Leonardo Shikata Augusto Tubota

Controladores Robustos Aplicados em Robôs Bípedes

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 Ciências, Programa de Engenharia Elétrica.

Área de Concentração:
Sistemas Dinâmicos

Orientador:
Prof. Dr. Marco Henrique Terra

São Carlos

2011

¹Trata-se da versão corrigida da dissertação. A versão original se encontra disponível na EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica.

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, 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

T885c Tubota, Leonardo Shikata Augusto.
 Controladores robustos aplicados em robôs bípedes. /
Leonardo Shikata Augusto Tubota; orientador Marco
Henrique Terra. São Carlos, 2011.

 Dissertação (Mestrado-Programa de Pós-Graduação em
Engenharia Elétrica e Área de Concentração em Sistemas
Dinâmicos) -- Escola de Engenharia de São Carlos da
Universidade de São Paulo, 2011.

 1. Robótica. 2. Controle (Teoria de Sistemas e
Controle). 3. Sistemas de controle. 4. Sistemas
autônomos. I. Título.

FOLHA DE JULGAMENTO

Candidato(a): Engenheiro **LEONARDO SHIKATA AUGUSTO TUBOTA**.

Dissertação defendida e julgada em 31/03/2011 perante a Comissão Julgadora:

Prof. Associado **MARCO HENRIQUE TERRA** – (Orientador)
(Escola de Engenharia de São Carlos/USP)

APROVADO

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

APROVADO

Prof. Dr. **FERNANDO SANTOS OSÓRIO**
(Instituto de Ciências Matemáticas e de Computação/USP)

APROVADO

Prof. Titular **DENIS VINÍCIUS COURY**
Coordenador do Programa de Pós-Graduação em
Engenharia Elétrica

Prof. Associado **PAULO CÉSAR LIMA SEGANTINE**
Presidente da Comissão da Pós-Graduação da EESC

*Dedico este trabalho aos meus pais
Cleide e Minoru que me deram todo
o suporte durante esta etapa de minha
vida. Também dedico a todos os meus
familiares pelo apoio.*

Agradecimentos

A Deus por tudo que me proporcionou neste período que estive no laboratório.

Ao Prof. Dr. Marco Henrique Terra pela oportunidade, confiança, orientação e paciência dispensadas ao longo deste trabalho.

Ao Prof. Dr. Adriano Almeida Gonçalves Siqueira pelo apoio e importantes contribuições para realização deste trabalho.

À instituição Universidade de São Paulo em São Carlos por toda a infraestrutura oferecida.

Aos professores e funcionários do Departamento de Engenharia Elétrica da USP São Carlos pela dedicação na realização de seus trabalhos.

Aos amigos do Laboratório de Sistemas Inteligentes (LASI) que tive a oportunidade de conviver tanto nos momentos de alegria como nos momentos difíceis: Aline Bianco, Amanda Manfrim, Daniel Vaz, Darby Albuquerque, Gildson de Jesus, Gustavo Brito, Igor Breda, João Paulo Cerri, José Miguel, Pedro Campos, Roberto Inoue, Samuel Lourenço, Tatiana Pazelli, Tatiane Francisco.

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

E, finalmente, à toda a minha família e aos demais amigos por demonstrarem todo o amor e apoio para a realização de um sonho.

Resumo

Este trabalho de mestrado propõe a aplicação de um controlador robusto recursivo, de um controlador robusto com critério H_∞ , obtido via Teoria dos Jogos e de um controlador H_∞ obtido através de LMIs, em um modelo dinâmico de robô bípede com joelhos e tronco. Considera-se um bípede planar de caminhar passivo, ou seja, possui um ciclo limite e sete graus de liberdade, estando sujeito a incertezas paramétricas e a distúrbios de torque externo. A principal vantagem do controlador robusto recursivo é a ausência de ajustes de parâmetros auxiliares, facilitando a sua implementação online. Já o controlador H_∞ obtido através da Teoria dos Jogos possui a vantagem de não precisar de cálculos offline, tais como solucionar Desigualdades Matriciais Lineares. Já o último controlador implementado possui a vantagem de ter seu ganho variando no tempo, assim como o Controlador Robusto Recursivo. Os resultados de simulação apresentados mostram a eficácia na implementação desses controladores em robôs bípedes, obtendo-se índices de desempenho que tendem a mostrar um melhor resultado do controlador H_∞ obtido através da Teoria dos Jogos.

Palavras-chave: Robótica, Controle (Teoria de Sistemas e Controle), Sistemas de Controle, Sistemas Autônomos.

Abstract

This work proposes the implementation of a Recursive Robust Control, a Robust Control with H_∞ criterion obtained from Game Theory and a Robust Control with H_∞ criteria obtained from the solutions of LMIs, in a dynamic model of a biped robot with knees and torso. We consider a planar biped robot with passive walk, i.e. it has limit-cycle, and has seven degrees of freedom. Also it is subject to parametric uncertainties and external disturbances. The main advantage of the Recursive Robust Controller is the absence of auxiliary parameter settings, facilitating its implementation online. Since the controller H_∞ obtained from the Game Theory has the advantage of not needing an offline calculation, such as solving Linear Matrix Inequalities. And the third one has the advantage of the variable gain in time, which occur with the Recursive Control too. The simulation results show the effectiveness of these controllers and the performance index show a tendency of better results applying the H_∞ obtained from Game Theory.

Keywords: Robotic, Control (Systems and Control Theory), Control Systems, Autonomous Systems.

Lista de Figuras

1.1	Robô humanóide Asimo.	p. 24
1.2	Robô bípede M2 da empresa Yobotics.	p. 25
1.3	Robô de caminhar passivo proposto por [15].	p. 26
2.1	Modelo do robô bípede planar sem pés.	p. 29
2.2	Modelo para a colisão com o solo.	p. 34
2.3	Retratos de fase do ciclo limite.	p. 37
3.1	Modelo do robô bípede planar sem pés em um plano inclinado.	p. 39
4.1	Distúrbios de torque aplicados nas juntas dos bípedes durante as simulações.	p. 55
4.2	Retratos de fase das juntas do robô durante 3 passos na presença de distúrbio utilizando o controlador robusto recursivo.	p. 57
4.3	Retratos de fase das juntas do robô na presença de distúrbios sob ação do controle H_∞ por Teoria dos Jogos.	p. 58
4.4	Retratos de fase das juntas do robô na presença de distúrbios sob ação do controle H_∞ por LMI.	p. 60
4.5	Retratos de fase das juntas do robô na presença de incertezas sob ação do controle Robusto Recursivo.	p. 61
4.6	Retratos de fase das juntas do robô na presença de incertezas sob ação do controle H_∞ via Teoria dos Jogos.	p. 62
4.7	Retratos de fase das juntas do robô na presença de incertezas sob ação do controle H_∞ via LMI.	p. 62
A.1	Formato da animação resultante da simulação do robô bípede.	p. 89

Lista de Tabelas

4.1	Parâmetros do robô bípede utilizados.	p. 53
4.2	Índices de Desempenho - Simulação	p. 60
4.3	Índices de Desempenho - Simulação	p. 63

Sumário

1	Introdução	p. 23
1.1	Motivação	p. 23
1.2	Modelagem e Estratégias de Controle	p. 23
	Caminhantes estáticos	p. 23
	Possuem atuadores em algumas juntas	p. 24
	Utilizam CPG	p. 25
	Caminhantes passivos	p. 26
	Abordagens híbridas	p. 27
1.3	Objetivos	p. 28
2	Modelo do Bípede	p. 29
2.1	Modelo do Robô Bípede sem Pés	p. 29
	2.1.1 Fase de Balanço	p. 30
	2.1.2 Fase de Colisão	p. 34
2.2	Geração do Ciclo Limite	p. 36
3	Controladores	p. 39
3.1	Compensador de Gravidade	p. 39
3.2	Controlador Robusto Recursivo	p. 41
	3.2.1 Regulador Linear Quadrático	p. 41
	3.2.2 Regulador Robusto	p. 42
3.3	Controle H_∞ via Teoria dos Jogos	p. 44

3.3.1	Dinâmica do Erro	p. 44
3.3.2	Controlador H_∞	p. 45
3.3.3	Simplificação da Equação de Riccati	p. 48
3.3.4	Cálculo de K e T_0	p. 49
3.4	Controle H_∞ via LMI	p. 49
4	Resultados	p. 53
4.1	Estratégia de Controle	p. 54
4.2	Simulação na presença de distúrbio	p. 55
4.2.1	Controlador Robusto Recursivo	p. 56
4.2.2	Controlador H_∞ via Teoria dos Jogos	p. 57
4.2.3	Controlador H_∞ via LMI	p. 58
4.3	Simulação na presença de distúrbios externos e incertezas paramétricas	p. 61
4.3.1	Controlador Robusto Recursivo	p. 61
4.3.2	Controlador H_∞ via Teoria dos Jogos	p. 61
4.3.3	Controlador H_∞ via LMI	p. 62
5	Conclusões	p. 65
5.1	Trabalhos Futuros	p. 65
	Referências Bibliográficas	p. 67
	Apêndice A – Simulador	p. 71
A.1	main.m	p. 71
A.2	walk1.m	p. 76
A.3	kneelock.m	p. 79
A.4	ctrj.m	p. 80
A.5	walk1b.m	p. 80

A.6	walk2.m	p. 85
A.7	interpolation.m	p. 88
A.8	anim.m	p. 89
A.9	Arquivos auxiliares	p. 94

1 Introdução

1.1 Motivação

O homem sempre tem buscado criar ferramentas para tornar mais fácil a realização de tarefas. Recentemente esse esforço está direcionado na criação de robôs capazes de fazê-las. Essa é uma das razões da existência de tantos tipos de máquinas robotizadas, cada uma especializada para realizar certo tipo de ação. Quando esta ação é a locomoção, tem-se que levar em conta o tipo de terreno, colocando os robôs com pernas, inclusive o bípede, em vantagem por sua habilidade de se adaptar a vários tipos de solo [1], possibilitando sua atuação em ambientes perigosos tais como reatores nucleares e químicos, exploração de terrenos inóspitos como o fundo do oceano e regiões polares, auxiliando em trabalhos pesados como em colheitas [2].

Devido a este fato, sistemas robóticos com duas pernas são amplamente estudados tanto em pesquisa básica como na indústria [3]. O maior desafio com relação ao controle desses robôs é a não linearidade dos impactos causados durante o travamento do joelho e durante o contato com o solo, o que gera mudanças bruscas nas velocidades das juntas.

1.2 Modelagem e Estratégias de Controle

Devido a sua popularidade nos meios acadêmicos e industriais, várias estratégias de controle e modelagem de robôs bípedes foram propostas. As principais abordagens: Caminhantes estáticos; Aqueles que possuem atuadores apenas em algumas juntas; Aqueles que usam um Gerador Central de Padrão (CPG em inglês) para a geração de trajetórias de referência; e Caminhantes passivos são apresentadas a seguir.

Caminhantes estáticos

Os caminhantes estáticos, cujo maior representante é o Asimo, mostrado na Figura 1.1, possuem baixa velocidade e a estabilidade é dada pela projeção normal do centro de massa do

robô, isto é, o caminhar é estabilizado se o centro de massa permanece dentro do polígono de suporte. Este polígono de suporte é calculado através da posição do "Zero Moment Point" (ZMP) cuja definição é dada por [4]:

"Considere um sistema de coordenadas Cartesianas cuja origem se encontra no ponto onde as forças resultantes de reação do solo estão agindo, com dois dos eixos (x e y) tangenciais ao solo, então a expressão matemática para a realização do balanceamento dinâmico é: $\sum M_x = 0$ e $\sum M_y = 0$. Incluindo todos os momentos, inclusive relativos a gravidade, forças inerciais e forças externas. Esse ponto onde $\sum M_x = 0$ e $\sum M_y = 0$ é denominado ZMP".

Figura 1.1: Robô humanoíde Asimo.

A maioria das estratégias de controle encontradas na literatura está baseada no controle de posição do ZMP [5]. No entanto, a existência e a estabilidade de órbitas periódicas (movimentos periódicos) em robôs bípedes não depende apenas da estabilidade do ZMP. Em [6], encontrou-se condições de estabilidade para os parâmetros de um bípede utilizando o controle de posição do ZMP afim de garantir a existência de órbitas periódicas.

Possuem atuadores em algumas juntas

Robôs que possuem atuadores somente em algumas juntas são utilizados como estratégia de forma a aproveitar as características naturais do caminhar, sendo seu principal representante

o robô M2 da empresa Yobotics, mostrado na Figura 1.2. Normalmente utilizam atuadores elásticos em série [7], que consiste em um atuador com um elemento elástico, geralmente uma mola, fazendo o acoplamento entre a transmissão (motor) e a carga. O funcionamento desse tipo de atuador consiste no movimento do motor comprimir um conjunto de molas, transmitindo a força para a carga. A modelagem é feita através de um sistema massa-mola com amortecimento [8]. A inserção desse elemento elástico aumenta a eficiência do atuador se comparado com atuadores tradicionais através da filtragem de impactos e distúrbios externos, além da redução do custo.

Figura 1.2: Robô bípede M2 da empresa Yobotics.

Utilizam CPG

Outra estratégia é a utilização de referências de trajetórias geradas por osciladores. A utilização de osciladores não lineares se deve ao fato de, segundo [9], a locomoção em seres vertebrados ser comandada por um circuito oscilatório localizado na medula espinhal e denominado de "*Central Pattern Generator*"(CPG). Em [10], os osciladores são implementados através de redes neurais. Neste caso para cada oscilador não linear é formado por dois neuro-osciladores, sendo que para cada neuro-oscilador existem dois neurônios. Estes neuro-osciladores são modelos para o CPG.

Em [11] e [12], osciladores não lineares são utilizados para a geração de trajetórias de referência, pois sua estratégia está baseada no fato de que o controle dos robôs bípedes serem feito em duas etapas: a) Sistema gerador de movimento e b) Sistema de controle de movimento.

Os osciladores são aplicados no Sistema gerador de movimento, enquanto que o Sistema de controle de movimento é feito através de um controlador Proporcional-Derivativo (PD) em cada junta. Em [13], osciladores não lineares são utilizados acoplados de forma hierárquica e unidirecional, tendo como base um oscilador mestre. O controle do robô bípede é feito de duas maneiras simultâneas: 1) Controle de alto nível, onde é feito o controle dos parâmetros dos osciladores, já que é possível através dos ajustes destes, escolher o tipo de movimento realizado (caminhar, correr, pular); e 2) Controle de baixo nível que consiste, como em [11] e [12], no controle das juntas para que a trajetória determinada pelos osciladores seja realizada com erro mínimo.

