- w_{lji} : é o peso sináptico conectando o j -ésimo neurônio da camada l ao i -ésimo neurônio da camada $(l - 1)$;

Figura 4.3: Modelo de rede perceptron multicamadas.

- I_{lj} : é a entrada ponderada do j -ésimo neurônio da camada l ;

$$I_{lj} = \left(\sum_{i=1}^{N_l} w_{lji} Y_{(l-1)j} \right), \quad (4.3)$$

onde $Y_{(l-1)j}$ é o sinal funcional de saída de cada neurônio. No caso Y_{0j} , o sinal funcional é dado pelos valores do vetor de entrada x .

Para o modelo mostrado na Figura 4.3, temos:

$$I_{lj} = \begin{cases} I_{1j} = \sum_{i=0}^N w_{1ji} Y_{0j} \\ I_{2j} = \sum_{i=0}^{N_1} w_{2ji} Y_{1j} \\ I_{3j} = \sum_{i=0}^{N_2} w_{3ji} Y_{2j}, \end{cases} \quad \text{e} \quad Y_{(l-1)j} = \begin{cases} Y_{0j} = x \\ Y_{1j} = g(I_{1j}) \\ Y_{2j} = g(I_{2j}) \\ Y_{3j} = g(I_{3j}). \end{cases} \quad (4.4)$$

A fase forward é realizada utilizando a Eq. 4.3 e então, um vetor de saída $y(n) = [y_1(n), y_2(n), \dots, y_{N_3}(n)]^T$ é produzido. Os padrões são enviados para a rede e ao final de cada época são calculados os erros locais e o erro global, considerando todas as saídas da rede. O *erro quadrático médio (EQM)* é uma função dos pesos da rede, e o objetivo geral é encontrar os pesos que minimizem o *EQM* e façam a rede aproximar da melhor maneira possível o mapeamento

entre todos os n dados de entrada, $x(n)$, para seus respectivos valores desejados, $d(n)$. A função relativa ao erro quadrático (local) é dada por $E(n) = \frac{1}{2} \sum_{j=1}^{N_3} (d_j(n) - y_j(n))^2$, enquanto que o erro quadrático médio é fornecido pela função $EQM = \frac{1}{n} \sum_{i=1}^n E(i)$.

Depois de realizada a fase forward, realizam-se os ajustes dos pesos de cada camada, partindo-se da última em direção a primeira camada. Os ajustes das matrizes de pesos podem ser divididos em dois casos como a seguir:

CASO 1: Neurônios da camada de saída

A idéia básica ao ajustar os pesos é encontrar os valores de w_{lji} que minimizam o erro global, EQM . Para isso, usa-se o erro derivado da camada mais próxima da saída, ou seja, o fator de correção do erro Δw_{lji} para o peso w_{lji} é proporcional à derivada parcial $\frac{\partial EQM}{\partial w_{lji}}$. Essa derivada por sua vez, representa um fator de sensibilidade, determinando a direção de busca no espaço de pesos.

Então, fazendo uso da regra da cadeia, tem-se:

$$\nabla EQM = \frac{\partial EQM}{\partial w_{3ji}} = \frac{\partial EQM}{\partial Y_{3j}} \frac{\partial Y_{3j}}{\partial I_{3j}} \frac{\partial I_{3j}}{\partial w_{3ji}}. \quad (4.5)$$

Pode ser verificado que

$$\frac{\partial I_{3j}}{\partial w_{3ji}} = Y_{2j}, \quad (4.6)$$

$$\frac{\partial Y_{3j}}{\partial I_{3j}} = g'(I_{3j}), \quad (4.7)$$

$$\frac{\partial EQM}{\partial Y_{3j}} = -(d_j - Y_{3j}). \quad (4.8)$$

A derivada de g em relação a I_3 somente é válida para função monótona crescente. Substituindo as Eq. 4.6, 4.7 e 4.8 em 4.5, temos:

$$\frac{\partial EQM}{\partial w_{3ji}} = -(d_j - Y_{3j})g'(I_{3j})Y_{2i}. \quad (4.9)$$

Logo, o ajuste de w_{3ji} deve ser realizado na direção oposta ao gradiente, ou seja:

$$\Delta w_{3ji} = -\eta \frac{\partial EQM}{\partial w_{3ji}} = \eta \delta_{3j} Y_{2i}^T. \quad (4.10)$$

Dessa forma, os pesos são atualizados de acordo com:

$$w_{3ji}(n+1) = w_{3ji}(n) + \eta \delta_{3j}(n) Y_{2i}^T(n), \quad (4.11)$$

onde

$$\delta_{3j} = (d_j - Y_{3j}) g'(I_{3j}). \quad (4.12)$$

CASO 2: Neurônios das camadas intermediárias (escondidas)

⇒ **Para a segunda camada intermediária:** Quando o neurônio j está localizado em uma camada oculta da rede, não existe uma resposta desejada específica para os neurônios. Desse modo, o sinal de erro deve ser determinado recursivamente, em termos dos sinais de erro de todos os neurônios aos quais o neurônio oculto está diretamente conectado. Então, utilizando novamente a regra da cadeia, e calculando o gradiente local, temos:

$$\nabla EQM = \frac{\partial EQM}{\partial w_{2ji}} = \frac{\partial EQM}{\partial Y_{2j}} \frac{\partial Y_{2j}}{\partial I_{2j}} \frac{\partial I_{2j}}{\partial w_{2ji}}. \quad (4.13)$$

Mas, pode ser observado que

$$\frac{\partial I_{2j}}{\partial w_{2ji}} = Y_{1i}, \quad (4.14)$$

$$\frac{\partial Y_{2j}}{\partial I_{2j}} = g'(I_{2j}), \quad (4.15)$$

$$\begin{aligned} \frac{\partial EQM}{\partial Y_{2j}} &= \sum_{k=1}^{N_3} \frac{\partial EQM}{\partial I_{3k}} \frac{\partial I_{3k}}{\partial Y_{2j}} = \sum_{k=1}^{N_3} \frac{\partial EQM}{\partial I_{3k}} \frac{\partial (\sum_{k=1}^{N_3} w_{3kj} Y_{2j})}{\partial Y_{2j}} \Rightarrow \\ &\Rightarrow \sum_{k=1}^{N_3} \frac{\partial EQM}{\partial I_{3kj}} w_{3kj} \Rightarrow \frac{\partial EQM}{\partial Y_{2j}} = - \sum_{k=1}^{N_3} \delta_{3k} w_{3kj}. \end{aligned} \quad (4.16)$$

Substituindo as Eq. 4.14, 4.15 e 4.16 em 4.13, obtém-se:

$$\frac{\partial EQM}{\partial w_{2ji}} = \left(\sum_{k=1}^{N_3} \delta_{3k} w_{3kj} \right) g'(I_{2j}) Y_{1i}. \quad (4.17)$$

Ajustando w_{2ji} na direção oposta ao gradiente, tem-se:

$$\Delta w_{2ji} = -\eta \frac{\partial EQM}{\partial w_{2ji}} = \eta \delta_{2j} Y_{1i}, \quad (4.18)$$

ou ainda:

$$w_{2ji}(n+1) = w_{2ji}(n) + \eta \delta_{2j}(n) Y_{1i}(n), \quad (4.19)$$

onde

$$\delta_{2j} = g'(I_{2j}) \sum_{k=1}^{N_3} (\delta_{3k} w_{3kj}). \quad (4.20)$$

$\Rightarrow$ **Para a primeira camada intermediária:** A adaptação dos pesos relativos à primeira camada escondida segue as mesmas regras do caso anterior. Calcula-se o gradiente local, sendo que os valores do vetor de entrada, x , influenciam diretamente no campo local induzido.

Nesse caso, temos:

$$\nabla EQM = \frac{\partial EQM}{\partial w_{1ji}} = \frac{\partial EQM}{\partial Y_{1j}} \frac{\partial Y_{1j}}{\partial I_{1j}} \frac{\partial I_{1j}}{\partial w_{1ji}}, \quad (4.21)$$

sendo,

$$\frac{\partial I_{1j}}{\partial w_{1ji}} = x_i, \quad (4.22)$$

$$\frac{\partial Y_{1j}}{\partial I_{1j}} = g'(I_{1j}), \quad (4.23)$$

$$\begin{aligned} \frac{\partial E}{\partial Y_{1j}} &= \sum_{k=1}^{N_2} \frac{\partial EQM}{\partial I_{2k}} \frac{\partial I_{2k}}{\partial Y_{1j}} = \sum_{k=1}^{N_2} \frac{\partial EQM}{\partial I_{2k}} \frac{\partial (\sum_{k=1}^{N_2} w_{2kj} Y_{1j})}{\partial Y_{1j}} \Rightarrow \\ &\Rightarrow \sum_{k=1}^{N_2} \frac{\partial E}{\partial I_{2kj}} w_{2kj} \Rightarrow \frac{\partial E}{\partial Y_{1j}} = - \sum_{k=1}^{N_2} \delta_{2k} w_{2kj}. \end{aligned} \quad (4.24)$$

Substituindo as Eq. 4.22, 4.23 e 4.24 em 4.21, obtém-se:

$$\frac{\partial EQM}{\partial w_{1ji}} = \left(- \sum_{k=1}^{N_2} \delta_{2k} w_{2kj} \right) g'(I_{1j}) x_i. \quad (4.25)$$

Ajustando w_{1ji} na direção oposta ao gradiente tem-se:

$$\Delta w_{1ji} = -\eta \frac{\partial EQM}{\partial w_{1ji}} = \eta \delta_{1j} x_i, \quad (4.26)$$

ou ainda:

$$w_{1ji}(n+1) = w_{1ji}(n) + \eta \delta_{1j}(n) x_i(n), \quad (4.27)$$

onde

$$\delta_{1j} = g'(I_{1j}) \sum_{k=1}^{N_2} (\delta_{2k} w_{2kj}). \quad (4.28)$$

Com essa estrutura, é possível generalizar as equações descritas nessa seção, e aplicá-las a qualquer rede MLP com algoritmo backpropagation.

4.2.2 Algoritmo de treinamento Levenberght-Marquardt

Enquanto o backpropagation é um algoritmo de descida (*steepest descent*), o algoritmo Levenberg-Marquardt é uma aproximação do método de Newton, com maior ordem de convergência do que o método do gradiente. Porém, o método de Levenberg-Marquardt possui a característica de não calcular a matriz Hessiana¹ como no método de Newton, o que o torna o processo de otimização mais rápido.

Suponha que temos uma função de erro $E(w)$ a qual queremos minimizar com relação ao parâmetro w , então o método de Newton nos dá:

$$\Delta w = -[\nabla^2 E(w)]^{-1} \nabla E(w), \quad (4.29)$$

onde $\nabla^2 E(w)$ é a matriz Hessiana e $\nabla E(w)$ é o gradiente de E em relação a w . Se assumirmos que $E(w)$ é uma função da soma dos quadrados dos erros locais, ou seja:

$$E(w) = \sum_{i=1}^N e_i^2(w). \quad (4.30)$$

Então, pode-se mostrar que:

¹A Matriz Hessiana de uma função de n variáveis é a matriz quadrada $n \times n$ das derivadas parciais de segunda ordem da função.

$$\nabla E(w) = J^T(w)e(w), \quad (4.31)$$

$$\nabla^2 E(w) = J^T(w)J(w) + S(w), \quad (4.32)$$

onde $J(w)$ é a matriz Jacobiana com relação a w , e

$$S(w) = \sum_{i=1}^N e_i(w) \nabla^2 e_i(w). \quad (4.33)$$

Para o método de Newton é assumido que $S(w) \approx 0$, e a atualização de w é dada por:

$$\Delta w = [J^T(w)J(w)]^{-1} J^T(w)e(w). \quad (4.34)$$

O método de Levenberg-Marquardt atualiza w da seguinte maneira:

$$\Delta w = [J^T(w)J(w) + \mu I]^{-1} J^T(w)e(w). \quad (4.35)$$

Dessa forma, os valores de w são atualizados de acordo com a Eq. 4.36.

$$w_{(k+1)} = w_{(k)} - (J^T(w)J(w) + \mu I)^{-1} J^T(w)e_{(k)}. \quad (4.36)$$

O fator μ interfere no método da seguinte maneira: quando μ é muito grande, o algoritmo se torna o mesmo que o gradiente descendente, enquanto para valores pequenos de μ o algoritmo se torna o método de Newton.

A contribuição principal deste algoritmo é o cálculo da matriz Jacobiana. Enquanto no método de Newton há a necessidade de se calcular a matriz Hessiana, tornando o processo de otimização mais demorado, no método de Levenberg-Marquardt, calcula-se apenas a matriz Jacobiana, tornando o processo mais rápido, e garantindo maior ordem de convergência, se comparado com outros métodos. Para o uso do algoritmo em uma rede MLP, os termos da matriz Jacobiana podem ser calculados com pequenas modificações no algoritmo backpropagation, (HAGAN e MENHAJ , 1994; ZHAO et al. , 2005).

4.2.3 MLPs como aproximadores universais de funções

Uma rede MLP treinada com o algoritmo backpropagation é capaz de implementar qualquer mapeamento não-linear que relacione as entradas e saídas, (HORNIK , 1991).

Teorema 4.2.1 (Teorema da Aproximação Universal) *Dado que $g(\cdot)$ é limitada, monótona crescente e contínua, então, qualquer função contínua arbitrária f pode ser implementada por uma MLP com uma única camada escondida de “ M ” neurônios, ou seja:*

$$F(x_1, x_2, \dots, x_n) = \sum_{i=1}^M \gamma_i g \left(\sum_{j=1}^N w_{ij} x_j - \theta_i \right), \quad (4.37)$$

cujo erro é dado por:

$$|F(x_1, x_2, \dots, x_n) - f(x_1, x_2, \dots, x_n)| \leq \epsilon. \quad (4.38)$$

De acordo com o Teorema 4.2.3, uma vasta gama de funções não lineares pode ser implementada através de uma MLP, sem que haja a necessidade de conhecer todos os pontos da função. Note que os neurônios da última camada em uma MLP trabalham como combinadores lineares. Se as funções de ativação são lineares, a convergência da rede é mais rápida.

4.2.4 Aspectos de treinamento de redes MLP

Além dos algoritmos de aprendizagem utilizados para treinar as redes MLP, outros fatores influenciam diretamente no desempenho das redes, entre eles, o número de neurônios nas camadas escondidas, o número de camadas escondidas, as funções de ativação e a inicialização dos pesos. Existem técnicas criadas para auxiliar na escolha desses parâmetros, mas nenhuma delas garante total certeza de convergência da rede. Algumas técnicas que podem ser utilizadas para a incrementar o desempenho do treinamento de redes MLP são citadas a seguir.

Validação Cruzada (*Cross Validation*)

A *validação cruzada* consiste em uma técnica estatística, que utiliza um subconjunto de dados derivados do conjunto de treinamento para testar o poder de generalização da rede. A metodologia adotada neste processo consiste em escolher aleatoriamente um subconjunto de

dados, entre 10% e 30% do total, e aplicar à rede já treinada para avaliar a sua capacidade de generalização. Se a rede estiver bem treinada, o erro obtido no processo de validação cruzada deverá ser mínimo.

O método pode ajudar a validar um modelo de rede já treinado, visto que, em certos casos, uma topologia pode ter desempenho satisfatório durante o treinamento, porém durante a sua utilização a rede produz resultados insatisfatórios, com *overfitting*, Figura 4.4.

Figura 4.4: Saída da rede sem *overfitting* e com *overfitting*.

A rede com *overfitting* foi treinada excessivamente, e com isso, apenas os dados de treinamento são reproduzidos pela rede com eficiência. Isso significa que a rede não generaliza de forma correta todos os dados do conjunto global e terá que ser treinada novamente, ajustando-se os parâmetros de treinamento para que a rede convirja de forma correta.

O processo de validação cruzada pode ser utilizado durante o treinamento, sempre depois de cada época, permitindo que o método seja utilizado como um critério de parada, ou depois do treinamento, como verificação.

Normalização dos dados de treinamento

Os dados de treinamento da rede devem ser normalizados dentro do intervalo $[0, 1]$, ou seja, $x_{norm} = ((x - x_{min}) / (x_{max} - x_{min}))$, quando se utiliza a função de ativação do tipo logística. Se a função de ativação for do tipo tangente hiperbólica, a normalização deve ser feita no intervalo

$[-1, 1]$, $x_{norm} = [2((x - x_{min})/(x_{max} - x_{min})) - 1]$. Se esse processo não for realizado, a rede pode convergir muito rapidamente sem estar correta.