Caminhantes passivos

Finalmente, o movimento dos caminhantes passivos é criado a partir da ação da gravidade em um plano inclinado sem a necessidade de nenhuma atuação em suas juntas [14]. Neste tipo de robô, a cada passo, quando o pé da perna em balanço toca o solo, há gasto de energia que é regenerada através da inclinação do chão. O caminhar passivo ocorre quando as posições e velocidades das juntas no final do passo são idênticas às condições iniciais. O movimento gerado pode ser caracterizado como periódico ou, como é conhecido em sistemas não lineares, de ciclo limite. Um dos representantes dessa abordagem é proposto por [15] e a Figura 1.3 mostra o robô proposto.

Figura 1.3: Robô de caminhar passivo proposto por [15].

Em [16], é abordada a questão da otimização da energia consumida pelo sistema. A maioria dos trabalhos feitos nesta área abordam o conceito de restauração da energia perdida a cada impacto do joelho como sendo uma condição comum a todas as dinâmicas de geração de caminhada. A proposta do trabalho é fazer uma lei de controle usando como referência uma trajetória de energia, pois com isso a robustez do sistema é garantida sem a necessidade de introduzir trajetórias dependentes do tempo tais como posição angular e velocidade angular.

Abordagens híbridas

Outras técnicas de controle do caminhar são apresentadas na literatura, onde se encontra abordagens mistas de controle e modelagens utilizadas anteriormente.

Em [17], o conceito de “*swing-leg retraction*” (relação negativa entre o tamanho do passo e o tempo do passo) é aplicado no controle de um robô bípede em malha aberta. Este procedimento independe do estado do robô e portanto é constante em cada passo. Para garantir estabilidade neste caso são utilizados controladores PD em cada junta.

Utilizando os conceitos de *ZMP* e de osciladores não lineares, [18] faz o controle de caminhada, sendo o sistema modelado como um pêndulo invertido. Apesar da impossibilidade de representação da dinâmica do bípede através de uma simples dinâmica de fase, ainda sim é possível fazer a detecção desta através do *ZMP*. A geração das referências de trajetória é feita com o uso de funções periódicas simples tais como senóides, o que proporciona um número relativamente pequeno de parâmetros e consequentemente facilitando o projeto.

Um dos poucos trabalhos que utilizam controladores robustos para a estabilização do caminhar quando distúrbios externos são aplicados no robô, [19] faz uso de um controlador H_∞ não linear para o controle da trajetória de um robô bípede planar. O modelo utilizado nesse caso é de um robô com nove graus de liberdade com pés, joelhos e um tronco, além de apresentar um caminhar dinâmico passivo. Quando as juntas do robô são posicionadas próximas às condições iniciais ideais (bacia de atração), o robô consegue manter o caminhar sem a necessidade de nenhuma atuação de controle em suas juntas. Os resultados obtidos mostram um aumento da bacia de atração do ciclo limite. Em [20], também é utilizado o controle H_∞ , mas deduzido através da Teoria dos Jogos, enquanto em [19], o controle é calculado de maneira offline através de Desigualdades Matriciais Lineares (DMLs).

Outro trabalho utilizando controle robusto é visto em [21], sendo que neste, foi formulado um controlador robusto específico para um bípede em três dimensões utilizando como trajetória de referência o deslocamento do *ZMP* desejado para que o robô atinja um caminhar estável.

Em [22], a estabilidade de um bípede é estudada através do método de Poincaré, em um robô com 4 graus de liberdade (4 DOF). O centro de massa desse robô está localizado nos quadris (*hip*) através da colocação de contrapesos nas pernas, o que simplifica o modelo, baseado em um pêndulo invertido. Além disso foi desenvolvido um controlador robusto do tipo "*Sliding Mode*" para o acompanhamento das trajetórias das juntas do robô. Sendo [23] outro trabalho que utiliza esta técnica de controle.

1.3 Objetivos

O objetivo deste trabalho é a utilização do controlador robusto recursivo proposto em [24], do controle H_∞ utilizado em [20] e do controlador utilizado em [19], em um robô bípede passivo similar ao utilizado em [19], a menos da ausência de pés, e sujeito a distúrbios de torque e a incertezas paramétricas. A vantagem da utilização do primeiro controlador é que a recursividade pode ser realizada sem a necessidade do ajuste de parâmetros auxiliares, o que torna este procedimento de controle bastante útil para aplicações online. Já a vantagem do segundo controlador se encontra em sua formulação que evita a necessidade de utilização de DMLs para o cálculo dos ganhos do controlador, passo este necessário no terceiro controlador.

2 *Modelo do Bípede*

2.1 Modelo do Robô Bípede sem Pés

Nesta seção, é apresentado o modelo de um bípede planar com sete graus de liberdade (dois graus de liberdade do plano Y-Z, dois graus de liberdade de cada perna e um grau de liberdade do tronco), consistindo de pernas com joelhos e um tronco como mostra a Figura 2.1.

Figura 2.1: Modelo do robô bípede planar sem pés.

Os ângulos θ_1 da perna de suporte, θ_2 da coxa da perna em balanço, θ_3 da canela da perna em balanço e θ_T do tronco são definidos de forma absoluta com relação ao eixo vertical.

Para a dedução das equações, assume-se que a perna de suporte mantém-se esticada durante

o passo e, portanto, as massas da coxa e da canela desta perna podem ser combinadas em uma só. Além disso, considera-se que as massas estão concentradas nos centros de massa de cada elo do robô, de tal forma que os momentos de inércia dos elos não são incluídos na determinação do modelo dinâmico. O caminhar do robô é dividido em duas fases:

1. Fase de balanço, consistindo de todo o período em que o robô fica apoiado em uma única perna (maior parte do passo);
2. Fase de colisão da perna em balanço com o solo, abrange o período em que o robô fica apoiado sobre suas duas pernas, considerada instantânea.

2.1.1 Fase de Balanço

A fase de balanço é dividida em duas partes, antes e depois do impacto no joelho. Antes do impacto no joelho, a canela da perna em balanço é livre para mover-se em torno do joelho, sempre considerando que o movimento da canela se dá apenas para a parte posterior da perna em balanço, ou seja, $\theta_3 > \theta_2$ e após o impacto, a perna em balanço fica esticada devido à força de restrição no joelho.

Antes do impacto do joelho

Para a fase antes do impacto, as equações dinâmicas são determinadas pelas equações de Lagrange

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = B\tau \quad (2.1)$$

sendo

- B a matriz de entrada definida como

$$B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 \end{bmatrix}, \quad (2.2)$$

- $\theta = [\theta_1 \ \theta_2 \ \theta_3 \ \theta_T]^T$,

- $M(\theta) \in \mathbb{R}^{4 \times 4}$ a matriz de inércia (simétrica e positiva definida), definida como

$$M(\theta) = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & 0 \\ M_{31} & M_{32} & M_{33} & 0 \\ M_{41} & 0 & 0 & M_{44} \end{bmatrix}, \quad (2.3)$$

em que

$$\begin{aligned} M_{11} &= \frac{m_z^2 + l_1^2 m_1^2 - 2l_1 m_z m_1 + l_1^2 m_1 m_H + l_1^2 m_1 m_2 + l_1^2 m_1 m_3 + l_1^2 m_1 m_T}{m_1} \\ M_{12} = M_{21} &= -l_1 \cos(\theta_1 - \theta_2) (m_z + l_2 m_3) \\ M_{13} = M_{31} &= -l_1 m_z \cos(\theta_1 - \theta_3) \\ M_{14} = M_{41} &= l_1 m_z \cos(\theta_1 - \theta_T) \\ M_{21} &= -l_1 \cos(\theta_1 - \theta_2) (m_z + l_2 m_3) \\ M_{22} &= \frac{m_2 m_3 l_2^2 + m_z^2}{m_2} \\ M_{23} = M_{32} &= l_2 m_z \cos(\theta_2 - \theta_3) \\ M_{33} &= \frac{m_z^2}{m_3} \\ M_{44} &= \frac{m_z^2}{m_T} \end{aligned} \quad (2.4)$$

- $C(\theta, \dot{\theta}) \in \mathbb{R}^{4 \times 4}$ a matriz de Coriolis, definida como

$$C(\theta, \dot{\theta}) = \begin{bmatrix} 0 & C_{12} & C_{13} & C_{14} \\ C_{21} & 0 & C_{23} & 0 \\ C_{31} & C_{32} & 0 & 0 \\ C_{41} & 0 & 0 & 0 \end{bmatrix}, \quad (2.5)$$

em que

$$\begin{aligned}
 C_{12} &= -l_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) (m_{z2} + l_2 m_3) \\
 C_{13} &= -l_1 m_{z3} \dot{\theta}_3 \sin(\theta_1 - \theta_3) \\
 C_{14} &= l_1 m_{zT} \dot{\theta}_T \sin(\theta_1 - \theta_T) \\
 C_{21} &= l_1 \dot{\theta}_1 \sin(\theta_1 - \theta_2) (m_{z2} + l_2 m_3) \\
 C_{23} &= l_2 m_{z3} \dot{\theta}_3 \sin(\theta_2 - \theta_3) \\
 C_{31} &= l_1 m_{z3} \dot{\theta}_1 \sin(\theta_1 - \theta_3) \\
 C_{32} &= -l_2 m_{z3} \dot{\theta}_2 \sin(\theta_2 - \theta_3) \\
 C_{41} &= -l_1 m_{zT} \dot{\theta}_1 \sin(\theta_1 - \theta_T)
 \end{aligned} \tag{2.6}$$

- $G(\theta) \in \mathbb{R}^4$ o vetor dos torques gravitacionais, definido como

$$G(\theta) = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix}, \tag{2.7}$$

em que

$$\begin{aligned}
 g_1 &= -g \sin(\theta_1) (l_1 m_1 - m_{z1} + l_1 m_H + l_1 m_2 + l_1 m_3 + l_1 m_T) \\
 g_2 &= g \sin(\theta_2) (m_{z2} + l_2 m_3) \\
 g_3 &= m_{z3} g \sin(\theta_3) \\
 g_4 &= -m_{zT} g \sin(\theta_T)
 \end{aligned}$$

- $\tau = [\tau_1 \ \tau_2 \ \tau_3 \ \tau_4]^T$ o vetor de torques aplicados, no qual τ_1 é o torque no tornozelo da perna de suporte, τ_2 é o torque entre a perna de suporte e o tronco, τ_3 é o torque entre a perna em balanço e o tronco e τ_4 é o torque no joelho;
- $m_{z1} = m_1 c_1, m_{z2} = m_2 c_2, m_{z3} = m_3 c_3, m_{zT} = m_T c_T$;
- g é a aceleração da gravidade, adotada como sendo $g = 9,81 \text{ m/s}^2$;
- m_1 massa da perna de suporte, m_2 massa da coxa da perna em balanço, m_3 massa da canela em balanço, m_T massa do tronco e $m_H = m_2 + m_3 + m_T$;
- l_1 comprimento da perna de suporte, l_2 comprimento da coxa, l_3 comprimento da canela e l_T comprimento do tronco;

- c_1 centro de massa da perna de suporte, c_2 centro de massa da coxa, c_3 centro de massa da canela e c_T centro de massa do tronco.

Após o Impacto do Joelho

No instante em que ocorre o impacto no joelho, a coxa e a canela da perna em balanço possuem o mesmo ângulo $\theta_2 = \theta_3$. Assume-se que existe um mecanismo que trava o joelho e impede qualquer movimento relativo entre a coxa e a canela, mantendo a perna em balanço esticada. Os ângulos das juntas do bípede não se alteram durante o impacto, entretanto ocorre uma mudança brusca em suas velocidades. Esta mudança é modelada pela conservação do momento angular com relação à perna de suporte dada por

$$M(\theta^+) \dot{\theta}^+ = M(\theta^-) \dot{\theta}^- - J_i^T \lambda_i, \quad (2.8)$$

sendo $J_i^T \lambda_i$ a força impulsiva gerada pelo impacto, θ^+ e θ^- os valores de θ imediatamente antes e após o impacto no joelho e $J_i = [0 \ 1 \ -1 \ 0]$ a Jacobiana gerada pela restrição que $\theta_2 = \theta_3$ durante o impacto, ou seja,

$$J_i \dot{\theta}^+ = 0. \quad (2.9)$$

Notando que a inércia antes do impacto é igual à inércia depois do impacto ($\theta^+ = \theta^-$), obtém-se a partir de (2.8) as velocidades das juntas após o impacto como função das velocidades das juntas antes do impacto e da força gerada pelo impacto

$$\dot{\theta}^+ = \dot{\theta}^- - M(\theta)^{-1} J_i^T \lambda_i. \quad (2.10)$$

De (2.9) e (2.10), a seguinte relação é obtida:

$$J_i \dot{\theta}^+ = 0 = J_i \dot{\theta}^- - J_i M(\theta)^{-1} J_i^T \lambda_i.$$

Portanto, a força gerada pelo impacto é dada por

$$\lambda_i = X_i^{-1} J_i \dot{\theta}^- \quad (2.11)$$

sendo $X_i = J_i M(\theta)^{-1} J_i^T$.

Após o impacto no joelho, a perna em balanço mantém-se esticada devido à restrição na junta do joelho ($\theta_2 = \theta_3$). O modelo dinâmico com a força de restrição é dado por

$$M(\theta) \ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} + G(\theta) = B\tau - J_r^T \lambda_r, \quad (2.12)$$

sendo $J_r = [0 \ 1 \ -1 \ 0]$. A força de restrição é modelada da seguinte forma

$$\lambda_r = -X_r^{-1} J_r M(\theta)^{-1} (C(\theta, \dot{\theta}) \dot{\theta} + G(\theta)), \quad (2.13)$$

sendo $X_r = J_r M(\theta)^{-1} J_r^T$.

2.1.2 Fase de Colisão

A segunda fase do passo é caracterizada pelo impacto entre dois corpos rígidos de forma inelástica e considerada instantânea. Para obter as equações de colisão, modela-se o robô bípede considerando as pernas sem joelhos, conforme mostra a Figura 2.2, possuindo cinco graus de liberdade (dois graus de liberdade do plano Y-Z, um grau de liberdade para cada perna e um grau de liberdade do tronco). Assim, o modelo é considerado da seguinte forma

$$M_e(\theta_e) \ddot{\theta}_e + C_e(\theta_e, \dot{\theta}_e) \dot{\theta}_e + g_e(\theta_e) = B_e \tau_e + \delta F_{ext}, \quad (2.14)$$

sendo $\theta_e = [\theta_{e1} \ \theta_{e2} \ \theta_{eT} \ y \ z]^T$, com y e z sendo as coordenadas Cartesianas da perna de suporte. δF_{ext} são as forças externas agindo no robô nos pontos de contato.

Figura 2.2: Modelo para a colisão com o solo.

Assume-se que a colisão é instantânea e, portanto as forças externas podem ser represen-

tadas por impulsos, provocando uma mudança abrupta nas velocidades das juntas mantendo a continuidade nas posições. Além disso, assume-se que os torques fornecidos pelos atuadores não são impulsivos e podem ser desprezados durante o impacto. Sob estas condições e integrando (2.14) durante o impacto, tem-se

$$M_e(\theta_e)(\dot{\theta}_e^+ - \dot{\theta}_e^-) = F_{ext}, \quad (2.15)$$

sendo F_{ext} o resultado da integração das forças impulsivas, $\dot{\theta}_e^+$ são as velocidades imediatamente após a colisão e $\dot{\theta}_e^-$ são as velocidades imediatamente antes da colisão. Como as posições não se alteram durante a colisão, $\theta_e^+ = \theta_e^-$.