Termo de Momentum (*Aceleração do Treinamento*)

A velocidade de convergência do algoritmo backpropagation pode ser aumentada por meio da inclusão do termo de *momentum*, α , como na Eq. 4.39.

$$\Delta w_{lji}(n+1) = \alpha \Delta w_{lji}(n) + \eta \delta_{lj}(n) Y_{(l-1)i}(n). \quad (4.39)$$

Assim, os pesos são atualizados de acordo com a Eq. 4.40.

$$w_{lji}(n+1) = w_{lji}(n) + \alpha(w_{lji}(n) - w_{lji}(n-1)) + \eta \delta_{lj}(n) Y_{(l-1)i}(n). \quad (4.40)$$

O termo de momentum atua da seguinte forma: quando o ponto em que estamos está longe do ponto de convergência, $\alpha \Delta w_{lji}(n)$ é quem imperará sobre o tamanho do passo, pois $\Delta w_{lji}(n)$ é grande. Conforme chegamos perto do ponto de convergência, $\Delta w_{lji}(n) \rightarrow 0$. Logo, $\alpha \Delta w_{lji}(n) \rightarrow 0$ e quem imperará no tamanho do passo será $\eta \delta_{lj} Y_{(l-1)i}$. O valor de α é escolhido empiricamente e depende da reação da rede durante o treinamento. Valores muito altos para α podem tornar a rede instável já nas primeiras épocas de treinamento. Por outro lado, uma escolha bem feita para α permite utilizar valores maiores para a taxa de aprendizado, o que aumenta a velocidade de convergência da rede.

4.3 Sistema de Adaptação do Padrão de Marcha com uso de Redes MLP

Com o objetivo de melhorar o desempenho do sistema de geração de trajetórias e adaptação da marcha, propostos nos Capítulos 2 e 3, é desenvolvido neste trabalho um sistema de adaptação utilizando redes MLP, com treinamento backpropagation, e dados de treinamento gerados a partir do modelo analítico. Foi considerado apenas o algoritmo de adaptação baseado na dinâmica inversa, descrito na Seção 3.2, visto que a maior parte dos problemas que envolvem a dinâmica de sistemas robóticos está em encontrar os torques necessários para atuação do sistema a partir de posições, velocidades e acelerações das juntas.

As redes foram implementadas primeiramente no processo de otimização e adaptação da

Figura 4.5: Diagrama do sistema com o uso de MLPs.

trajetória, redes NN1 e NN2 mostradas no diagrama da Figura 4.5, permitindo a inclusão do critério de estabilidade ZMP indiretamente durante a adaptação da marcha. O uso de uma MLP no processo de adaptação do padrão de marcha nada mais é do que uma espécie de automação do método do gradiente, mostrado na Seção 2.3.1. O algoritmo backpropagation, Seção 4.2.1, faz uso desse método e reproduz com grande precisão as saídas desejadas. Em seguida, o sistema de geração de trajetórias, mostrado no Capítulo 2 foi incorporado em uma rede MLP, NN3 na Figura 4.5, permitindo que as posições, velocidades e acelerações das juntas fossem calculadas com maior rapidez, sem perda de generalização dos pontos. Também nesse caso, o cálculo do ZMP foi inserido durante a geração dos dados de treinamento.

4.3.1 Rede MLP aplicada à adaptação do padrão de marcha

O sistema de adaptação proposto emprega duas redes MLPs, NN1 e NN2, mostradas no diagrama da Figura 4.5. A finalidade da implementação dessas duas redes é reduzir o tempo dispendido durante o processo de otimização e adaptação dos parâmetros da trajetória.

Rede NN1

A primeira rede neural, NN1, é treinada *offline* utilizando o modelo dinâmico paciente-órtese e a mudança na trajetória de referência, δq_r , para obter a variação no torque, $\delta\tau(\delta q_r)$, Eq. 3.8. O uso de uma rede neural para aproximar a variação $\delta\tau(\delta q_r)$ é justificada porque, neste caso, a otimização do ZMP não é realizada analiticamente. Este procedimento consome muito tempo,

mas com o uso de redes neurais, o processo de otimização do ZMP é realizado analiticamente somente durante a geração dos dados de treinamento da rede. O diagrama da Figura 4.6 mostra como foram gerados os dados para treinamento da rede NN1.

Figura 4.6: Geração analítica dos dados de treinamento da rede NN1.

Arquitetura da rede NN1: A rede NN1 é composta por três entradas relacionados ao valor atual de T_c , ao valor adaptado de $T_{c_{adap}}$ e ao instante de tempo em que ocorre o caminhar, t , acrescida de um bias com valor -1 . Desse modo, o vetor de valores de entrada da rede é denotado por $x(n) = [-1, T_{c_{atual}}(n), T_{c_{adap}}(n), t(n)]$, sendo $T_{c_{atual}}, T_{c_{adap}} \in [0, 80; 0, 90]$ s, e $t \in [0; 1, 07]$ s. A rede possui apenas uma camada escondida sendo composta por 12 neurônios com função de ativação sigmoidal do tipo logística, $g(u) = \left[\frac{1}{1 + \exp(-au)} \right]$, com $a = 1$. A camada de saída possui sete neurônios relativos a variação no torque de cada umas das juntas do robô, $y(n) = [y_1(n), \dots, y_7(n)] = [\delta\tau_1(n), \dots, \delta\tau_7(n)]$. A Figura 4.7 mostra a arquitetura da rede NN1.

Figura 4.7: Arquitetura da rede MLP NN1.

Os seguintes parâmetros foram considerados durante a fase de treinamento, com 5000 épocas: termo de momentum $\alpha = 0,89$; taxa de aprendizagem $\eta = 0,4$; $t \in [0 : 0,03 : 1,07]$ s; $T_{c_{atual}}, T_{c_{adap}} \in [0,80 : 0,01 : 0,90]$ s. É necessário incluir o valor atual do tempo no passo,

$T_{c_{atual}}$, visto que, em todo passo de otimização, $\delta\tau(\delta q_r)$ é calculado como uma variação da trajetória atual. Os valores de α e η foram obtidos empiricamente, de acordo com o comportamento da rede. Além disso, note que o valor de t é restrito ao intervalo $[0; 1,07]$ s, pois considera-se o intervalo de T_c um subconjunto do intervalo de t . Desse modo, mesmo que o tempo do passo seja maior que 0,9 s, mas menor que 1,07 s, o intervalo de t pode ser repetido para qualquer passo, iniciando-se em 0 s e terminado no tempo final do passo atual, determinado por T_c . Esse procedimento foi realizado para que a rede pudesse ser utilizada com qualquer quantidade de passos, sem que haja a necessidade de se treinar a rede com maior quantidade de dados, o que tornaria o trabalho altamente dispendioso em relação ao tempo de treinamento da rede. Com isso, o total de padrões gerados para treinamento chegou a 8.712 e todos os parâmetros de entrada da rede foram normalizados para valores entre 0 e 1, utilizando os valores máximos e mínimos de cada variável. Na fase de treinamento, os padrões foram inseridos aleatoriamente na rede, garantindo que a rede fosse realmente treinada, e não simplesmente armazenando padrões em sequência, pois esse processo tende a fazer com que o ajuste de pesos tenha um caráter estocástico ao longo do treinamento.

Os pesos da rede foram inicializados aleatoriamente, recebendo valores no intervalo $[-1; 1]$, e foram ajustados durante a fase de treinamento através do algoritmo backpropagation, com as equações descritas na Seção 4.2.1. Além disso, o termo de momentum, α , também foi acrescentado para melhorar a convergência da rede.

O erro de treinamento da rede NN1 teve o comportamento descrito na Figura 4.8:

Figura 4.8: Erro da rede NN1 treinada com 5.000 épocas.

Nota-se que com apenas 500 épocas o erro já se encontra estabilizado, com valores abaixo de 10^{-2} , mostrando que os valores escolhidos para α e η satisfazem a convergência da rede.