Assume-se também que após a colisão, a perna de suporte deixa o solo sem interação com este, portanto, as forças externas agindo no ponto de contato da perna de suporte são nulas. Assim, as forças externas F_{ext} devem conter apenas as forças atuando na extremidade da perna em balanço.

Seja Ψ o vetor das coordenadas Cartesianas da extremidade da perna em balanço:

$$\Psi(\theta_e) = \begin{bmatrix} y + l\sin(\theta_{e1}) - l\sin(\theta_{e2}) \\ z + l\cos(\theta_{e1}) - l\cos(\theta_{e2}) \end{bmatrix}. \quad (2.16)$$

Portanto,

$$F_{ext} = E^T \begin{bmatrix} F_T \\ F_N \end{bmatrix}, \quad (2.17)$$

sendo

$$E(\theta_e) = \frac{\partial \Psi(\theta_e)}{\partial \theta_e} = \begin{bmatrix} l\cos(\theta_{e1}) & -l\cos(\theta_{e2}) & 0 & 1 & 0 \\ l\sin(\theta_{e1}) & l\sin(\theta_{e2}) & 0 & 0 & 1 \end{bmatrix} \quad (2.18)$$

e F_T e F_N as forças tangencial e normal, respectivamente, aplicadas na extremidade da perna em balanço.

Assim, a equação (2.15) representa cinco equações com sete incógnitas. As incógnitas são $\dot{\theta}_e^+$ e F_{ext} . Um conjunto adicional de duas equações é obtido da condição que a extremidade da perna em balanço não se move após a colisão, $d/dt\Psi(\theta_e) = (\partial\Psi/\partial\theta_e)\dot{\theta}_e^+ = 0$, ou seja,

$$E(\theta_e)\dot{\theta}_e^+ = 0. \quad (2.19)$$

De (2.15) e (2.19), tem-se:

$$\begin{bmatrix} M_e(\theta_e) & -E(\theta_e)^T \\ E(\theta_e) & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_e^+ \\ F \end{bmatrix} = \begin{bmatrix} \dot{\theta}_e^- \\ 0 \end{bmatrix}, \quad (2.20)$$

com $F = [F_T \ F_N]^T$. O valor de $\dot{\theta}^+$ é obtido através do valor de $\dot{\theta}_e^+$ da forma

$$\dot{\theta}^+ = [\dot{\theta}_{e_1}^+ \ \dot{\theta}_{e_2}^+ \ \dot{\theta}_{e_2}^+ \ \dot{\theta}_{e_T}^+]^T. \quad (2.21)$$

Após o impacto, a perna em balanço passa a ser a perna de suporte e vice-versa. Portanto, é necessário realizar uma transformação de coordenadas para reiniciar a simulação do modelo (2.1). Considerando $x = [\theta \ \dot{\theta}]$, a transformação é dada por

$$x = \Delta(x), \quad (2.22)$$

sendo

$$\Delta(x) = [\theta_2 \ \theta_1 \ \theta_1 \ \theta_T \ \dot{\theta}_2 \ \dot{\theta}_1 \ \dot{\theta}_1 \ \dot{\theta}_T]^T. \quad (2.23)$$

2.2 Geração do Ciclo Limite

Em sistemas dinâmicos ciclo limite consiste em uma trajetória fechada no plano de fase, tendo a propriedade de possuir pelo menos uma trajetória que tende a este ciclo limite quando o tempo tende ao infinito. Um ciclo limite é uma forma de representação de movimentos oscilatórios não lineares. O robô bípede com tronco descrito neste capítulo não possui ciclo limite se a posição do tronco é livre, ou seja, não consegue completar um passo. Portanto, é necessário realizar o controle do tronco para alcançar o caminhar passivo. Neste caso, deseja-se que o tronco mantenha um ângulo com relação à vertical. Para isto, utiliza-se um controlador PD atuando na junta entre a perna de suporte e o tronco. Para cancelar o torque atuando na perna de suporte por causa deste controle, uma ação igual e de sentido contrário é realizada no calcanhar. O controlador PD é dado por:

$$\tau_2 = K_P(\theta_T^d - \theta_T) - K_D\dot{\theta}_T,$$

sendo K_P e K_D os ganhos proporcional e derivativo do controlador, respectivamente, e θ_T^d o ângulo desejado do tronco com a vertical. Neste caso os valores adotados foram $K_P = 110$, $K_D = 60$ e $\theta_T^d = 0$.

Através desse modelo e com o controle PD apresentado, o robô alcança o caminhar passivo para uma determinada inclinação do solo, sendo que as trajetórias das juntas formam um ciclo limite, ou seja, uma trajetória periódica não linear como mostra a Figura 2.3, onde os pontos em destaque são

A. Ponto de início do passo, onde uma perna está na frente da outra;

- B. Ponto de impacto do joelho, onde o ângulo da canela é igual ao ângulo da coxa e a perna em balanço fica esticada;
- C. Ponto de impacto com o solo, onde a perna de suporte está atrás da perna em balanço.

Figura 2.3: Retratos de fase do ciclo limite.

3 *Controladores*

Quando se trata de sistemas robóticos reais, os parâmetros do sistema dificilmente são conhecidos com exatidão (variação de carga, dos coeficientes de atrito, não linearidades desprezadas na modelagem), gerando interferências imprevisíveis no sistema. Dessa forma, torna-se necessário considerar os efeitos devido às incertezas paramétricas e perturbações externas. Existem basicamente duas estratégias de controle para sistemas incertos: controle adaptativo e controle robusto. Controladores adaptativos tentam estimar e compensar as dinâmicas não modeladas do sistema. Enquanto controladores robustos são concebidos por apresentarem bom desempenho mesmo na presença de incertezas paramétricas no modelo do sistema e de distúrbios externos [25].

3.1 Compensador de Gravidade

Figura 3.1: Modelo do robô bípede planar sem pés em um plano inclinado.

Considere o sistema bípede definido na Figura 3.1. Suponha que exista um conjunto de condições iniciais $x = x(\phi_0) = (\theta_1(0), \theta_2(0), \theta_3(0), \theta_T(0), \dot{\theta}_1(0), \dot{\theta}_2(0), \dot{\theta}_3(0), \dot{\theta}_T(0))$ tal que, com apenas o controle do tronco, um ciclo limite é obtido para a inclinação ϕ_0 . O controlador não-linear apresentado em [26] induz um ciclo limite para uma outra inclinação ϕ . Sejam $\beta = \phi_0 - \phi$, R_β a rotação definida pelo ângulo β e P a energia potencial. No espaço cartesiano, R_β é representado matricialmente por

$$R_\beta = \begin{bmatrix} \cos(\beta) & \sin(\beta) \\ -\sin(\beta) & \cos(\beta) \end{bmatrix},$$

enquanto que no espaço das juntas do robô bípede, R_β é representado por

$$R_\beta(\theta) = \begin{bmatrix} \theta_1 - \beta \\ \theta_2 - \beta \\ \theta_3 - \beta \\ \theta_T - \beta \end{bmatrix}.$$

Defina $P_\beta(\theta) = P(R_\beta(\theta))$ e

$$g_\beta = \frac{\partial P_\beta^T}{\partial \theta},$$

sendo P_β^T o transposto de P_β .

A lei de controle por realimentação é dada por:

$$\tau = B^{-1}(g(\theta) - g_\beta(\theta)).$$

Esta lei de controle induz um ciclo estável na inclinação ϕ com o conjunto de condições iniciais

$$x(\phi) = (\theta_1(0) + \beta, \theta_2(0) + \beta, \theta_3(0) + \beta, \theta_T(0) + \beta, \dot{\theta}_1(0), \dot{\theta}_2(0), \dot{\theta}_3(0), \dot{\theta}_T(0)),$$

que agora pertence à bacia de atração deste ciclo limite. Há o cancelamento do torque gravitacional atuando no robô e a criação de um vetor de gravidade virtual que corresponde à inclinação ϕ_0 . Com isso o vetor de torques gravitacionais em 2.1 se torna

$$g_\phi = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} \quad (3.1)$$

em que

$$\begin{aligned} g_1 &= -g \sin(\theta_1 - \phi) (l_1 m_1 - m_{z1} + l_1 m_H + l_1 m_2 + l_1 m_3 + l_1 m_T) \\ g_2 &= g \sin(\theta_2 - \phi) (m_{z2} + l_2 m_3) \\ g_3 &= m_{z3} g \sin(\theta_3 - \phi) \\ g_4 &= -m_{zT} g \sin(\theta_T - \phi). \end{aligned}$$

Este procedimento é utilizado neste trabalho para gerar um vetor de gravidade virtual correspondente a uma inclinação ϕ_0 , estando o robô se movendo sobre uma superfície com inclinação nula ($\phi = 0$). Assim, o robô caminha numa superfície com inclinação nula, mas com forças gravitacionais relativas a uma inclinação ϕ_0 .

3.2 Controlador Robusto Recursivo

Nesta seção é apresentado um regulador robusto recursivo para sistemas lineares de tempo discreto sujeitos a incertezas paramétricas. O cálculo é baseado na extensão de funcional quadrático utilizado para a dedução do regulador nominal recursivo. O funcional obtido a partir desta extensão é a combinação de uma função penalidade e de um funcional com custo ponderado definido na teoria dos jogos [27].

Assume-se então que o sistema discreto no tempo é dado pelas seguintes equações

$$x_{i+1} = (F_i + \delta F_i)x_i + (G_i + \delta G_i)u_i; \quad i = 0, \dots, N, \quad (3.2)$$

sendo $F_i \in \mathbb{R}^{n \times n}$ e $G_i \in \mathbb{R}^{n \times m}$ as matrizes de parâmetros nominais, $x_i \in \mathbb{R}^n$ o vetor de estado, $u_i \in \mathbb{R}^m$ a entrada de controle e o estado inicial x_0 assumido conhecido. Sejam $\delta F_i \in \mathbb{R}^{n \times n}$ e $\delta G_i \in \mathbb{R}^{n \times m}$ as matrizes de incertezas desconhecidas modeladas como

$$\begin{bmatrix} \delta F_i & \delta G_i \end{bmatrix} = H_i \Delta_i \begin{bmatrix} E_{F_i} & E_{G_i} \end{bmatrix}; \quad i = 0, \dots, N, \quad (3.3)$$

sendo $H_i \in \mathbb{R}^{n \times k}$, $E_{F_i} \in \mathbb{R}^{l \times n}$, $E_{G_i} \in \mathbb{R}^{l \times m}$ matrizes conhecidas e $\Delta_i \in \mathbb{R}^{k \times l}$ uma matriz arbitrária com $\|\Delta_i\| \leq 1$.

3.2.1 Regulador Linear Quadrático

Considere o seguinte funcional quadrático

$$J = x_{N+1}^T P_{N+1} x_{N+1} + \sum_{j=0}^N \mathcal{L}_j(x_j, u_j)$$

e o seguinte modelo linear no espaço de estado sem incertezas

$$x_{i+1} = F_i x_i + G_i u_i; \quad i = 0, \dots, N \quad \text{com } x_0 = cte,$$

sendo

$$\mathcal{L}_j(x_j, u_j) = x_j^T Q_j x_j + u_j^T R_j u_j; \quad j = 0, \dots, N.$$

Define-se o seguinte problema de minimização com restrição

$$\begin{aligned} \min_{x_{i+1}, u_i} \{ & x_{N+1}^T P_{N+1} x_{N+1} + \sum_{j=0}^N \mathcal{L}_j(x_j, u_j) \} \\ \text{s.a } & x_{i+1} = F_i x_i + G_i u_i; \quad i = 0, \dots, N \end{aligned}$$

A solução recursiva ótima para tal problema é dada por

$$\begin{bmatrix} x_{i+1}^* \\ u_i^* \end{bmatrix} = \begin{bmatrix} L_i \\ K_i \end{bmatrix} x_i; \quad i = 0, \dots, N, \quad (3.4)$$

sendo L_i e K_i obtidos de acordo com a seguinte recursão

$$\begin{bmatrix} L_i \\ K_i \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} P_{i+1}^{-1} & 0 & 0 & 0 & I & 0 \\ 0 & R_i^{-1} & 0 & 0 & 0 & I \\ 0 & 0 & Q_i^{-1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & -G_i \\ I & 0 & 0 & I & 0 & 0 \\ 0 & I & 0 & -G_i^T & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ -I \\ F_i \\ 0 \\ 0 \end{bmatrix}, \quad (3.5)$$

$$P_i = L_i^T P_{i+1} L_i + K_i^T R_i K_i + Q_i; \quad i = N, \dots, 0. \quad (3.6)$$

3.2.2 Regulador Robusto

Com base no que foi apresentado na subseção anterior, o controlador considerado é obtido a partir da solução ótima $(x_{i+1}^*(\mu), u_i^*(\mu))$ do seguinte problema de otimização

$$\min_{x_{i+1}, u_i} \max_{\delta F_i, \delta G_i} \{ J_i^\mu(x_{i+1}, u_i, \delta F_i, \delta G_i) \} \quad (3.7)$$

$$\begin{bmatrix} \delta F_i & \delta G_i \end{bmatrix} = H_i \Delta_i \begin{bmatrix} E_{F_i} & E_{G_i} \end{bmatrix},$$

sendo $J_i^\mu(x_{i+1}, u_i, \delta F_i, \delta G_i)$ o funcional custo quadrático dado pela seguinte expressão

$$J_i^\mu(x_{i+1}, u_i, \delta F_i, \delta G_i) = \begin{bmatrix} x_{i+1} \\ u_i \end{bmatrix}^T \begin{bmatrix} P_{i+1} & 0 \\ 0 & R_i \end{bmatrix} \begin{bmatrix} x_{i+1} \\ u_i \end{bmatrix} + \left\{ \left(\begin{bmatrix} 0 & 0 \\ I & -G_i \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -\delta G_i \end{bmatrix} \right) \begin{bmatrix} x_{i+1} \\ u_i \end{bmatrix} - \left(\begin{bmatrix} -I \\ F_i \end{bmatrix} x_i + \begin{bmatrix} 0 \\ \delta F_i \end{bmatrix} x_i \right) \right\}^T \begin{bmatrix} Q_i & 0 \\ 0 & \mu I \end{bmatrix} \left\{ \bullet \right\}, \quad (3.8)$$

com $\mu > 0$ fixado, $P > 0$, $Q > 0$ e $R > 0$ matrizes de ponderação conhecidas. Como mostra o Teorema 3.2.1, extraído de [27].