Para garantir que a convergência fosse satisfatória foi realizada a validação dos dados pelo processo de validação cruzada, com aproximadamente 10% dos dados de treinamento. O erro obtido nesse processo foi de 0,000524, mostrando que a rede generaliza com precisão os dados de treinamento. As Figuras 4.9 e 4.10 mostram a validação dos dados sendo aplicadas com os dados normalizados e não normalizados, respectivamente.

Figura 4.9: Validação cruzada para a rede NN1 com normalização das saídas.

Além da aplicação da validação dos dados, foi realizado um teste considerando as variáveis da trajetória para verificar se o mínimo do gradiente do funcional J_{inv} , descrito na Eq. 3.7, é coerente com o cálculo analítico. Desse modo, foi calculado o valor de ∇J_{inv} em relação ao parâmetro T_c desejado. Nas Figuras 4.11 e 4.12 podemos notar que a rede reproduz o mínimo de ∇J_{inv} como era esperado. Isso garante que a rede NN1 possa ser utilizada no processo de adaptação do padrão de marcha.

Figura 4.10: Validação cruzada para a rede NN1 sem normalização das saídas.

Figura 4.11: Mínimo de $\nabla J_{inv} - T_c$ desejado 0,82 s.

Figura 4.12: Mínimo de $\nabla J_{inv} - T_c$ desejado 0,87 s.

Rede NN2

A segunda rede neural, NN2, aplicada ao processo de adaptação do padrão de marcha tem como objetivo encontrar o valor de $T_{c_{adap}}$ que minimiza o funcional J_{inv} da Eq. 3.7. O treinamento dessa rede é realizado *online*, ou seja, não há um conjunto de dados pré-definidos para o treinamento. Para que a rede convirja para o valor desejado, é necessário que a entrada para a fase backward do backpropagation seja o gradiente de J_{inv} com relação a $T_{c_{adap}}$, $\nabla J_{inv} = \left(\frac{\partial J_{inv}}{\partial T_{c_{adap}}} \right)$, ou seja, ∇J_{inv} é o erro de saída da rede. Notemos que este valor é calculado utilizando a rede NN1, isto é, para cada $\delta\tau(\delta(q_r))$ é calculado o valor de J_{inv} e seu respectivo gradiente ∇J_{inv} . Com isso, a rede tende a seguir o valor do gradiente, convergindo para o valor ótimo de T_c que minimiza o funcional J_{inv} .

Arquitetura da rede NN2: Os valores de entrada para a rede NN2 são $T_{c_{atual}}$ e o atual valor do tempo no passo, t (o mesmo valor de t da rede NN1). A saída da rede é o valor adaptado de T_c , que gerará a nova trajetória, $T_{c_{adap}}$.

Além das duas entradas, a rede NN2 é composta por uma camada escondida contendo doze neurônios, e apenas um neurônio na camada de saída. Como na rede NN1, todos os neurônios possuem função de ativação do tipo logística e a convergência da rede ocorre entre cinco e dez épocas apenas, dependendo do ajuste dos parâmetros de treinamento. A Figura 4.13 mostra a arquitetura empregada na rede NN2:

Figura 4.13: Rede MLP NN2, treinada *online*.

Os valores da taxa de aprendizagem η , e do termo de momentum α , foram escolhidos empiricamente, e fixados em 10^{-3} e 10^{-6} , respectivamente. Os dados de entrada e saída da rede foram normalizadas com os valores máximos e mínimos de cada variável. Optou-se também por inicializar os pesos da rede com valores 0 a cada treinamento. Fazendo isso, observou-se que a rede encontra os pesos ideais para que a convergência ocorra mais rapidamente para o valor do gradiente de J_{inv} . Os pesos foram ajustados através do algoritmo backpropagation, Seção 4.2.1.

O erro da rede NN2 é mostrado na Figura 4.14, considerando o treinamento *online* com apenas 10 épocas e apenas um processo de otimização do parâmetro T_c . É possível observar que a convergência ocorre rapidamente já na primeira iteração, e posteriormente o erro decresce de maneira gradual chegando muito próximo do valor 0.

Figura 4.14: Erro da rede NN2 depois do primeiro passo.

O sistema de adaptação com as redes NN1 e NN2 foi implementado de duas maneiras: no final de cada passo e em instantes de tempo pré-determinados durante o passo. Os resultados são mostrados no Capítulo 5.

4.3.2 Rede MLP aplicada ao gerador de trajetórias

Rede NN3

A ideia inicial era aplicar o sistema de adaptação descrito na Seção 4.3.1, para que a trajetória da órtese fosse adaptada em cada instante de tempo. No entanto, a geração de trajetórias mostrou-se muito demorada, já que, para cada instante de tempo é necessário recalcular as posições, velocidade e acelerações de cada junta, além de realizar o processo de otimização do ZMP. Para minimizar esse problema e realizar a adaptação mais rapidamente durante o passo, optou-se por criar uma rede, para reproduzir o gerador de trajetórias, incorporando o modelo dinâmico da órtese e o critério de estabilidade ZMP.

As primeiras tentativas de reproduzir o gerador de trajetórias foram realizadas utilizando *redes de Funções de Base Radial* (RBF). Mas os resultados foram insatisfatórios, devido a não linearidade das funções que se pretendia reproduzir. Para conseguir a convergência da rede RBF foi necessário reduzir significativamente o raio das funções, o que interferiu diretamente na generalização da rede. Com isso, partiu-se para a utilização de redes MLP, como feito com as redes NN1 e NN2, pois sua convergência e poder de generalização mostraram-se satisfatório para esse caso.

Foi criada, então, uma rede MLP tendo como entradas o valor do parâmetro T_c e o instante de tempo t , e como saídas as posições, velocidades e acelerações, q , $\dot{q}$ e $\ddot{q}$, respectivamente, de cada uma das 7 juntas da órtese. Os dados de treinamento foram gerados utilizando o gerador de trajetórias estáveis, descrito no Capítulo 2. O diagrama da Figura 4.15 ilustra a obtenção dos dados para treinamento da rede NN3, utilizando como entrada o valor de T_c e do instante de tempo, t . Foram gerados 2.268 padrões de treinamento, considerando seus respectivos valores desejados de q , $\dot{q}$ e $\ddot{q}$.

Um dos problemas encontrados ao definir o número de neurônios e camadas da rede foi o fato de estar trabalhando com funções não lineares bastante complexas, sendo sua complexidade aumentada a cada derivação da função original, ou seja, a função de velocidade é mais complexa do que a de posição, e a de aceleração mais complexa do que a função de velocidade.

Figura 4.15: Geração analítica dos dados de treinamento da rede NN3.

Sendo assim, optou-se por dividir o problema em 3 subproblemas, criando uma rede MLP para representar cada uma dessas variáveis. Essa decisão foi tomada para que o erro de treinamento e generalização de uma rede, para cada uma das variáveis não influenciassem na geração dos dados de outra variável.

Tomando como base dados empíricos obtidos através de testes com a rede, variando-se o número de neurônios e camadas para cada treinamento, foi definido o número de neurônios nas camadas escondidas para cada uma das 3 variáveis a serem reproduzidas. Para o treinamento da rede com os dados de posição, foram utilizadas duas camadas escondidas contendo 12 neurônios cada. Para os dados de velocidade foram utilizadas três camadas escondidas contendo 10 neurônios cada, e por fim, para os dados de velocidade foram utilizadas também três camadas, mas estas com 15 neurônios cada. Com essa variação no número de neurônios e no número de camadas escondidas, pretendeu-se melhorar a convergência da rede de acordo com cada variável, compensando o aumento da não-linearidade em cada uma delas. Com isso, os dados podem ser obtidos como em sistemas independentes, não havendo influências entre eles. A Figura 4.16 mostra a rede NN3, incorporando as 3 sub-redes utilizadas para gerar os dados da trajetória da órtese.

Para a rede NN3, as funções de ativação utilizadas nas camadas escondidas foram do tipo tangente hiperbólica, $g(u) = \left[\frac{1 - \exp(-u)}{1 + \exp(-u)} \right]$, e para os neurônios de saída, a função de ativação linear, ou seja, $g(u) = u$. Como citado na Seção 4.2.3, o uso de uma função de ativação do tipo linear nos neurônios da última camada funciona como um combinador linear das funções representadas pela rede.

Alguns parâmetros de treinamento da rede NN3 não foram pré-determinados, visto que o mesmo foi realizado utilizando o Toolbox do Matlab.

As entradas para treinamento da NN3 foram o valor de T_c , o valor do tempo $t \in [0; 1,07]$ s (o mesmo das redes anteriores), e mais um bias com valor +1. Além disso, todas as entradas foram normalizadas utilizando os valores máximos e mínimos de cada variável.

Figura 4.16: Rede NN3 que substitui o gerador de trajetórias.

A rede foi treinada com 3.000 épocas e utilizando o algoritmo de treinamento Levenberg-Marquardt mostrado na Seção 4.2.2, devido a sua maior ordem de convergência. O algoritmo backpropagation também foi utilizado, mas sua convergência não obteve melhores resultados que o método anterior. Os erros de treinamento da rede para os dados de posição, velocidade e aceleração são mostrados nas Figuras 4.17(a), 4.17(b) e 4.17(c), respectivamente, sendo que os valores alcançados ao final deste processo foram $6,8 \times 10^{-5}$, $4,2 \times 10^{-3}$ e $8,4 \times 10^{-1}$, respectivamente.

Após o treinamento, o método de validação cruzada foi aplicado separadamente para as variáveis de posição, velocidade e aceleração, Figuras 4.18(a), 4.18(b) e Figura 4.18(c), respectivamente, considerando apenas uma saída apenas para uma melhor visualização dos resultados. Os dados para verificação da validação cruzada foram retirados a partir do conjunto de treinamento da rede, representando cerca de 11% do total de padrões. Os erros obtidos no processo de validação das redes foram de cerca de 0,002%, 0,05% e 1% do total de dados, respectivamente para as variáveis de posição, velocidade e aceleração. Pode-se notar que apesar de o erro ser consideravelmente maior para a velocidade e a aceleração, a rede mostra uma boa generalização dos dados, adaptando a função de acordo com os valores de treinamento.

(a) Erro da rede NN3 - Posição(q). (b) Erro da rede NN3 - Velocidade($\dot{q}$). (c) Erro da rede NN3 - Aceleração($\ddot{q}$).

Figura 4.17: Erro de treinamento da rede NN3.

(a) Validação cruzada - Posição(q). (b) Validação cruzada - Velocidade($\dot{q}$). (c) Validação cruzada - Aceleração($\ddot{q}$).

Figura 4.18: Validação Cruzada para os dados de treinamento da rede NN3.

4.4 Conclusão sobre o capítulo

As redes com arquitetura MLP mostram-se bastante eficientes ao reproduzir funções não lineares complexas. A vantagem de se utilizar redes MLP é que pode-se obter generalizações dessas funções a partir de amostras de dados obtidas do mapeamento do problema a ser resolvido.

A escolha do número de neurônios e das camadas escondidas muitas vezes é um inconveniente que pode ser resolvido testando-se valores durante a fase de treinamento da rede. Como esse processo consome muito tempo, a aplicação da Validação Cruzada depois do treinamento mostra-se uma boa alternativa para verificar se o número de neurônios e de camadas da rede foi satisfatório para reproduzir os dados desejados.

Capítulo 5

Resultados

As redes desenvolvidas e apresentadas no Capítulo 4 foram implementadas no simulador desenvolvido em Matlab, considerando os algoritmos de adaptação do padrão de marcha baseados na dinâmica inversa do modelo. Todas as simulações e resultados apresentados foram realizadas utilizando um computador pessoal com as seguintes características: processador Core 2 Duo 2.0GHz 2MB Cache e 3GB de memória RAM, rodando o Sistema Operacional Microsoft Windows Vista.

Mesmo utilizando uma máquina com configurações de alto desempenho, o tempo de treinamento das redes neurais é altamente dispendioso. Com isso, optou-se por realizar a adaptação de apenas um parâmetro da trajetória, sendo esse o tempo total de um passo, ou seja, o parâmetro T_c . A inclusão do parâmetro D_s pode ser facilmente implementada, pois o mesmo procedimento é realizado para a atualização dos dois parâmetros. No entanto, o tempo de treinamento das redes *offline* seria duas vezes maior, o que tornaria o trabalho mais demorado, além de ter que recalibrar todos os parâmetros de treinamento das redes. Ajustando-se o parâmetro T_c e adaptando a trajetória para seus novos valores, obtém-se variações significativas da velocidade do caminhar.

O simulador utilizado incorpora um modelo analítico da órtese, considerando a interação do paciente e forças de reação do solo, como descrito nos Capítulos 2 e 3. O torque de interação paciente-órtese é o mesmo mostrado na Seção 3.2, ou seja, gerado a partir de um acoplamento virtual do tipo mola, apenas para questões de simulação.

Nesta seção, serão apresentados os resultados simulados do sistema proposto, utilizando o algoritmo de adaptação do padrão de marcha baseado na dinâmica inversa.

5.1 Simulação do sistema

Para realizar as simulações aqui apresentadas, consideramos um padrão de marcha descrito por $D_s = 0,50$ e $T_c = 0,90$, e com os mesmos parâmetros mostrados na Tabela 2.2. A adaptação do parâmetro T_c foi realizada de duas formas diferentes: a primeira sendo realizada no final de cada passo, e a segunda com a otimização do parâmetro realizada em intervalos de tempo pré-determinados, iniciando-se a partir da metade do passo.

5.1.1 Caso I: Adaptação no final do passo

No primeiro caso, o valor de T_c é adaptado considerando 5 pontos do passo já decorrido, para o cálculo de J_{gait} . Como o valor de t é restrito ao intervalo $[0; 1,07]$, em segundos, devido ao treinamento das redes descritas no Capítulo 4, a cada novo passo simulado, a entrada t nas redes é recalculada para que sempre o valor do instante de tempo pertença ao domínio dos dados de treinamento da rede, ou seja, $t \in [0; T_c]$. Esse procedimento não interfere no restante do sistema, já que é feito apenas para definir o valor de t na entrada das redes, e o restante do simulador trabalha com o valor real de $t \in [0; t_{final}]$, sendo t_{final} o tempo total de simulação, em segundos.

A otimização do parâmetro T_c ocorre em apenas 5 iterações, sendo essas iterações o número de épocas para treinamento *online* da rede NN2. Assim, um novo passo para o modelo paciente-órtese é realizado considerando o parâmetro adaptado.

Primeira Simulação: Na primeira simulação realizada, o valor desejado foi fixado em $T_c = 0,82$ para todo o tempo de simulação, $t_{final} = 2,90$ segundos, o que permitiu uma simulação de 3 passos, conseqüentemente, 3 otimizações do parâmetro T_c . Ao final da simulação, o valor adaptado foi $T_{c_{adap}} = 0,819$. A sequência de valores de T_c obtidos nas 3 otimizações realizadas foram 0,822, 0,821 e 0,819 segundos, respectivamente para os passo 1, 2 e 3.

As Figuras de 5.1 a 5.4 apresentam as trajetórias nominal, desejada pelo paciente, adaptada, atual e a trajetória fornecida pela rede NN3 que substitui o gerador de trajetórias. Nessas figuras são consideradas as trajetórias das juntas referentes aos pés esquerdo e direito, tibia esquerda e direita, fêmur esquerdo e direito, e a trajetória angular do torso. Nota-se que depois do segundo passo, ou seja, depois de aproximadamente $t = 1,05$ segundos, a trajetória é adaptada para o valor desejado de T_c .

(a) Trajetória do pé esquerdo.

(b) Trajetória do pé direito.

Figura 5.1: Trajetórias dos pés - T_c desejado fixo.

(a) Trajetória da tíbia esquerda.

(b) Trajetória da tíbia direita.

Figura 5.2: Trajetórias das tíbias - T_c desejado fixo.

(a) Trajetória do fêmur esquerdo.

(b) Trajetória do fêmur esquerdo.

Figura 5.3: Trajetórias dos fêmures - T_c desejado fixo.

Figura 5.4: Trajetórias do torso - T_c desejado fixo.

Pode ser observado que o algoritmo obtém resultados satisfatórios com relação a adaptação dos parâmetros usados na trajetória desejada pelo paciente. A rede NN3 consegue generalizar suas saídas, fornecendo as trajetórias com eficiência. Nota-se também a variação da trajetória do torso no momento de impacto dos pés com o chão, devido ao controle torque calculado aplicado ao sistema. Essa variação, no entanto, não é suficientemente significativa a ponto de desestabilizar o sistema, o que pode ser observado nas outras trajetórias.