Teorema 3.2.1 (Controlador Robusto): Considere o sistema linear sujeito a incertezas paramétricas em (3.3). Suponha que E_{G_i} , $i = 0, \dots, N$ seja uma matriz retangular posto linha pleno. Então, o controlador robusto recursivo para o sistema linear sujeito a incertezas (3.2) é dado por

$$\begin{bmatrix} x_{i+1}^* \\ u_i^* \end{bmatrix} = \begin{bmatrix} L_i \\ K_i \end{bmatrix} x_i; \quad i = 0, \dots, N, \quad (3.9)$$

sendo L_i e K_i obtidos de acordo com a recursão mostrada em (3.10) e P_i é atualizado através de (3.11)

$$\begin{bmatrix} L_i \\ K_i \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} P_i^{-1} & 0 & 0 & 0 & 0 & I & 0 \\ 0 & R_i^{-1} & 0 & 0 & 0 & 0 & I \\ 0 & 0 & Q_i^{-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & -G_i \\ 0 & 0 & 0 & 0 & 0 & 0 & -E_{G_i} \\ I & 0 & 0 & I & 0 & 0 & 0 \\ 0 & I & 0 & -G_i^T & -E_{G_i}^T & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ -I \\ F_i \\ E_{F_i} \\ 0 \\ 0 \end{bmatrix} \quad (3.10)$$

$$P_{i+1} = L_i^T P_i L_i + K_i^T R_i K_i + Q_i; \quad i = 0, \dots, N, \quad (3.11)$$

o que garante a seguinte igualdade

$$E_{F_i} + E_{G_i} K_i = 0 \quad (3.12)$$

quando $\mu \rightarrow \infty$ no funcional mostrado em (3.8).

Note que as equações do controlador apresentadas no Teorema 3.2.1 não dependem de H_i , ou seja as informações sobre as incertezas paramétricas estão contidas nas matrizes E_{F_i} e E_{G_i} . Note ainda que na solução recursiva, para cada $i = 0, \dots, N$ não há necessidade da atualização dos parâmetros do sistema. A forma recursiva encontrada depende somente das matrizes de pa-

râmetros e de ponderação já conhecidas do sistema, o que torna o procedimento mais eficiente, principalmente em aplicações online.

3.3 Controle H_∞ via Teoria dos Jogos

A abordagem do controle H_∞ ótimo tem sido amplamente discutida tanto por sua robustez de estabilidade quanto por sua capacidade de atenuação de distúrbios externos [25].

Agora considere o modelo do bípode descrito na Seção 2.1, mais exatamente a equação 2.1. Considere que cada um dos termos $M(\theta)$, $C(\theta, \dot{\theta})$ e $G(\theta)$ é dividido em uma parte nominal e uma parte contendo as incertezas com relação ao modelo, como mostrado a seguir

$$\begin{aligned} M(\theta) &= M(\theta) + \Delta M \\ C(\theta, \dot{\theta}) &= C(\theta, \dot{\theta}) + \Delta C \\ G(\theta) &= G(\theta) + \Delta G, \end{aligned} \quad (3.13)$$

sendo ΔM , ΔC e ΔG as incertezas paramétricas. O modelo com as incertezas e a adição de um termo de distúrbio de torque d , após alguma manipulação algébrica, fica

$$M(\theta) \ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} + G(\theta) = B\tau + \delta, \quad (3.14)$$

sendo $\delta = -(\Delta M \ddot{\theta} + \Delta C \dot{\theta} + \Delta G - d)$. Note que as incertezas paramétricas são inseridas no modelo como se fossem distúrbios externos.

3.3.1 Dinâmica do Erro

Considere o modelo com incertezas descrito em (3.14), mas escrito da seguinte maneira

$$\ddot{\theta} = -M(\theta)^{-1}C(\theta, \dot{\theta})\dot{\theta} + M(\theta)^{-1}(B\tau - G(\theta) + \delta). \quad (3.15)$$

Adicionando e subtraindo $\ddot{\theta}^d$ e $(-M(\theta)^{-1}C(\theta, \dot{\theta}))\dot{\theta}^d$ na equação (3.15), sendo $\dot{\theta}^d$ velocidades desejadas e $\ddot{\theta}^d$ as acelerações desejadas das juntas do bípode, chega-se na seguinte expressão

$$\ddot{\theta} - \ddot{\theta}^d = -M(\theta)^{-1}C(\theta, \dot{\theta})\left(\dot{\theta} - \dot{\theta}^d\right) + M(\theta)^{-1}\left(B\tau - M(\theta)\ddot{\theta}^d - C(\theta, \dot{\theta})\dot{\theta}^d - G(\theta) + \delta\right). \quad (3.16)$$

Define-se

$$e = \begin{bmatrix} \dot{q} \\ q \end{bmatrix} = \begin{bmatrix} \dot{\theta} - \dot{\theta}^d \\ \theta - \theta^d \end{bmatrix},$$

sendo θ^d as trajetórias desejadas das juntas, tem-se a dinâmica do erro de trajetória das juntas dado por

$$\dot{e} = Ae + B_1u + B_2w, \quad (3.17)$$

sendo

$$\bullet A = \begin{bmatrix} -M(\theta)^{-1}C(\theta, \dot{\theta}) & 0 \\ I & 0 \end{bmatrix};$$

$$\bullet B_1 = \begin{bmatrix} I \\ 0 \end{bmatrix};$$

$$\bullet B_2 = \begin{bmatrix} I \\ 0 \end{bmatrix};$$

$$\bullet u = B\tau - M(\theta)\ddot{\theta}^d - C(\theta, \dot{\theta})\dot{\theta}^d - G(\theta);$$

$$\bullet w = M(\theta)^{-1}\delta.$$

3.3.2 Controlador H_∞

Com o intuito de se eliminar qualquer possível redundância existente no modelo, mantendo a ordem do sistema [28], considera-se a transformação $e = T_0^{-1}z$, sendo

$$T_0 = \begin{bmatrix} T_{11} & T_{12} \\ 0 & I \end{bmatrix},$$

com $T_{11} = t_{11}I$, $t_{11} \in \mathbb{R}^*$.

Aplicando essa transformação em (3.17) e considerando $M(\theta)^{-1}C(\theta, \dot{\theta}) = \alpha(\theta, \dot{\theta})$ tem-se

$$\begin{aligned} \dot{z} &= T_0 \begin{bmatrix} -\alpha & 0 \\ I & 0 \end{bmatrix} T_0^{-1} z + T_0 \begin{bmatrix} I \\ 0 \end{bmatrix} u + T_0 \begin{bmatrix} I \\ 0 \end{bmatrix} w \\ &= \begin{bmatrix} \alpha & 0 \\ T_{11}^{-1} & -T_{11}^{-1}T_{12} \end{bmatrix} z + \begin{bmatrix} T_{12}T_{11}^{-1} & T_{11}\alpha T_{11}^{-1}T_{12} - T_{12}T_{11}^{-1}T_{12} \\ 0 & 0 \end{bmatrix} z + \begin{bmatrix} T_{11} \\ 0 \end{bmatrix} u + \begin{bmatrix} T_{11} \\ 0 \end{bmatrix} w. \end{aligned} \quad (3.18)$$

Agora aplicando-se a transformação inversa, ou seja $z = T_0 e$ à equação (3.18), obtém-se

$$\begin{aligned} \dot{e} &= T_0^{-1} \begin{bmatrix} -\alpha & 0 \\ T_{11}^{-1} & -T_{11}^{-1}T_{12} \end{bmatrix} T_0 e + T_0^{-1} \begin{bmatrix} T_{12}T_{11}^{-1} & T_{11}\alpha T_{11}^{-1}T_{12} - T_{12}T_{11}^{-1}T_{12} \\ 0 & 0 \end{bmatrix} T_0 e + T_0^{-1} \begin{bmatrix} T_{11} \\ 0 \end{bmatrix} w \\ &= T_0^{-1} \begin{bmatrix} -\alpha & 0 \\ T_{11}^{-1} & -T_{11}^{-1}T_{12} \end{bmatrix} T_0 e + T_0^{-1} \begin{bmatrix} T_{12}\dot{q} + M(\theta)^{-1}C(\theta, \dot{\theta})T_{12}q \\ 0 \end{bmatrix} + T_0^{-1} \begin{bmatrix} T_{11} \\ 0 \end{bmatrix} u + T_0^{-1} \begin{bmatrix} T_{11} \\ 0 \end{bmatrix} w \\ &= T_0^{-1} \begin{bmatrix} -\alpha & 0 \\ T_{11}^{-1} & -T_{11}^{-1}T_{12} \end{bmatrix} T_0 e + T_0^{-1} \begin{bmatrix} M(\theta)^{-1} \\ 0 \end{bmatrix} T_{11} \cdot \\ &\quad \left[-M(\theta) (\ddot{\theta}^d - T_{11}^{-1}T_{12}\dot{q}) - C(\theta, \dot{\theta}) (\dot{\theta}^d - T_{11}^{-1}T_{12}q) + B\tau - G(\theta) \right] + T_0^{-1} \begin{bmatrix} M(\theta)^{-1} \\ 0 \end{bmatrix} T_{11}w. \end{aligned} \quad (3.19)$$

Finalmente fazendo as identificações

- $A_T = T_0^{-1} \begin{bmatrix} -\alpha & 0 \\ T_{11}^{-1} & -T_{11}^{-1}T_{12} \end{bmatrix} T_0$
- $B_T = T_0^{-1} \begin{bmatrix} M(\theta)^{-1} \\ 0 \end{bmatrix}$
- $F = M(\theta) (\ddot{\theta}^d - T_{11}^{-1}T_{12}\dot{q}) + C(\theta, \dot{\theta}) (\dot{\theta}^d - T_{11}^{-1}T_{12}q) + G(\theta)$
- $w = T_{11}w$
- $u = T_{11}(-F + B\tau)$

tem-se

$$\dot{e} = A_T e + B_T u + B_T w. \quad (3.20)$$

Para a dedução do controlador H_∞ , e afim de atenuar possíveis distúrbios no sistema, o problema de controle é transformado em um problema não linear de controle de custo sujeito a uma restrição *minimax*. A teoria dos jogos diferenciais é utilizada para tratar o problema *minimax* através da solução de uma equação de Bellman-Isaacs, levando a uma condição suficiente para a solução do problema de controle baseado em modelo com desejado nível de atenuação de distúrbios. A estratégia proporciona uma solução ótima para o problema de acompanhamento de trajetória na ausência de perturbações, além de ser capaz de limitar o efeito do pior caso de perturbação durante o acompanhamento de trajetória a um nível desejado γ .

Considera-se que o termo F da equação (3.20) é bem determinado e está disponível para o controle. Os efeitos do distúrbio combinado no sistema, w , devem ser minimizados pela estratégia de controle H_∞ através de uma realimentação de estado da forma $u = K(x)x$. Com essa intenção, e sujeito à dinâmica do erro de acompanhamento, o seguinte critério de desempenho é proposto

$$\min_{u(\cdot) \in \mathcal{L}_2} \max_{0 \neq w(\cdot) \in \mathcal{L}_2} \frac{\int_0^\infty \left(\frac{1}{2} e^T Q e + \frac{1}{2} u^T R u \right) dt}{\int_0^\infty \left(\frac{1}{2} w^T w \right) dt} \leq \gamma^2, \quad (3.21)$$

sendo $e = e(t)$, $u = u(t)$ e $w = w(t)$, γ o nível desejado de atenuação dos distúrbios e $Q = Q^T > 0$, $R = R^T > 0$ e $P_0 = P_0^T > 0$ matrizes de ponderação.

De acordo com a teoria dos jogos, a solução do problema *minimax*, (3.21), é encontrada se existe uma função de Lyapunov contínua e diferenciável $V(e, t)$, que satisfaça a seguinte equação de Bellman-Isaacs:

$$-\frac{\partial V(e, t)}{\partial t} = \min_{u(\cdot)} \max_{w(\cdot)} \left\{ L(e, u, w) + \left(\frac{\partial V(e, t)}{\partial e} \right)^T e \right\},$$

com a condição final $V(e(\infty), \infty) = 0$ e sendo $L(e, u, w)$ o Lagrangiano dado por

$$L(e, u, w) = \frac{1}{2} e^T(t) Q e(t) + \frac{1}{2} u^T(t) R u(t) - \frac{1}{2} \gamma^2 w^T(t) w(t).$$

Uma possível equação de Lyapunov pode ser definida como

$$V(e, t) = \frac{1}{2} e^T P(e, t) e, \quad (3.22)$$

sendo $P(e, t)$ uma matriz simétrica positiva definida para todo e e t , a equação de Bellman-Isaacs torna-se a seguinte equação de Riccati

$$\begin{aligned} \dot{P}(e, t) + P(e, t) A_T(q, \dot{q}) + A_T^T(q, \dot{q}) P(e, t) + \\ - P(e, t) B_T(q) \left(R^{-1} - \frac{1}{\gamma^2} I \right) B_T^T(q) P(e, t) + Q = 0. \end{aligned} \quad (3.23)$$

Para que (3.23) satisfaça (3.21), a lei de controle ótima u^* e o pior caso de distúrbio w^* são dados por

$$u^* = -R^{-1}B_T^T(q)P(e,t)e, \quad (3.24)$$

$$w^* = \frac{1}{\gamma^2}B_T^T(q)P(e,t)e. \quad (3.25)$$

3.3.3 Simplificação da Equação de Riccati

Observando a equação de Riccati (3.23), nota-se uma certa dificuldade em sua implementação devido a sua forma diferencial. Com o intuito de simplificar essa expressão escolhe-se, apropriadamente, a matriz P como sendo

$$P(e,t) = T_0^T \begin{bmatrix} M(\theta) & 0 \\ 0 & K \end{bmatrix} T_0, \quad (3.26)$$

sendo K uma matriz constante, simétrica e definida positiva representando a rigidez na oscilação em torno da posição de referência desejada e $M(\theta)$ representando a energia cinética.

Dessa maneira pode-se entender a função de Lyapunov escolhida como uma função da energia total do estado e no tempo t . A condição final $V(e(\infty), \infty) = 0$ também é satisfeita para $P(e,t)$ na forma apresentada em (3.26).

Assim, a lei de controle ótima u^* e o pior caso de distúrbio w^* podem ser reescritos

$$u^* = -R^{-1}B^T T_0 e, \quad (3.27)$$

$$w^* = \frac{1}{\gamma^2}B^T T_0 e \quad (3.28)$$

e utilizando (3.26), a equação (3.23) torna-se a equação algébrica matricial

$$\begin{bmatrix} 0 & K \\ K & 0 \end{bmatrix} - T_0^T B \left(R^{-1} - \frac{1}{\gamma^2} I \right) B^T T_0 + Q = 0, \quad (3.29)$$

portanto para resolver o problema de controle H_∞ , deve-se encontrar matrizes K e T_0 que solucionem a equação (3.29).