Segunda Simulação: A segunda simulação foi realizada considerando também a adaptação ao final de cada passo, mas com diferentes valores desejados de T_c , sendo $T_c = 0,87$, $T_c = 0,82$ e $T_c = 0,84$, respectivamente para o primeiro, segundo e terceiro passos. O tempo final de simulação foi fixado em $t_{final} = 2,90$ segundos.

Nessa simulação, os resultados para os valores adaptados de T_c foram muito precisos, sendo os valores obtidos após cada processo de adaptação, $T_c = 0,870$, $T_c = 0,821$ e $T_c = 0,841$. As Figuras de 5.5 a 5.8 mostram as trajetórias para cada junta que compõe o exoesqueleto.

Como no caso anterior, pode-se notar que o sistema realiza a adaptação da trajetória com eficiência, e do mesmo modo, a rede NN3 reproduz o gerador de trajetórias, seguindo a trajetória adaptada, em verde.

Além disso, um dos principais objetivos da utilização de redes neurais nesse sistema foi alcançado, ou seja, o tempo gasto durante cada processo de otimização diminuiu consideravelmente, ficando em torno de 4 segundos. Comparando com o processo analítico, sem o uso de nenhuma rede neural artificial, nota-se que há uma redução de cerca de 90% no tempo de cálculo do

(a) Trajetória do pé esquerdo.

(b) Trajetória do pé direito.

Figura 5.5: Trajetórias dos pés - T_c desejado variando em cada passo.

(a) Trajetória da tíbia esquerda.

(b) Trajetória da tíbia direita.

Figura 5.6: Trajetórias das tíbias - T_c desejado variando em cada passo.

(a) Trajetória do fêmur esquerdo.

(b) Trajetória do fêmur direito.

Figura 5.7: Trajetórias dos fêmures - T_c desejado variando em cada passo.

Figura 5.8: Trajetórias do torso - T_c desejado variando em cada passo.

processo de otimização dos parâmetros. Isso mostra a funcionalidade do uso de redes neurais no sistema.

O controlador do tipo torque calculado, citado na Seção 3.1, mostrou bons resultados, mantendo as trajetórias atuais do sistema sempre próximas dos valores desejados. Os parâmetros K_P e K_D do controlador, citado na Seção 3.1, foram definidos com valores de 1400 e 15, respectivamente. Uma vez que esses valores foram definidos, o controle se comportou de maneira eficiente em todas as simulações, sendo desnecessários novos ajustes.

5.1.2 Caso II: Adaptação durante o passo

No segundo caso, a adaptação ocorre depois da metade de cada passo, em intervalos definidos de 0,10 segundos. Nesse caso, são considerados cinco pontos do passo corrente para realizar a otimização do parâmetro T_c , ou seja, cinco pontos da trajetória já realizada considerando o passo atual.

Primeira Simulação: Na primeira simulação o valor de T_c desejado foi fixado em 0,82 para todos os passos. As Figuras de 5.9 a 5.12 mostram a adaptação da trajetória.

Pode-se observar que a adaptação ocorre simultaneamente nas trajetórias de todas as juntas do exoesqueleto, sendo que a primeira adaptação é realizada no momento em que a simulação encontra-se exatamente na metade do primeiro passo. Os resultados mostram que, caso o paciente mantenha a mesma trajetória, a adaptação continuará ocorrendo de forma satisfatória. Ocorrem algumas variações na trajetória gerada pela, mas são corrigida no decorrer do passo.

(a) Trajetória do pé esquerdo.

(b) Trajetória do pé direito.

Figura 5.9: Trajetórias dos pés - T_c desejado fixo e adaptação durante o passo.

(a) Trajetória da tíbia esquerda.

(b) Trajetória da tíbia direita.

Figura 5.10: Trajetórias das tíbias - T_c desejado fixo e adaptação durante o passo.

(a) Trajetória do fêmur esquerdo.

(b) Trajetória do fêmur direito.

Figura 5.11: Trajetórias dos fêmures - T_c desejado fixo e adaptação durante o passo.

Figura 5.12: Trajetórias do torso - T_c desejado fixo e adaptação durante o passo.

Segunda Simulação: Aqui são mostrados os resultados das simulações caso o paciente deseje modificar a trajetória mais de uma vez. A simulação foi realizada considerando uma adaptação no primeiro passo para $T_c = 0,82$, passando para $T_c = 0,86$ no segundo passo, e posteriormente para $T_c = 0,81$ no terceiro passo.

Figura 5.13: Trajetórias dos pés - T_c desejado variando e adaptação durante o passo.

Figura 5.14: Trajetórias das tíbias - T_c desejado variando e adaptação durante o passo.

Figura 5.15: Trajetórias dos fêmures - T_c desejado variando e adaptação durante o passo.

Figura 5.16: Trajetórias do torso - T_c desejado variando e adaptação durante o passo.

As Figuras de 5.13 a 5.16 mostram a adaptação sendo realizada para os valores desejados, sempre a partir da metade de cada passo, como definido anteriormente. Nesse caso, também é fácil observar que as redes reproduzem as trajetórias com erro mínimo, e o controlador mantém as trajetórias atuais próximas dos valores desejados, corrigindo as trajetórias no momento em que ocorrem as adaptações dos padrões de marcha.

Capítulo 6

Conclusões

O desenvolvimento de um sistema de adaptação do padrão de marcha proposto neste trabalho, para uso em um exoesqueleto para membros inferiores, é uma contribuição para o campo da engenharia de reabilitação, que tem ampliado seu campo de atuação, proporcionando melhora na qualidade de vida de pessoas com necessidades especiais.

O trabalho apresenta um algoritmo de adaptação do padrão de marcha utilizando redes neurais artificiais, considerando as forças de interação órtese-paciente e o critério de estabilidade ZMP, permitindo que o paciente modifique o padrão de marcha de acordo com o seu grau de locomoção voluntária. Três redes neurais com arquitetura MLP são utilizadas no sistema proposto, sendo que duas redes foram treinadas com o algoritmo backpropagation e uma rede utilizou o algoritmo de Levenberg-Marquardt em seu treinamento por mostrar resultados mais satisfatórios na reprodução dos dados da trajetória. A primeira rede aproxima a dinâmica inversa do modelo, incluindo o cálculo do ZMP, a segunda trabalha no procedimento de otimização dos parâmetros que serão adaptados, nesse caso, o tempo do passo T_c , e a terceira rede reproduz a geração de trajetórias estáveis do exoesqueleto.

Os resultados simulados mostram que o sistema proposto é eficiente para realizar a adaptação dos parâmetros que influenciam nas características do padrão de marcha. Vale notar que é possível o uso de outros valores e outros parâmetros de adaptação, de acordo com os valores desejados. Além disso, o custo computacional do sistema teve uma redução considerável, o que permite uma implementação quase que em tempo real no sistema físico. As ferramentas matemáticas utilizadas nesse trabalho podem ser melhoradas, ou mesmo, fazer uso de ferramentas mais robustas para que o resultado seja mais refinado, possibilitando também a ocorrência de

adaptações em tempo real.

Para trabalhos futuros, envolvendo a adaptação do padrão de marcha, pretende-se criar um gerador de trajetórias baseados em osciladores neurais, que são mais eficientes e menos dependentes do modelo dinâmico, o que pode melhorar a adaptação da trajetória durante o caminhar, podendo até mesmo ser realizada em tempo real. Também prevê-se o desenvolvimento e aplicação de novas arquiteturas de redes neurais para melhorar a convergência nos processo de a otimização dos parâmetros envolvidos. Além disso, deverão ser considerados outros aspectos do sistema, principalmente a dinâmica do usuário e a interação com o ambiente.