3.3.4 Cálculo de K e T_0

Para se obter os valores de K e T_0 , considera-se a matriz Q simétrica definida positiva fatorada na forma

$$Q = \begin{bmatrix} Q_1^T Q_1 & Q_{12} \\ Q_{12}^T & Q_2^T Q_2 \end{bmatrix}. \quad (3.30)$$

A equação (3.29) fornece quatro equações distintas

$$Q_1^T Q_1 - T_{11}^T \left(R^{-1} - \frac{1}{\gamma^2} I \right) T_{11} = 0, \quad (3.31)$$

$$K + Q_{12}^T - T_{12}^T \left(R^{-1} - \frac{1}{\gamma^2} I \right) T_{11} = 0, \quad (3.32)$$

$$K + Q_{12} - T_{11}^T \left(R^{-1} - \frac{1}{\gamma^2} I \right) T_{12} = 0, \quad (3.33)$$

$$Q_2^T Q_2 - T_{12}^T \left(R^{-1} - \frac{1}{\gamma^2} I \right) T_{12} = 0. \quad (3.34)$$

Considerando-se também que R_1 é o resultado da fatoração de Cholesky

$$R_1^T R_1 = \left(R^{-1} - \frac{1}{\gamma^2} I \right)^{-1}, \quad (3.35)$$

a solução da equação (3.29) é dada por:

$$T_0 = \begin{bmatrix} R_1^T Q_1 & R_1^T Q_2 \\ 0 & I \end{bmatrix} \quad (3.36)$$

$$K = \frac{1}{2} (Q_1^T Q_2 - Q_2^T Q_1) - \frac{1}{2} (Q_{21}^T + Q_{12}). \quad (3.37)$$

Como $Q_1^T Q_1$ e $Q_2^T Q_2$ são definidas positivas, de (3.31) e (3.34), a matriz $\left(R^{-1} - \frac{1}{\gamma^2} I \right)$ deve ser definida positiva, e portanto, $R < \gamma^2 I$. Observa-se que o valor de γ é limitado pela norma da matriz de ponderação da entrada de controle R , ou seja, um valor muito baixo de γ levará a um controle de alto ganho.

3.4 Controle H_∞ via LMI

Neste controle H_∞ a garantia de que o ganho $\mathcal{L}_2$ entre o distúrbio e a saída seja limitado por um nível de atenuação γ é representada pela existência de solução para uma equação diferencial parcial de Hamilton-Jacobi. Entretanto, a obtenção de tal solução pode ser consideravelmente complexa para sistemas com muitos estados. Um procedimento alternativo consiste

em representar o sistema não linear como um sistema quase-Linear com Parâmetros Variantes (quase-LPV), no qual os parâmetros dependem do tempo e dos estados do sistema, e utilizar as soluções via Desigualdades Matriciais Lineares (DMLs ou do inglês LMIs) desenvolvidas para sistemas Lineares com Parâmetros Variantes (LPV), nos quais os parâmetros depende apenas do tempo, [29]. Em [30, 31, 32], o controle H_∞ não linear via representação quase-LPV foi aplicado no robô manipulador subatuado UArm II, obtendo resultados que comprovaram a rejeição aos distúrbios.

Considere o seguinte sistema LPV, sendo x as variáveis de estado, u a variável de controle e $z = [z_1 \ z_2]^T$ a variável de saída:

$$\begin{bmatrix} \dot{x} \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} A(\rho) & B_1(\rho) & B_2(\rho) \\ C_1(\rho) & 0 & 0 \\ C_2(\rho) & 0 & I \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix}. \quad (3.38)$$

Assume-se que o parâmetro ρ varie no conjunto permissível:

$$F_P^v = \{\rho \in \mathcal{C}^1(\mathfrak{R}^+, \mathfrak{R}^m) : \rho(t) \in P, |\dot{\rho}_i| \leq v_i\}$$

para $i = 1, \dots, m$, sendo $P \subset \mathfrak{R}^m$ um conjunto compacto, e $v = [v_1 \dots v_m]^T$ com $v_i \geq 0$. O sistema (3.38) possui um ganho $\mathcal{L}_2 \leq \gamma$ no intervalo $[0, T]$ se:

$$\int_0^T \|z(t)\|^2 dt \leq \gamma^2 \int_0^T \|w(t)\|^2 dt \quad (3.39)$$

para todo $T \geq 0$ e todo $w \in \mathcal{L}_2(0, T)$ com o sistema iniciando de $x(0) = 0$.

O objetivo é encontrar uma função contínua em $\rho(t)$, $F(\rho(t))$, tal que o sistema em malha fechada possua ganho $\mathcal{L}_2$ menor que γ com lei de realimentação do estado igual a $u(t) = F(\rho(t))x(t)$.

Lema 3.4.1 [29] *Se existe uma função matricial contínua diferenciável $X(\rho) > 0$ para todo $\rho \in P$ que satisfaz*

$$\begin{bmatrix} E(\rho) & X(\rho)C_1^T(\rho) & B_1(\rho) \\ C_1(\rho)X(\rho) & -I & 0 \\ B_1^T(\rho) & 0 & -\gamma^2 I \end{bmatrix} < 0 \quad (3.40)$$

sendo

$$E(\rho) = -\sum_{i=1}^m \pm \left(v_i \frac{\partial X}{\partial \rho_i} \right) - B_2(\rho)B_2^T(\rho) + \hat{A}(\rho)X(\rho) + X(\rho)\hat{A}(\rho)^T$$

$\hat{A}(\rho) = A(\rho) - B_2(\rho)C_2(\rho)$, então, com lei de realimentação do estado

$$u(t) = -(B_2(\rho(t))X^{-1}(\rho(t)) + C_2(\rho(t)))x(t) \quad (3.41)$$

o sistema em malha fechada possui ganho $\mathcal{L}_2 \leq \gamma$ para toda trajetória paramétrica $\rho(t) \in F_P^\gamma$.

Portanto, é preciso resolver a DML paramétrica (3.40), que é um problema de otimização convexo com dimensão infinita. Um esquema computacional prático [29] pode ser utilizado para resolver as DMLs presentes em (3.40). Tal esquema consiste na escolha de um conjunto de funções $\mathcal{C}^1$, $\{f_i(\rho(t))\}_{i=1}^M$, como base para $X(\rho)$, ou seja,

$$X(\rho(t)) = \sum_{i=1}^M f_i(\rho(t))X_i, \quad (3.42)$$

sendo $X_i \in S^{n \times n}$ a matriz coeficiente para $f_i(\rho(t))$.

Note que (3.40) agora são DMLs em termos das variáveis matriciais $\{X_i\}_{i=1}^M$ que devem ser satisfeitas para todo parâmetro $\rho(t)$ em P . Para resolver este problema de otimização de dimensão infinita, divide-se o conjunto de parâmetros P em L pontos $\{\rho_k\}_{k=1}^L$ em cada dimensão. Então calcula-se as DMLs acima para estes pontos. Desde que (3.40) consiste em 2^m vínculos, um total de $(2^m + 1)L^m$ desigualdades matriciais afins em termos das M variáveis matriciais $\{X_i\}$ devem ser resolvidas.

4 *Resultados*

Neste capítulo são apresentados os principais resultados de simulação da implementação dos controladores descritos no Capítulo 3 aplicados ao bípede descrito no Capítulo 2, utilizando os valores de massa e comprimento mostrados na Tabela 4.1.

Tabela 4.1: Parâmetros do robô bípede utilizados.

Parâmetro	Valor	Unidade
m_1	5	kg
m_2	3,5	kg
m_3	1,5	kg
m_H	10	kg
m_T	10	kg
l_1	1	m
l_2	0,5	m
l_3	0,5	m
l_T	0,5	m
c_1	0,47	m
c_2	0,35	m
c_3	0,25	m
c_T	0,5	m

As simulações foram feitas utilizando-se um simulador desenvolvido no laboratório e descrito no Anexo A. Neste simulador é adotado a estratégia de controle descrita em 4.1.

Para averiguar a eficácia dos controladores de forma quantitativa foram adotados dois índices de desempenho:

1. Norma $\mathcal{L}_2$ do erro de acompanhamento dos estados

$$\mathcal{L}_2[e] = \left(\frac{1}{(t_r - t_0)} \int_{t_0}^{t_r} \|e(t)\|_2^2 dt \right)^{1/2},$$

em que $\|\cdot\|_2$ a norma Euclidiana;

2. Soma dos torques aplicados

$$E[\tau] = \sum_{i=1}^2 \left(\int_{t_0}^{t_r} |\tau(t)| dt \right),$$

em que t_r o tempo gasto pelo robô para alcançar a posição desejada.

sendo norma $\mathcal{L}_2$ do erro de acompanhamento da trajetória um índice de desempenho amplamente utilizado na literatura [33, 34, 35] e a análise da soma dos torques aplicados estando diretamente relacionada ao consumo de energia.

4.1 Estratégia de Controle

A compensação da gravidade fornece um caminhar estável em inclinações diferentes para posição inicial próxima do ciclo limite. Entretanto, aplicações práticas requerem grande bacia de atração e alta robustez na presença de distúrbios, dinâmicas não modeladas e terrenos irregulares. O controle híbrido proposto em [36] combina os controladores descritos acima para produzir um caminhar robusto.

Durante um passo, é calculada a distância mínima entre a posição atual e o ciclo limite de referência obtido para uma certa inclinação. Esta distância é dada por:

$$\bar{d} = \min_i (\text{norm}(\theta - \theta_r(i))), \quad (4.1)$$

sendo norm a norma $\mathcal{L}_2$ e $\theta_r(i)$ os pontos discretos do ciclo limite de referência. Esta distância é comparada com um valor limite C , que corresponde a uma medida da bacia de atração do ciclo limite. Se a posição inicial está dentro deste limite C , o robô bípede convergirá para o ciclo limite. Neste caso, utiliza-se o controle de compensação da gravidade e o controle do tronco.

Agora, caso a distância $\bar{d}$ seja maior que o valor C , ou seja, se a trajetória está fora da bacia de atração, utiliza-se uma estratégia de controle baseada nos controles descritos no Capítulo 3 para fazer com que a trajetória retorne para o ciclo limite.

O ponto de mudança dos controles descritos no Capítulo 3 para o controle de compensação da gravidade e do tronco deve ser escolhido de forma que o caminhar seja suave e natural. Observou-se que o melhor ponto de mudança é o momento do impacto do joelho. Neste ponto, a perna de suporte já alcançou ou ultrapassou o eixo normal com o solo e, portanto, não há perigo de ocorrer a queda para trás. A trajetória desejada, θ^d , entre a posição em que o robô deixa a bacia de atração do ciclo limite e a posição do impacto do joelho é definida neste trabalho por um polinômio do quinto grau, com tempo de duração que corresponde ao tempo restante para o impacto no joelho.

Os controles descritos no Capítulo 3 são aplicado apenas durante a fase antes do impacto com o joelho. Se o robô deixa a bacia de atração do ciclo limite após o impacto com o joelho ou, após a utilização deste controlador na fase pré-impacto no joelho, a posição do robô está fora da bacia de atração, utiliza-se os controles da gravidade e do tronco até o instante de colisão com o solo e em seguida é aplicado o controlador ativo até o próximo impacto com o joelho.

4.2 Simulação na presença de distúrbio

Nesta Seção serão mostrados os resultados obtidos quando se aplica ao bípede um distúrbio de torque, como mostrado na Figura 4.1, durante a fase inicial do primeiro passo.

Figura 4.1: Distúrbios de torque aplicados nas juntas dos bípides durante as simulações.

4.2.1 Controlador Robusto Recursivo

Para a implementação do controlador robusto descrito na Seção 3.2 é necessário considerar um modelo linear a parâmetros variantes. Note a possibilidade de se escrever o modelo da dinâmica do erro descrito em (3.17) como quase-linear a parâmetros variantes, em termos de espaço de estado, já que não é possível obter-se parâmetros independentes dos estados para gerar um modelo linear a parâmetros variantes.

Em virtude do modelo do robô bípede considerado atuar descendo um plano inclinado, as variações nos ângulos das juntas são relativamente pequenas. Veja o retrato de fase da Figura 2.3. Sendo também as velocidades relativamente baixas, as variações da norma da matriz $A(\theta, \dot{\theta})$ são relativamente pequenas. Portanto, pode-se utilizar o modelo (3.17) diretamente no projeto do controlador considerado. Para isto precisa-se discretizar esse modelo. Na discretização é utilizada uma taxa de amostragem variável de acordo com a integração realizada pelo simulador. Após as manipulações para que a equação (3.17) fique na forma descrita em (3.2) foram feitas as simulações do robô com as incertezas paramétricas, sendo a matriz $F(\theta, \dot{\theta})$ calculada a cada instante com os respectivos valores de θ e $\dot{\theta}$, ou seja, para um determinado conjunto de posições das juntas e suas respectivas velocidades para que o caminhar seja alcançado conforme Figura 2.3.

As incertezas descritas em (3.3) foram consideradas como sendo

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad (4.2)$$

E_{F_i} e E_{G_i} foram calculadas através dos valores singulares das matrizes F e G do sistema nominal ao longo do tempo durante o passo.

Quando o controlador da Seção 3.2 é implementado, com as matrizes de ponderação dadas por $P = I_{8 \times 8}$, $Q = 0,1 \cdot I_{8 \times 8}$ e $R = 10 \cdot I_{4 \times 4}$, o sistema permanece estável durante todo o ciclo de caminhar, mesmo na presença de distúrbios. A Figura 4.2 mostra o comportamento das juntas do bípede durante um passo. Percebe-se, através da análise da Figura 4.2, que as trajetórias tendem a referência desejada (curva vermelha) conforme os passos vão acontecendo, demonstrando a eficácia do controlador.

Figura 4.2: Retratos de fase das juntas do robô durante 3 passos na presença de distúrbio utilizando o controlador robusto recursivo.

4.2.2 Controlador H_∞ via Teoria dos Jogos

Como o controlador H_∞ descrito na Seção 3.3 é um controlador não linear, não existe necessidade de se fazer as considerações realizadas anteriormente para a aplicação do controlador robusto recursivo. A fim de se aplicar o controlador H_∞ em simulações, foram adotados os seguintes valores para as matrizes de ponderação: $R = 0,5$, $Q_1 = 0,7 \cdot I_4$, $Q_2 = 0,7 \cdot I_4$, $Q_{12} = 0_4$ e $\gamma = 5$. Utilizando as equações em (3.36) e (3.37) e considerando que a fatoração de Cholesky da equação (3.35) resulta em $R_1 = 0,7746$ tem-se $K = 0_{3 \times 3}$ e

$$T_0 = \begin{bmatrix} 0,5422 & 0 & 0 & 0 & 0,5422 & 0 & 0 & 0 \\ 0 & 0,5422 & 0 & 0 & 0 & 0,5422 & 0 & 0 \\ 0 & 0 & 0,5422 & 0 & 0 & 0 & 0,5422 & 0 \\ 0 & 0 & 0 & 0,5422 & 0 & 0 & 0 & 0,5422 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

As simulações realizadas também envolvem a aplicação de distúrbios. Os resultados são mostrados na Figura 4.3, onde se percebe, como no controlador robusto recursivo, que as trajetórias convergem para o desejado (linha pontilhada) ao longo dos passos.

Figura 4.3: Retratos de fase das juntas do robô na presença de distúrbios sob ação do controle H_∞ por Teoria dos Jogos.

4.2.3 Controlador H_∞ via LMI

Para aplicar o controlador H_∞ via LMI descrito na Seção 3.4, o sistema de controle do robô bípede deve ser representado pela Equação (3.38). Portanto, escolhe-se como parâmetros os estados representando os erros de posição das juntas 1 e 2, ou seja, $m = 2$ e

$$\rho(e) = \begin{bmatrix} e_1 & e_2 \end{bmatrix}^T$$

sendo e_1 e e_2 os erros de posição das juntas 1 e 2, respectivamente. Foram escolhidos apenas dois elementos para os parâmetros pois, uma escolha de ρ que também considere os demais erros de posição e os erros de velocidade das juntas, ou seja, ρ contendo 8 elementos, faz com que número de desigualdades matriciais a serem resolvidas cresça absurdamente (veja Seção 3.4).

Consideram-se como saídas do sistema, z_1 e z_2 , os erros de posição e velocidade representados pelo estado e a variável de controle u , respectivamente. Portanto o sistema pode ser descrito pela Equação (3.38) com:

$$\begin{aligned} A(\rho(x)) &= A(\rho(\tilde{x})) \\ B_1(\rho(x)) &= B_1 \\ B_2(\rho(x)) &= B_2 \\ C_1(\rho(x)) &= I \\ C_2(\rho(x)) &= 0 \end{aligned}$$

sendo as matrizes $A(\rho(\tilde{x}))$, B_1 e B_2 obtidas da Equação (3.17).

Os possíveis valores para os parâmetros estão contidos no conjunto compacto, P , definido por $\rho \in [-0.3, 0.3]rad \times [-0.4, 0.4]rad$. A taxa de variação dos parâmetros é limitada por $|\dot{\rho}| \leq [1, 4 \ 2, 0]rad/s$.

Como visto anteriormente, uma alternativa para a escolha das funções utilizadas como base para $X(\rho)$, consiste em utilizar funções contidas nas matrizes de estado. As funções trigonométricas *seno* e *coseno* estão presentes nas matrizes $M(\theta)$ e $C(\theta, \dot{\theta})$ que compõem a matriz de estado A , e portanto, as funções escolhidas foram:

$$\begin{aligned} f_1(\rho(e)) &= 1 \\ f_2(\rho(e)) &= \cos(e_1) \\ f_3(e) &= \cos(e_2). \end{aligned}$$

O espaço dos parâmetros foi dividido em 15 pontos ($L = 15$). Cada um dos 225 sistemas lineares, L^m , gerados pela combinação dos parâmetros ρ fornece 5 DMLs, $1 + 2^m$. Ou seja, 1125 DMLs devem ser resolvidas simultaneamente para as variáveis X_i , $i = (1, 2, 3)$ (veja Seção 3.4). Para resolver este problema foi utilizado o *toolbox Linear Matrix Inequalities (LMI)* do MatLab[®]. O valor de atenuação mínimo encontrado foi $\gamma = 1.5$. Os valores das matrizes X_i são dados por:

$$X_1 = \begin{bmatrix} 0.0658 & -0.0169 & -0.0016 & -0.0001 & -0.0787 & 0.0143 & 0.0014 & 0.0005 \\ -0.0169 & 0.2287 & 0.0140 & 0.0123 & 0.0134 & -0.2186 & -0.0104 & -0.0090 \\ -0.0016 & 0.0140 & 0.2459 & -0.0074 & 0.0022 & -0.0105 & -0.2317 & 0.0054 \\ -0.0001 & 0.0123 & -0.0074 & 0.2464 & 0.0009 & -0.0090 & 0.0054 & -0.2327 \\ -0.0787 & 0.0134 & 0.0022 & 0.0009 & 0.1672 & -0.0162 & -0.0014 & -0.0004 \\ 0.0143 & -0.2186 & -0.0105 & -0.0090 & -0.0162 & 0.3306 & 0.0033 & 0.0030 \\ 0.0014 & -0.0104 & -0.2317 & 0.0054 & -0.0014 & 0.0033 & 0.3370 & -0.0017 \\ 0.0005 & -0.0090 & 0.0054 & -0.2327 & -0.0004 & 0.0030 & -0.0017 & 0.3374 \end{bmatrix},$$

$$X_2 = \begin{bmatrix} 0.0441 & 0.0003 & 0.0019 & 0.0018 & -0.0188 & -0.0006 & -0.0002 & -0.0003 \\ 0.0003 & -0.0045 & 0.0026 & 0.0009 & 0.0012 & -0.0019 & 0.0014 & 0.0015 \\ 0.0019 & 0.0026 & -0.0014 & -0.0004 & -0.0017 & 0.0011 & -0.0009 & -0.0009 \\ 0.0018 & 0.0009 & -0.0004 & 0.0003 & -0.0016 & 0.0013 & -0.0009 & -0.0010 \\ -0.0188 & 0.0012 & -0.0017 & -0.0016 & 0.0003 & -0.0002 & 0.0000 & 0.0001 \\ -0.0006 & -0.0019 & 0.0011 & 0.0013 & -0.0002 & 0.0000 & -0.0000 & -0.0000 \\ -0.0002 & 0.0014 & -0.0009 & -0.0009 & 0.0000 & -0.0000 & 0.0000 & 0.0000 \\ -0.0003 & 0.0015 & -0.0009 & -0.0010 & 0.0001 & -0.0000 & 0.0000 & 0.0000 \end{bmatrix},$$

e

$$X_3 = \begin{bmatrix} 0.0149 & 0.0008 & 0.0007 & 0.0003 & -0.0063 & -0.0007 & -0.0002 & 0.0000 \\ 0.0008 & -0.0006 & 0.0005 & 0.0011 & 0.0003 & -0.0009 & 0.0006 & 0.0003 \\ 0.0007 & 0.0005 & -0.0002 & -0.0006 & -0.0007 & 0.0004 & -0.0004 & -0.0002 \\ 0.0003 & 0.0011 & -0.0006 & -0.0007 & -0.0005 & 0.0002 & -0.0002 & -0.0001 \\ -0.0063 & 0.0003 & -0.0007 & -0.0005 & 0.0001 & 0.0000 & 0.0000 & -0.0000 \\ -0.0007 & -0.0009 & 0.0004 & 0.0002 & 0.0000 & 0.0001 & 0.0000 & -0.0000 \\ -0.0002 & 0.0006 & -0.0004 & -0.0002 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0003 & -0.0002 & -0.0001 & -0.0000 & -0.0000 & 0.0000 & 0.0000 \end{bmatrix}.$$

As simulações, mais uma vez foram feitas adicionando-se distúrbios de torque nas juntas do robô. Os resultados são mostrados na Figura 4.4, onde se percebe, como no controlador robusto recursivo, que as trajetórias convergem para o desejado (linha vermelha) ao longo dos passos.

Figura 4.4: Retratos de fase das juntas do robô na presença de distúrbios sob ação do controle H_∞ por LMI.

Por fim a Tabela 4.2 mostra os valores dos índices de desempenho obtidos para os três controladores nas condições mencionadas anteriormente.

Tabela 4.2: Índices de Desempenho - Simulação

	$\mathcal{L}_2[e]$	$E[\tau](\text{Nms})$
Robusto Recursivo	25.1674	67.4141
H_∞ Teoria dos Jogos	25.1520	67.4080
H_∞ LMI	25.2811	67.5197

4.3 Simulação na presença de distúrbios externos e incertezas paramétricas

Agora são mostrados os resultados obtidos, quando é acrescentado no modelo do bípede incertezas em sua matriz de inércia da ordem de 20% e aplicando-se o distúrbio de torque mostrado na Figura 4.1. Foram utilizados os mesmos controladores da Seção 4.2.

4.3.1 Controlador Robusto Recursivo

Os gráficos mostrados na Figura 4.5 mostram os retratos de fase do bípede. Como se pode observar nos retratos de fase, percebe-se que o controle é efetivo ao fazer com que o robô volte ao ciclo limite após alguns passos.

Figura 4.5: Retratos de fase das juntas do robô na presença de incertezas sob ação do controle Robusto Recursivo.

4.3.2 Controlador H_∞ via Teoria dos Jogos

Os gráficos mostrados na Figura 4.6 também mostram os retratos de fase das juntas do bípede. Como se pode observar nos retratos de fase, percebe-se que o controle é efetivo ao fazer com que o robô volte ao ciclo limite após alguns passos.

Figura 4.6: Retratos de fase das juntas do robô na presença de incertezas sob ação do controle H_∞ via Teoria dos Jogos.

4.3.3 Controlador H_∞ via LMI

Os gráficos mostrados na Figura 4.7 uma vez mais mostram os retratos de fase das juntas do bípede. Como se pode observar nos retratos de fase, percebe-se que o controle é efetivo ao fazer com que o robô volte ao ciclo limite após alguns passos.

Figura 4.7: Retratos de fase das juntas do robô na presença de incertezas sob ação do controle H_∞ via LMI.

A Tabela 4.3 mostra os valores dos índices de desempenho obtidos para os três controladores quando o bípede foi simulado contendo incertezas em sua matriz de inércia da ordem de 20%.

Tabela 4.3: Índices de Desempenho - Simulação

	$\mathcal{L}_2[e]$	$E[\tau](\text{Nms})$
Robusto Recursivo	29.4539	70.5523
H_∞ Teoria dos Jogos	29.4409	70.5473
H_∞ LMI	29.5501	70.6255

5 *Conclusões*

Neste trabalho foi apresentado um modelo de robô bípede com caminhar passivo, cuja principal vantagem está no fato de que, estando dentro de sua bacia de atração, ele converge para um ciclo limite estável sem a necessidade de ações de controle nas juntas.

Dessa maneira apresentou-se três tipos de controladores robustos. O primeiro é um controlador recursivo deduzido para lidar com incertezas paramétricas em sistemas lineares discretos no tempo. O segundo controlador é um H_∞ obtido através da Teoria dos Jogos desenvolvido para lidar com distúrbios externos em sistemas não lineares em tempo contínuo. E por fim, o terceiro controlador também é um controlador com critério H_∞ , mas obtido através da solução de LMIs.

Todos os controladores apresentados foram efetivos na obtenção de um caminhar estável sob a influência tanto de distúrbios externos como de incertezas paramétricas. No entanto analisando-se os índices de desempenho na Tabelas 4.2 e 4.3 nota-se uma tendência de que o controlador H_∞ obtido através da Teoria dos Jogos exiba um desempenho ligeiramente melhor que a dos outros dois. No entanto devido ao fato de serem resultados de simulação não se pode inferir que este controlador tenha um desempenho melhor em relação aos demais em uma situação real. Ainda deve se levar em conta que os controladores H_∞ via LMI e o controlador Robusto Recursivo foram desenvolvidos especialmente para o tratamento de distúrbios externos e incertezas paramétricas, o que leva a se acreditar que estes tenham um resultado melhor em situações reais.

5.1 **Trabalhos Futuros**

Para dar continuidade ao projeto propõe-se a implementação desses mesmos controladores em um modelo de robô bípede planar que contenha pés e em um modelo tridimensional, fazendo com que as simulações fiquem mais condizentes com a realidade.

Referências Bibliográficas

- [1] E. Garcia, M. Jimenez, P. De Santos, and M. Armada, “The evolution of robotics research,” *IEEE Robotics & Automation Magazine.*, vol. 14, no. 1, pp. 90–103, March 2007.
- [2] A. Goswami, B. Thuilot, and B. Espiau, “A study of the passive gait of a compass-like biped robot: Symmetry and chaos,” *INTERNATIONAL JOURNAL OF ROBOTICS RESEARCH*, vol. 17, pp. 1282–1301, 1998.
- [3] H. Hirukawa, F. Kanehiro, K. Kaneko, S. Kajita, K. Fujiwara, Y. Kawai, F. Tomita, S. Hirai, K. Tanie, T. Isozumi, K. Akachi, T. Kawasaki, S. Ota, K. Yokoyama, H. Handa, Y. Fukase, J. Maeda, Y. Nakamura, S. Tachi, and H. Inoue, “Humanoid robotics platforms developed in hrp,” *Robotics and Autonomous Systems*, vol. 48, no. 4, pp. 165 – 175, 2004, humanoids 2003.
- [4] M. Vukobratovic and B. Borovac, “Dynamic balance concept and the maintenance of the dynamic balance in humanoid robotics,” in *6th International Symposium on Intelligent Systems and Informatics, SISY 2008.*, sept. 2008, pp. 1 –11.
- [5] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer, 2008.
- [6] C. Chevallereau, D. Djoudi, and J. Grizzle, “Stable bipedal walking with foot rotation through direct regulation of the zero moment point,” *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 390–401, April 2008.
- [7] J. Pratt, B. Krupp, C. Morse, and S. Collins, “The roboknee: an exoskeleton for enhancing strength and endurance during walking,” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 3, april-1 may 2004, pp. 2430 – 2435 Vol.3.
- [8] B. Jardim, “Atuadores elásticos em série aplicados no desenvolvimento de um exoesqueleto para membros inferiores,” Master’s thesis, Departamento de Engenharia Mecânica da Escola de Engenharia de São Carlos - Universidade de São Paulo, 2009.
- [9] J. S. Bay and H. Hemami, “Modeling of a neural pattern generator with coupled nonlinear oscillators,” *IEEE Transactions on Biomedical Engineering.*, vol. BME-34, no. 4, pp. 297–306, April 1987.
- [10] N. Sadati and K. Hamed, “Neural control of a fully actuated biped robot,” *IEEE International Conference on Robotics and Biomimetics, ROBIO '06.*, pp. 1299–1304, Dec. 2006.
- [11] K. Tsuchiya, S. Aoi, and K. Tsujita, “Locomotion control of a biped locomotion robot using nonlinear oscillators,” *IEEE/RSJ International Conference on Intelligent Robots and Systems. (IROS). Proceedings.*, vol. 2, pp. 1745–1750, Oct. 2003.