Referências Bibliográficas

- BANALA, S. K.; AGRAWAL, S. K. (2005). Gait Rehabilitation with an Active Leg Orthosis. In: *Proceedings of the IDETC/CIE 2005 ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. 2005, Califórnia, USA, p. 1–7.
- IBGE, I. B. D. G. E. E. (2003). *Censo Demográfico 2000 - Características Gerais da População*. Ministério do Planejamento, Orçamento e Gestão.
- DIETZ, V.; COLOMBO, G.; JENSEN, L.; BAUMGARTNER, L. (1995). Locomotor Capacity of Spinal Cord in Paraplegic Patients. *Annals of Neurology*, v.37, n.5, p.574–582, May.
- FERRIS, D. P.; SAWICKI, G. S.; DOMINGO, A. R. (2005). Powered Lower Limb Orthoses for Gait Rehabilitation. *Top Signal Cord Injuries Rehabilitation*, v.11, n.2, p.34–49.
- FU, K.; GONZALEZ, R. C.; LEE, C. S. G. (1987). *Robotics: Control, Sensing, Vision and Intelligence*. McGraw-Hill, Inc., New York, USA.
- FUKUDA, T.; KOMATA, Y.; ARAKAWA, T. (1997). Recurrent Neural Network with Self-adaptive GAs for Biped Locomotion Robot. In: *International Conference on Neural Networks*. 1997, volume 3, p. 1710–1715.
- GUIZZO, E.; GOLDSTEIN, H. (2005). The Rise of the Body Bots Robotic Exoskeletons. *IEEE Spectrum*, v.42, n.10, p.50–56, Oct.
- HAGAN, M. T.; MENHAJ, M. B. (1994). Training Feedforward Networks with the Marquardt Algorithm. *IEEE Transactions on Neural Networks*, v.5, n.6, p.989–993, Nov.
- HAYASHI, T.; KAWAMOTO, H.; SANKAI, Y. (2005). Control Method of Robot Suit HAL Working as Operator’s Muscle using Biological and Dynamical Information. In: *Proceedings*

of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2005, p. 3063–3068.

HAYKIN, S. (2001). *Redes Neurais: Princípios e Prática*. Bookman Companhia Editora, Porto Alegre, RS.

HESSE, S.; BERTELT, C.; JAHNKE, M. T.; SCHAFFRIN, A.; BAAKE, P.; MALEZIC, M.; MAURITZ, K. H. (1995). Treadmill Training with Partial Body Weight Support Compared with Physiotherapy in Nonambulatory Hemiparetic Patients. *Stroke*, v.26, p.976–981.

HORNIK, K. (1991). Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, v.4, n.2, p.251–257.

HUANG, Q.; YOKOI, K.; KAJITA, S.; KANEKO, K.; ARAI, H.; KOYACHI, N.; TANIE, K. (2001). Planning Walking Patterns for a Biped Robot. *IEEE Transactions on Robotics and Automation*, v.17, n.3, p.280–289, Jun.

JARDIM, B.; SIQUEIRA, A. A. G. (2008). Desenvolvimento de Atuadores Elásticos em Série para Acionamento de uma Órtese Tornozelo–Pé Ativa. *Anais do XVII Congresso Brasileiro de Automática*, p. 1–6.

JASPERS, P.; VAN PETEGEM, W.; VAN DER PERRE, G.; PEERAER, L. (1996). Design of an Automatic Step Intention Detection System for a Hybrid Gait Orthosis. In: *Proceedings of the 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 1996, volume 1, Amsterdam, Holanda, p. 457–458.

JEZERNIK, S.; COLOMBO, G.; KELLER, T.; FRUEH, H.; MORARI, M. (2003). Robotic Orthosis Lokomat: A Research and Rehabilitation Tool. *Neuromodulation*, v.6, n.2, p.108–115, April.

JEZERNIK, S.; COLOMBO, G.; MORARI, M. (2004). Automatic Gait-pattern Adaptation Algorithms for Rehabilitation with a 4-DOF Robotic Orthosis. *IEEE Transactions on Robotics and Automation*, v.20, n.3, p.574–582, June.

KAZEROONI, H. (1989). Human/robot Interaction via the Transfer of Power and Information Signals. In: *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 1989, volume 3, p. 908–909.

-
- KAZEROONI, H. (2005). Exoskeletons for Human Power Augmentation. In: *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005, Edmonton, Canada, p. 3459–3464.
- KIM, S.; ANWAR, G.; KAZEROONI, H. (2004). High-speed Communication Network for Controls with the Application on the Exoskeleton. In: *Proceedings of the 2004 American Control Conference*. 2004, volume 1, Boston, USA, p. 355–360.
- KIM, S.; SANKAI, Y. (2005). Stair Climbing Task of Humanoid Robot by Phase Composition and Phase Sequence. 2005, Nashville, TN, USA, p. 531–536.
- KIM, S.; SANKAI, Y. (2005). Task Generation for Humanoid Robot Walking using Human Motion by Phase Sequence. 2005, Nashville, TN, USA, p. 524–530.
- KITAMURA, S.; KUREMATSU, Y.; IWATA, M. (1990). Motion Generation of a Biped Locomotive Robot using an Inverted Pendulum Model and Neural Networks. In: *Proceedings of the 29th IEEE Conference on Decision and Control*. 1990, volume 6, p. 3308–3312.
- LEE, S.; SANKAI, Y. (2002). Power Assist Control for Walking Aid with HAL-3 Based on EMG and Impedance Adjustment around Knee Joint. In: *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2002, volume 2, Lausanne, Switzerland, p. 1499–1504.
- LEWIS, F. L.; ABDALLAH, C. T.; DAWSON, D. M. (1993). *Control of Robot Manipulators*. Macmillan Publishing Company, New York.
- MOUSAVI, P. N.; NATARAJ, C.; BAGHERI, A.; ENTEZARI, M. A. (2008). Mathematical Simulation of Combined Trajectory Paths of a Seven Link Biped Robot. *Applied Mathematical Modelling*, v.32, n.7, p.1445 – 1462.
- PALUSKA, D. J. (2000). Design of a Humanoid Biped for Walking Research. Thesis (Master), Massachusetts Institute of Technology, Cambridge, MA.
- PONS, J. L. (2008). *Wearable Robots: Biomechatronic Exoskeletons*. John Wiley & Sons, Ltd., Madrid, Spain.
- PRATT, G.; WILLIAMSON, M. (1995). Series Elastic Actuators. In: *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1995, volume 1, Pittsburgh, p. 399–406.

- PRATT, J.; DILWORTH, P.; PRATT, G. (1997). Virtual Model Control of a Biped Walking Robot. In: *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*. 1997, volume 1, Albuquerque, New Mexico, USA, p. 193–198.
- PÉREZ-ORIVE, J.; MAYAGOITIA, R. (1994). A Closed-loop Control System to be used a Hybrid RGO System. In: *Proceedings of the 16th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 1994, volume 1, p. 410–411.
- RIENER, R.; LÜNENBURGER, L.; JEZERNIK, S.; ANDERSCHITZ, M.; COLOMBO, G.; DIETZ, V. (2005). Patient-cooperative Strategies for Robot-aided Treadmill Training: First Experimental Results. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v.13, n.3, p.380–394, Sept.
- ROBINSON, D. W.; PRATT, J.; PALUSKA, D.; PRATT, G. (1999). Series Elastic Actuator Development for a Biomimetic Walking Robot. In: *Proceedings of the 1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. 1999, Atlanta, p. 561–568.
- ROSENBLATT, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, v.65, p.386–407. (Reprinted in *Neurocomputing* (MIT Press, 1988)).
- SIQUEIRA, A.; JARDIM, B.; VILELA, P.; WINTER, T. (2008). Analysis of Gait-pattern Adaptation Algorithms applied in an Exoskeleton for Lower Limbs. In: *Proceedings of the 16th Mediterranean Conference on Control and Automation*. 2008, p. 920–925.
- TO, C.; KIRSCH, R.; KOBETIC, R.; TRIOLO, R. (2005). Simulation of a Functional Neuromuscular Stimulation Powered Mechanical Gait Orthosis With Coordinated Joint Locking. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, v.13, n.2, p.227–235, June.
- VANDERBORGHT, B.; VERRELST, B.; VAN HAM, R.; VAN DAMME, M.; LEFEBER, D. (2008). Objective Locomotion Parameters based Inverted Pendulum Trajectory Generator. *Robotics and Autonomous Systems*, v.56, n.9, p.738–750.
- VUKOBRATOVIC, M.; JURICIC, D. (1969). Contribution to the Synthesis of Biped Gait. In: *Proceedings of the 1969 IEEE Transactions on Bio-Medical Engineering*. 1969, volume BME-16, p. 1–6.

- WERBOS, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Thesis (PhD), Harvard University, Cambridge, MA.
- WIBERG, T. (1976). Computation of Principal Components when Data are Missing. In: *Proceedings of the 2nd Symposium of Computational Statistics*. 1976, volume 107, Berlin, Germany, p. 229–326.
- WIDROW, B.; HOFF, M. E. (1960). Adaptive Switching Circuits. *IRE WESCON Convention Record*, v.4,p.96–104.
- ZHAO, J.; XIE, Z.; JIANG, L.; CAI, H. (2005). Levenberg-Marquardt Based Neural Network Control for a Five-fingered Prosthetic Hand. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. 2005, Barcelona, Spain, p. 4482–4487.
- ZOSS, A.; CHU, A.; KAZEROONI, H. (2006). Biomechanical Design of the Berkeley Lower Extremity Exoskeleton (BLEEX). *IEEE/ASME Transactions on Mechatronics*, v.11, n.2, p.128–138.
- ZOSS, A.; KAZEROONI, H.; CHU, A. (2005). On the Mechanical Design of the Berkeley Lower Extremity Exoskeleton (BLEEX). In: *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005, Edmonton, Canada, p. 3465–3472.

Apêndice A

Critério de estabilidade ZMP

O ZMP é definido como o ponto sobre o solo onde a somatória dos momentos de forças ativas é igual a zero. O ZMP pode ser calculado a partir das seguintes equações:

$$x_{zmp} = \frac{\sum_{i=1}^n m_i(\ddot{z}_i + g)x_i - \sum_{i=1}^n m_i\ddot{x}_i z_i - \sum_{i=1}^n I_{iy}\ddot{\Omega}_{iy}}{\sum_{i=1}^n m_i(\ddot{z}_i + g)}, \quad (A.1)$$

$$y_{zmp} = \frac{\sum_{i=1}^n m_i(\ddot{z}_i + g)y_i - \sum_{i=1}^n m_i\ddot{y}_i z_i - \sum_{i=1}^n I_{ix}\ddot{\Omega}_{ix}}{\sum_{i=1}^n m_i(\ddot{z}_i + g)}, \quad (A.2)$$

sendo m_i é a massa do link i , I_{ix} e I_{iy} são as componentes inerciais, $\ddot{\Omega}_{ix}$ e $\ddot{\Omega}_{iy}$ são as componentes das velocidades angulares ao longo do eixo x e do eixo y no centro de gravidade do link i , g é a aceleração da gravidade, $(x_{zmp}, y_{zmp}, 0)$ é a coordenada do ZMP, e (x_i, y_i, z_i) é a coordenada do centro de massa do link i sobre o sistema de coordenadas Cartesiano absoluto.

Figura A.1: Região estável e margem de estabilidade.

Se o ZMP está dentro do polígono convexo que determina a região estável, ou seja, o polígono

convexo formado pelos pontos de contato do robô com o solo, o caminhar do robô será estável. A distância mínima, d_{zmp} entre o ZMP e o contorno da região de estabilidade é chamada de margem de estabilidade.

Apêndice B

Interpolação por *splines* cúbicas

Para $n+1$ pontos $t_1 < t_2 < \dots < t_{n+1}$, de um espaço discreto, temos que $f_i(t)$, $i = 1, 2, \dots, n$, é um polinômio de grau 3 para cada par de pontos (t_i, t_{i+1}) , sendo a primeira derivada, $f'(t)$, e a segunda derivada, $f''(t)$, contínuas em (t_1, t_{n+1}) , e $f''(t)$ uma função linear de t . As funções f_i podem ser descritas por $f_i(t) = a_i + b_i t + c_i t^2 + d_i t^3$, ($i = 1, 2, \dots, n$), para $t_i \leq t \leq t_{i+1}$. Os 4 coeficientes de cada f_i devem ser determinados levando em consideração algumas restrições:

- Os valores das funções $f(t_i) = f_i$ ($i = 2, 3, \dots, n$) devem ser os mesmos para as duas *splines* em ambos os lados de t_i .
- A primeira e a segunda derivada das duas *splines*, em ambos os lados do ponto t_i , devem ser iguais.
- A primeira e a última *spline* devem passar pelos pontos t_1 e t_{n+1} , ou seja, $f_1(t_1) = f_1$ e $f_n(t_{n+1}) = f_{n+1}$, respectivamente.
- As curvaturas $f''(t)$ devem ser especificadas no primeiro e no último ponto, isto é, $f''_1(t_1) = f''_1$ e $f''_n(t_{n+1}) = f''_{n+1}$.

A partir dessas condições, é possível encontrar $4n$ equações lineares para as $4n$ coeficientes a_i, b_i, c_i , e d_i das *splines*. Esse conjunto de equações pode ser resolvido por algum método de resolução de sistemas de equações, como por exemplo, o método de eliminação de Gauss.

Fazendo uma abordagem a partir do polinômio de Lagrange de primeira ordem para a segunda derivada, pode-se determinar as funções f_i a partir de $f''_i(t)$.

Então, o polinômio de Lagrange de primeira ordem de $f_i''(t)$, no intervalo i , $t_i \leq t \leq t_{i+1}$, ($i = 1, 2, \dots, n$), é dado por:

$$f_i''(t) = \left(\frac{t - t_{i+1}}{t_i - t_{i+1}} \right) f_i'' + \left(\frac{t - t_i}{t_{i+1} - t_i} \right) f_{i+1}''. \quad (\text{B.1})$$

Integrando a Eq. B.1 duas vezes, temos as seguintes expressões:

$$f_i'(t) = \left(\frac{t^2/2 - tt_{i+1}}{t - t_{i+1}} \right) f_i'' + \left(\frac{x^2/2 - tt_i}{t_{i+1} - t_i} \right) f_{i+1}'' + C, \quad (\text{B.2})$$

$$f_i(t) = \left(\frac{t^3/6 - t^2 t_{i+1}/2}{t_i - t_{i+1}} \right) f_i''(t) + \left(\frac{t^3/6 - t^2 t_i/2}{t_{i+1} - t_i} \right) f_{i+1}'' + Ct + D. \quad (\text{B.3})$$

Avaliando a Eq. B.3 nos pontos t_i e t_{i+1} e combinando os resultados para eliminar as constantes de integração, temos:

$$\begin{aligned} f_i(t) = & \frac{f_i''}{6(t_{i+1} - t_i)} (t_{i+1} - t)^3 + \frac{f_{i+1}''}{6(t_{i+1} - t_i)} (t - t_i)^3 + \\ & \left[\frac{f_i}{t_{i+1} - t_i} - \frac{f_i''(t_{i+1} - t_i)}{6} \right] (t_{i+1} - t) + \\ & \left[\frac{f_{i+1}}{t_{i+1} - t_i} - \frac{f_{i+1}''(t_{i+1} - t_i)}{6} \right] (t - t_i). \end{aligned} \quad (\text{B.4})$$

A Eq. B.4 é a *spline* cúbica desejada para o incremento i , em termos dos valores desconhecidos f_i'' e f_{i+1}'' . Fazendo $f_{i-1}'(t_i) = f_i'(t_i)$, pode-se obter as derivadas de segunda ordem no interior do intervalo de interpolação, ou seja, para f_i'' ($i = 2, 3, \dots, n$).

Para encontrar uma expressão de $f_i'(t)$ pode-se diferenciar a Eq. B.3 e aplicá-la no intervalo $i - 1$ e i , avaliando em $t = t_i''$, dadas as expressões para $f_{i-1}'(t_i)$ e $f_i'(t_i)$. Desse modo, obtém-se:

$$\begin{aligned} (t_i - t_{i-1})f_{i-1}'' + 2(t_{i+1} - t_{i-1})f_i'' + (t_{i+1} - t_i)f_{i+1}'' = \\ 6 \left(\frac{f_{i+1} - f_i}{t_{i+1} - t_i} \right) - 6 \left(\frac{f_i - f_{i-1}}{t_i - t_{i-1}} \right). \end{aligned} \quad (\text{B.5})$$

Aplica-se a Eq. B.5 em $n - 1$ pontos do interior do intervalo, obtém-se $n - 1$ equações para $n + 1$ segundas derivadas, f_i'' ($i = 1, 2, \dots, (n + 1)$). Para finalizar, dois outros valores de f_i'' são necessários para fechar o sistema. Em geral, esses valores são obtidos fazendo $f_1'' = f_{n+1}'' = 0$.