- [12] S. Aoi and K. Tsuchiya, "Locomotion control of a biped robot using nonlinear oscillators," *Autonomous Robots*, vol. 19, no. 3, pp. 219–232, 2005.
- [13] L. Jalics, H. Hemami, and Y. Zheng, "Pattern generation using coupled oscillators for robotic and biorobotic adaptive periodic movement," *IEEE International Conference on Robotics and Automation, 1997. Proceedings.*, vol. 1, pp. 179–184 vol.1, Apr 1997.
- [14] T. McGeer, "Passive dynamic walking," *International Journal of Robotics Research*, vol. 9, no. 2, pp. 62–82, 1990.
- [15] M. Garcia, A. Chatterjee, and A. Ruina, "Speed, efficiency, and stability of small-slope 2d passive-dynamic bipedal walking," in *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Piscataway, USA, 1998.
- [16] F. Asano, Z.-W. Luo, and M. Yamakita, "Biped gait generation and control based on a unified property of passive dynamic walking," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 754–762, Aug. 2005.
- [17] D. Hobbelen and M. Wisse, "Swing-leg retraction for limit cycle walkers improves disturbance rejection," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 377–389, April 2008.
- [18] J. Morimoto, G. Endo, J. Nakanishi, and G. Cheng, "A biologically inspired biped locomotion strategy for humanoid robots: Modulation of sinusoidal patterns by a coupled oscillator model," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 185–191, Feb. 2008.
- [19] A. A. G. Siqueira and M. H. Terra, "Nonlinear h-infinity control applied to biped robots," in *Computer Aided Control System Design, IEEE International Conference on Control Applications, IEEE International Symposium on Intelligent Control*, oct. 2006, pp. 2190–2195.
- [20] T. F. P. A. Taveira, A. A. G. Siqueira, and M. H. Terra, "Adaptive nonlinear h-infinity; controllers applied to a free-floating space manipulator," in *Computer Aided Control System Design, IEEE International Conference on Control Applications, IEEE International Symposium on Intelligent Control*, 2006, oct. 2006, pp. 1476–1481.
- [21] H. Chuangfeng and F. Yuefa, "Robust control for stable dynamic walking of biped robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, oct. 2006, pp. 1750–1754.
- [22] A. Fattah and R. Dehghani, "Stability analysis and robust control of a biped robot with four links and three actuators," *IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS 2008.*, pp. 3420–3425, Sept. 2008.
- [23] S. Moosavian, A. Takhmar, and M. Alghooneh, "Regulated sliding mode control of a biped robot," in *International Conference on Mechatronics and Automation. ICMA 2007.*, aug. 2007, pp. 1547–1552.
- [24] J. P. Cerri, M. H. Terra, and J. Y. Ishihara, "Recursive robust regulator for discrete-time state-space systems," in *Proceedings of the 2009 American Control Conference (ACC)*, St. Louis, Missouri, USA, 2009.

- [25] T. F. P. A. Taveira, “Controladores adaptativos não-lineares com critério h-infinito aplicados a robôs espaciais,” Master’s thesis, Departamento de Engenharia Elétrica da Escola de Engenharia de São Carlos - Universidade de São Paulo, 2006.
- [26] M. W. Spong and G. Bhatia, “Further results on passivity based control of bipedal locomotion,” in *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, USA, 2003.
- [27] J. P. Cerri, “Regulador robusto recursivo para sistemas lineares de tempo discreto no espaço de estado,” Master’s thesis, Departamento de Engenharia Elétrica da Escola de Engenharia de São Carlos - Universidade de São Paulo, 2009.
- [28] R. Johansson, “Quadratic optimization of motion coordination and control,” in *IEEE International Conference on Robotics and Automation, 1990. Proceedings.*, may 1990, pp. 1204–1209 vol.2.
- [29] F. Wu, X. H. Yang, A. Packard, and G. Becker, “Induced $\mathcal{L}_2$ -norm control for lpv systems with bounded parameter variation rates,” *International Journal of Robust and Nonlinear Control*, vol. 6, no. 9/10, pp. 983–998, 1996.
- [30] A. A. G. Siqueira and M. H. Terra, “Control of underactuated manipulators using nonlinear $\mathcal{H}_\infty$ techniques,” in *Proceedings of the 2002 IEEE Conference on Decision and Control (CDC)*, Las Vegas, Nevada, USA, 2002.
- [31] ———, “Nonlinear $\mathcal{H}_\infty$ control via quasi-lpv representation applied in a underactuated manipulator,” in *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002.
- [32] ———, “Nonlinear and markovian $\mathcal{H}_\infty$ controls of underactuated manipulators,” *IEEE Transactions on Control Systems Technology*, vol. 12, no. 6, pp. 811–826, 2004.
- [33] L. L. Whitcomb, A. A. Rizzi, and D. E. Koditschek, “Comparative experiments with a new adaptive controller for robot arms,” *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 59–70, Feb 1993, doi:10.1109/70.210795.
- [34] H. Berghuis, H. Roebbers, and H. Nijmeijer, “Experimental comparison of parameter estimation methods in adaptive robot control,” *Automatica*, vol. 31, no. 9, pp. 1275–1285, Sept 1995, doi:10.1016/0005-1098(95)00046-Y.
- [35] R. S. Inoue, A. A. G. Siqueira, and M. H. Terra, “Experimental results on the nonlinear $\mathcal{H}_\infty$ control via quasi-lpv representation and game theory for wheeled mobile robots,” *Robotica*, vol. 27, no. 4, pp. 547–553, 2009, doi:10.1017/S0263574708004931.
- [36] A. A. G. Siqueira, M. H. Terra, and L. S. A. Tubota, *Climbing and Walking Robots*. Croatia, Vukovar: Intech, 2010, ch. Nonlinear H_∞ Control Applied to Biped Robots.

APÊNDICE A – Simulador

O simulador usado para os testes dos controladores foi desenvolvido no trabalho de [19] em ambiente MatLab, implementando a estratégia de controle descrita em 4.1. Este foi modificado para que simulasse o comportamento dinâmico de um robô bípede sem pés, tal qual apresentado no Capítulo 2, sendo dividido em oito arquivos principais:

1. main.m
2. walk1.m
3. kneelock.m
4. ctraj.m
5. walk1b.m
6. walk2.m
7. interpolation.m
8. anim.m

A.1 main.m

Este arquivo é o responsável por implementar a estratégia de controle da maneira correta, conforme pode ser visto no código a seguir.

```
%MAIN.M
clear all
close all
clc
global xr dxr X1f X2f X3f xknee Cba cont aux aux2 dr2 xdr index2 cont2 a_d pindex d
```

```

... Patual tknee xknee t_passo tau_ext contagem tempo
t0 = 0;
tfinal = 6;
t = t0;
tout = t0;
Patual = eye(8);
contagem=1;
tau_ext = zeros(4,1)
load x0_fixo_1.mat;
x0 = x0_fixo;%*1.5;
pindex=1;
cont=0;
load controle/feasibleX_1.mat
% Tamanho da bacia de atração
Cba = 0.01;
[cont,dr] = baciatract(x0,xr,Cba);
dr2=[];xdr=[];index2=[];cont2=[];
x= x0;
xout=x0;
dx=[];
aux=0;
aux2=0;
aux3=0;
t0_posimpact=0;
desej=0;
t_passo = 0;

for i=1:1 % Numero de passos
    Patual = eye(8);
    % Fase de balanço da perna
    [t1,x1,dx1,ie1] = walk1(t0,x0,'demo');
    nt = length(t1);
    t =[t; t1(2:nt)];
    if i==1
        dx=[dx;dx1];

```

```

else
    dx=[dx;dx1(2:nt,:)];
end
t0 = t1(nt);

% Travamento do joelho
if ie1 == 1
    x0 = kneelock(x1(nt,:));
    x= [x; x1(2:nt-1,:); x0];

else
    x0=x1(nt,:);
    dx0=dx1(nt,:);

    % Caso nao tenha conseguido a condicao
    % theta2=theta3 gera trajetoria desejada para o controle
    if (t0_posimpact+tknee-t0)>0.1
        time_final = t0_posimpact + tknee;%t0 + 0.3;
        xif=[x0(1:4);x0(5:8);dx0(5:8);xknee(1:4);xknee(5:8);dxknee(5:8)];
        [a_d]=ctrhaj(t0,time_final,xif);
        desej=1
    else
        desej=0
    end

    % Fase de balanço com controle
    x= [x; x1(2:nt-1,:); x0];
    [t1b,x1b,dx1b,ie1b] = walk1b(t0,x0,'demo');
    nt = length(t1b);
    t =[t; t1b(2:nt)];
    dx=[dx;dx1b(2:nt,:)];
    t0 = t1b(nt);
    if ie1b == 1
        x0 = kneelock(x1b(nt,:));
        x= [x; x1b(2:nt-1,:); x0];
    end
end

```

```

        else
            x0 = x1b(nt,:);
            x= [x; x1b(2:nt-1,:); x0];
            [t1c,x1c,dx1c,ie1c] = walk1(t0,x0,'demo');
            nt = length(t1c);
            t =[t; t1c(2:nt)];
            dx=[dx;dx1c(2:nt,:)];
            t0 = t1c(nt);
            x0 = kneelock(x1c(nt,:));
            x= [x; x1c(2:nt-1,:); x0];
        end

    end %do if ie1 == 1
    % Segunda fase do passo
    [t2,x2,dx2] = walk2(t0,x0,'demo');
    nt = length(t2);
    x2(nt,:)= interpolation(x2(nt-1,:),x2(nt,:),t2(nt-1),t2(nt));
    x0 = impact_lap_red(x2(nt,:));
    t0 = t2(nt);
    t0_posimpact = t0;
    t =[t; t2(2:nt)];
    x= [x ; x2(2:nt-1,:); x0];
    dx= [dx;dx2(2:nt,:)];
    t_passo=t(length(t));
    xout=[xout;x0];
    desej = 0;
    b_d = a_d;
end
save x0_fixo.mat xout x t
% Calculo dos indices de desempenho
[soma_erro soma_tau] = indice_desempenho(x1b',tau_ext,t1b,tempo)

doplots = [1; % positions
            1; % velocities
            0; % torques

```



```

0; % outputs
0; % foot and hip trajectories
0; % hip specs & knee deflection specs
0]; % stance let forces

doprints = 0;

%% Calculo trajetoria desejada quando esta longe
for k=1:length(t1b)
    [x_d(:,k),dx_d] = vtraj(t1b(k),b_d);
end
for k=1:length(t)
    disturbio(:,k) = perturbation(t(k),pindex);
end

%% Gradicos de fase
figure;plot([x(:,1)], [x(:,5)],xr(:,1),xr(:,5),'r',xr(1,1),xr(1,5),'*',...
...x(1,1),x(1,5),'b*', 'LineWidth',2)
hold on;plot([x_d(1,:)], [x_d(5,:)], 'g', 'LineWidth',2)
xlabel('$\theta_1$ (rad)', 'interpreter', 'latex')
ylabel('$\dot{\theta}_1$ (rad/s)', 'interpreter', 'latex')
figure;plot([x(:,2)], [x(:,6)],xr(:,2),xr(:,6),'r',xr(1,2),xr(1,6),'*',...
...x(1,2),x(1,6),'b*', 'LineWidth',2 )
hold on;plot([x_d(2,:)], [x_d(6,:)], 'g', 'LineWidth',2)
xlabel('$\theta_2$ (rad)', 'interpreter', 'latex')
ylabel('$\dot{\theta}_2$ (rad/s)', 'interpreter', 'latex')
figure;plot([x(:,3)], [x(:,7)],xr(:,3),xr(:,7),'r',xr(1,3),xr(1,7),'*',...
...x(1,3),x(1,7),'b*', 'LineWidth',2 )
hold on;plot([x_d(3,:)], [x_d(7,:)], 'g', 'LineWidth',2)
xlabel('$\theta_3$ (rad)', 'interpreter', 'latex')
ylabel('$\dot{\theta}_3$ (rad/s)', 'interpreter', 'latex')
figure;plot([x(:,4)], [x(:,8)],xr(:,4),xr(:,8),'r',xr(1,4),xr(1,8),'*',...
...x(1,4),x(1,8),'b*', 'LineWidth',2)
hold on;plot([x_d(4,:)], [x_d(8,:)], 'g', 'LineWidth',2)
xlabel('$\theta_T$ (rad)', 'interpreter', 'latex')

```

```

ylabel('$\dot{\theta}_T$ (rad/s)', 'interpreter', 'latex')

%% Graficos de disturbio
figure; plot(t, disturbio(1,:), 'b', 'LineWidth', 2)
hold on; plot(t, disturbio(2,:), 'r', 'LineWidth', 2)
hold on; plot(t, disturbio(3,:), 'g', 'LineWidth', 2)
hold on; plot(t, disturbio(4,:), 'y', 'LineWidth', 2)
xlabel('$Tempo$ (s)', 'interpreter', 'latex')
ylabel('$Disturbio$ (Nm)', 'interpreter', 'latex')

%% Grafico de torque
figure
plot(tempo, tau_ext(1,:), 'b', 'LineWidth', 2)
hold on
plot(tempo, tau_ext(2,:), 'r', 'LineWidth', 2)
hold on
plot(tempo, tau_ext(3,:), 'g', 'LineWidth', 2)
hold on
plot(tempo, tau_ext(4,:), 'y', 'LineWidth', 2)
ylim([-60 40])
xlabel('$Tempo$ (s)', 'interpreter', 'latex')
ylabel('$Torque$ (Nm)', 'interpreter', 'latex')
legend('\theta_1', '\theta_2', '\theta_3', '\theta_T', 'Location', 'North')

%make_plots(t,x,step_t,doplots,doprints,'fig');
%% Animacao do robo andando
anim(t,x,0.02);

```

Como pode ser visto é no arquivo main.m onde são chamados os outros sete arquivos.

A.2 walk1.m

Este arquivo faz a integração do modelo da primeira fase do caminhar do bípede, mostrado no Capítulo 2, utilizando somente o controle do tronco para que exista um ciclo limite. A integração do modelo é feita até que ocorra o travamento do joelho (fim da primeira fase) ou que as posições das juntas fiquem fora da bacia de atração, condição esta que acarreta no término da

integração para que seja iniciada a integração do modelo com o controlador atuando para que o bípede não saia da bacia de atração feita pelo arquivo walk1b.m. O código é mostrado a seguir.

```
function varargout = walk1(t0,x0,flag)
%WALK    Simulate the kneed biped walker.
%Copyright (c) 2003 by Eric Westervelt and Jessie Grizzle.  This code
%may be freely used for noncommercial ends.  If use of this code in
%part or in whole results in publication, proper citation must be
%included in that publication.  This code comes with no guarantees or
%support.
%Eric R. Westervelt & Jessie W. Grizzle
%10/2/00
%11/30/03 - updated for the CDC

global xr Cba cont aux aux2 pindex
if nargin == 2, flag = ' '; end
if nargin == 0, flag = 'demo'; end
switch flag
    case ''
        % Return dx/dt = dynamics(t,x).
        varargout{1} = f(t0,x0);
    case 'events'
        % Return [value,isterminal,direction].
        [varargout{1:3}] = events(t0,x0);
    case 'demo'
        % Run a demo.
        [varargout{1:4}] = demo(t0,x0);
    otherwise
        error(['Unknown flag ' flag '.']);
end

% -----
% This is the system dynamics function

function dx = f(t,x)
global aux2 pindex
dq(1:4) = x(5:8);
[D,C,G,G_alpha,B,E] = model_absolute_spongb_inc(x);
u=[0;0;0;0];
```

```

u_aux=[0;0;0;0];
x4absd = 0*pi/180;
u_aux(4) = 110*(x4absd - x(4)) + 60*(- x(8));
pertu = perturbation(t,pindex);
u_aux=u_aux + pertu;
u = inv(B)*u_aux;
ddq(1:4) = inv(D)*[-C*dq(1:4)'+G_alpha+B*u];
dx = [dq';ddq'];

% -----
function [value,isterminal,direction] = events(t,x)
global xr Cba cont aux2 aux t_passo
if 1
    % Deteccao do travamento do joelho
    value(1) = x(3)-x(2);
    isterminal(1) = 1; % stop when this event occurs
    direction(1) = -1; % direction detection
    % Deteccao se a trajetoria pertence a bacia de atracao
    [cont,dr] = baciatract(x',xr,Cba);
    %value(2) = 1-cont;
    value(2) = 1;
    if t < (0.01 + t_passo)&&cont==0
        value(2) = 1;
    else
        value(2) = Cba-dr;
    end

    isterminal(2) = 1; % stop when this event occurs
    direction(2) =-1; % direction detection
else
    %% noevents, just watch him fall!!
    value=1;
    isterminal=1;
    direction=1;
end

```

```

% -----
function [tout,xout,dxout,ie] = demo(t0,x0)
tfinal = 30;
options = odeset('Events','on','Refine',4,'RelTol',10^-5,'AbsTol',10^-6);
tout = t0;
xout = x0;
teout = [];
xeout = [];
ieout = [];
step_t = [];
modelParameters; % get model parameters
%tspan=linspace(t0,tfinal,30000);
% Solve until the first terminal event.
[t,x,te,xe,ie] = ode45('walk1',[t0 tfinal],x0,options);
nt = length(t);
for k = 1:nt
    dxout(k,:) = f(0,x(k,:))';
end

% Accumulate output. This could be passed out as output arguments.
tout = [tout; t(2:nt)];
xout = [xout; x(2:nt,:)];

```

A.3 kneelock.m

Este arquivo implementa a equação (2.10) que realiza o mapeamento das juntas antes e após o impacto do travamento do joelho. O código do arquivo é mostrado a seguir.

```

function x_plus = kneelock(x)
%KNEELOCK Map pre-knee lock state to post-knee lock state.
% X_PLUS = KNEELOCK(X_MINUS)
%Copyright (c) 2003 by Eric Westervelt and Jessie Grizzle. This code
%may be freely used for noncommercial ends. If use of this code in
%part or in whole results in publication, proper citation must be
%included in that publication. This code comes with no guarantees or

```

```
%support.
%Eric R. Westervelt & Jessy W. Grizzle
%8/2/01
%11/30/03 - updated for the CDC

[D] = model_absolute_spongb_inc(x);
Ji = [0 1 -1 0];
lambda = inv(Ji*inv(D)*Ji')*Ji*(x(5:8)');
x_plus(1:4)=x(1:4);
x_plus(5:8)= x(5:8) - (inv(D)*Ji'*lambda)';
```

A.4 ctraj.m

Esse arquivo implementa o cálculo dos coeficientes de um polinômio do quinto grau através das condições iniciais e finais de tempo, posição e velocidade, que no código mostrado abaixo está na forma matricial, para a geração da trajetória desejada das juntas do robô bípede.

```
function [a] = ctraj(ti,tf,xif)

T=[1  ti  ti^2  ti^3  ti^4  ti^5;
    0  1  2*ti  3*ti^2  4*ti^3  5*ti^4;
    0  0  2  6*ti  12*ti^2  20*ti^3;
    1  tf  tf^2  tf^3  tf^4  tf^5;
    0  1  2*tf  3*tf^2  4*tf^3  5*tf^4;
    0  0  2  6*tf  12*tf^2  20*tf^3];
a=inv(T)*xif;
return;
```

A.5 walk1b.m

O código contido nesse arquivo, e mostrado abaixo, faz a integração da primeira fase do modelo do robô com a ação de um dos controles apresentados no Capítulo 3 para que as trajetórias das juntas não fiquem fora da bacia de atração do ciclo limite, como dito na Seção A.2

```
function varargout = walk1b(t0,x0,flag)
```

```

%WALK    Simulate the kneed biped walker.
%Copyright (c) 2003 by Eric Westervelt and Jessy Grizzle.  This code
%may be freely used for noncommercial ends.  If use of this code in
%part or in whole results in publication, proper citation must be
%included in that publication.  This code comes with no guarantees or
%support.
%Eric R. Westervelt & Jessy W. Grizzle
%10/2/00
%11/30/03 - updated for the CDC

global xr dxr X1f X2f X3f xknee Cba cont aux aux2 a_d pindex desej
if nargin == 2, flag = ' '; end
if nargin == 0, flag = 'demo'; end
switch flag
    case ''
        % Return dx/dt = dynamics(t,x).
        varargout{1} = f(t0,x0);
    case 'events'
        % Return [value,isterminal,direction].
        [varargout{1:3}] = events(t0,x0);
    case 'demo'
        % Run a demo.
        [varargout{1:4}] = demo(t0,x0);
    otherwise
        error(['Unknown flag ' flag ''].');
end

% -----
% This is the system dynamics function

function dx = f(t,x)
global xr dxr X1f X2f X3f xknee Cba cont aux aux2 a_d pindex desej Patual tknee...
... xknee t_passo tau_ext contagem tempo
dq(1:4) = x(5:8);
[D,C,G,G_alpha,B,E] = model_absolute_spongb_inc(x);
u=[0;0;0;0];
u_aux=[0;0;0;0];

```

```

x4absd = 0*pi/180;
u_aux(4) = 110*(x4absd - x(4)) + 60*(- x(8));
B_2 = [eye(4); zeros(4)];
if deseje==1
    if t<tknee + t_passo
        [x_d,dx_d] = vtraj(t,a_d);
    else
        x_d = xknee;
    end
elseif deseje==0
    [x_d,dx_d] = x_desejado(x',xr,dxr);
end
errorqd = -(x_d(5:8)-x(5:8)');
errorq = -(x_d(1:4)-x(1:4)');
erro = [errorq;errorqd];
ddq_d =dx_d(5:8);
dq_d =x_d(5:8);

% Controle Hinf Teoria dos jogos
% SEM INCERTEZA
% soma_erro = 25.1520
% soma_tau = 67.4080
% COM INCERTEZA DE 20%
% soma_erro = 29.4409
% soma_tau = 70.5473
Q1 = 0.7*eye(4);
Q2 = 0.7*eye(4);
Q12 = zeros(4);
R1 = 0.1*eye(1);
gama = 5;
[u_controle] = hinf(Q1,Q2,Q12,R1,gama,x_d,x);

% Controle Robusto Recursivo
% SEM INCERTEZA
% soma_erro = 25.1674

```



```

% soma_tau = 67.4141
% COM INCERTEZA DE 20%
% soma_erro = 29.4539
% soma_tau = 70.5523
% Q = 0.1*eye(8);
% R = 10*eye(4);
% [Pdepois u_controle] = regulador_robusto(Q,R,x',x_d,Patual);
% Patual = Pdepois;
% Patual = Pdepois

% Controle Hinf LMI
% SEM INCERTEZA
% soma_erro = 25.2811
% soma_tau = 67.5197
% COM INCERTEZA DE 20%
% soma_erro = 29.5501
% soma_tau = 70.6255
% f1 = 1;
% f2 = cos(errorq(1));
% f3 = cos(errorq(2));
% X_nlh = (f1*X1f + f2*X2f + f3*X3f);
% u_controle = -(B_2'*inv(X_nlh))*[errorqd'; errorq'];
tau = 1*inv(B)*(D*(dx_d(5:8)' + u_controle) + C*x_d(5:8)' + G_alpha);
tau_ext(:,contagem) = tau;
tempo(contagem) = t;
contagem = contagem+1;
pertu = perturbation(t,pindex);
u_aux = tau + pertu;
u_aux(4) = 50*(x4absd - x(4)) + 60*(- x(8));
u = inv(B)*u_aux;
ddq(1:4) = inv(D)*[-C*dq(1:4)'-G_alpha+B*u];
dx = [dq';ddq'];

% -----
function [value,isterminal,direction] = events(t,x)

```

```

global xr X1f X2f X3f xknee Cba cont aux2 aux
if 1
    value(1) = x(3)-x(2);
    isterminal(1) = 1; % stop when this event occurs
    direction(1) = -1; % direction detection

    value(2) = norm(x' - xknee) -0.01;
    isterminal(2) = 1; % stop when this event occurs
    direction(2) = -1; % direction detection
else
    %% noevents, just watch him fall!!
    value=1;
    isterminal=1;
    direction=1;
end

% -----
function [tout,xout,dxout,ie] = demo(t0,x0)
tfinal = 30;
options = odeset('Events','on','Refine',4,'RelTol',10^-5,'AbsTol',10^-6);
tout = t0;
xout = x0;
teout = [];
xeout = [];
ieout = [];
step_t = [];

modelParameters; % get model parameters
% Solve until the first terminal event.
[t,x,te,xe,ie] = ode45('walk1b',[t0 tfinal],x0,options);
nt = length(t);
for k = 1:nt
    dxout(k,:) = f(0,x(k,:))';
end

```

```
% Accumulate output. This could be passed out as output arguments.
tout = [tout; t(2:nt)];
xout = [xout; x(2:nt,:)];
```

A.6 walk2.m

Este arquivo faz a integração da segunda fase do passo, sendo implementado da mesma maneira que a primeira fase, no entanto a condição de parada da simulação se dá quando o bípede atinge o solo. O código de implementação é amostrado a seguir.

```
function varargout = walk2(t0,x0,flag)
%WALK Simulate the kneed biped walker.
%Copyright (c) 2003 by Eric Westervelt and Jessie Grizzle. This code
%may be freely used for noncommercial ends. If use of this code in
%part or in whole results in publication, proper citation must be
%included in that publication. This code comes with no guarantees or
%support.
%Eric R. Westervelt & Jessie W. Grizzle
%10/2/00
%11/30/03 - updated for the CDC
if nargin == 2, flag = ' '; end
if nargin == 0, flag = 'demo'; end
switch flag
case ''
% Return dx/dt = dynamics(t,x).
varargout{1} = f(t0,x0);
case 'events'
% Return [value,isterminal,direction].
[varargout{1:3}] = events(t0,x0);
case 'demo'
% Run a demo.
[varargout{1:4}] = demo(t0,x0);
otherwise
error(['Unknown flag ' flag '.']);
end

% -----
% This is the system dynamics function
function dx = f(t,x)
```

```

dq(1:4) = x(5:8);
[D,C,G,G_alpha,B,E] = model_absolute_spongb_inc(x);
u=[0;0;0;0];
u_aux=[0;0;0;0];
x4absd = 0*pi/180;
u_aux(4) = 110*(x4absd - x(4)) + 60*(- x(8));
u = inv(B)*u_aux;
Ji = [0 1 -1 0];
lambda = -inv(Ji*inv(D)*Ji')*Ji*inv(D)*(C*dq'+G_alpha-B*u);
ddq = inv(D)*[-C*dq'-G_alpha+B*u - Ji'*lambda];
dx = [dq';ddq];

% -----
function [value,isterminal,direction] = events(t,x)
% Locate the time when critial angle of stance leg minus stance leg angle
% passes through zero in a decreasing direction and stop integration.

if 1
% O uso da funcao fri_point é necessário apenas quando o bipede possui
% pes, caso contrario nao utilizar
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dx = f(t,x);
% yD = fri_point(x,dx);
% modelParameters;
%
% value(2) = L_foot - yD;
% isterminal(2) = 1; % stop when this event occurs
% direction(2) = +1; % decreasing direction detection
% when swing leg foot touches ground
%x(2)=x(2)-pi;
step_len = step_length(x');
value(1) = 1;
if step_len > 0.1;
value(1) = swing_foot_height(x');%-0.001;
else

```

```

    value(1)=-1;
    end
    isterminal(1) = 1; % stop when this event occurs
    direction(1) = -1; % -1 decreasing direction detection
else
    %% noevents, just watch him fall!!
    value=1;
    isterminal=1;
    direction=1;
end

% -----
function [tout,xout,dxout,step_t] = demo(t0,x0)
tfinal = 30;
options = odeset('Events','on','Refine',4,'RelTol',10^-5,'AbsTol',10^-6);
tout = t0;
xout = x0;
teout = [];
xeout = [];
ieout = [];
step_t = [];
modelParameters; % get model parameters

% Solve until the first terminal event.
[t,x,te,xe,ie] = ode45('walk2',[t0 tfinal],x0,options);

nt = length(t);

for k = 1:nt
    dxout(k,:) = f(0,x(k,:))';
end

% Accumulate output. This could be passed out as output arguments.
tout = [tout; t(2:nt)];
xout = [xout; x(2:nt,:)];

```

A.7 interpolation.m

Este arquivo implementa um algoritmo para encontrar o exato instante do impacto do robô com o solo considerando que este instante se deu entre o último e o penúltimo instante mapeados pela integração da segunda fase do passo. O código dessa implementação é mostrado a seguir.

```
% Walking Simulation Example - INTERpolate instant of heelstrike
%      for Matlab 5.2 and up
% This file finds the exact instant that heelstrike occurs, given that it has
% occurred sometime
% between the last and the previous integration sample.
% Created by Martijn Wisse, Delft Bio-robotics Laboratory, August 2003
% Delft University of Technology
function [x,t] = interpolation(x1,x2,t1,t2)
    gy1 = swing_foot_height(x1);
    gy2 = swing_foot_height(x2);
    gyd1 = swing_foot_vel(x1);
    gyd2 = swing_foot_vel(x2);
    dt=t2-t1;
    ddt = zerocross([gy1 gy2], [gyd1 gyd2], dt);

    % now use third order polynome to interpolate q and qd
    ksi = ddt/dt;
    intv= [1-3*ksi^2+2*ksi^3, (ksi-2*ksi^2+ksi^3)*dt, 3*ksi^2-2*ksi^3, (-ksi^2+ksi^3)*dt];
    x(1:4)=intv*[x1(1:4); x1(5:8); x2(1:4); x2(5:8)];
    x(5:8)= x2(5:8);

function x = zerocross(f,fd,dx)
if (f(1)*f(2))>1e-2
error('Zero or more than one boundary crossings')
end
    y = [f(1); f(2)];
    yd = [fd(1); fd(2)];
    DCBA=[0 0 0 1; dx^3 dx^2 dx 1; 0 0 1 0; 3*dx^2 2*dx 1 0]\[y;yd];
    sol=roots(DCBA);
    corr_sol=[(sol>0)&(sol<dx)]';
```

```
x=corr_sol*sol;
```

A.8 anim.m

Este arquivo faz a animação no formato apresentado na Figura A.1, sendo o código mostrado abaixo.

Figura A.1: Formato da animação resultante da simulação do robô bípede.

```
function [te,xe]=anim(t,x,ts)
%ANIM  Animate kneed biped walker.  Note:  (1) TS is an options
%      parameters and (2) T may be a scalar.
%Copyright (c) 2003 by Eric Westervelt and Jessy Grizzle.  This code
%may be freely used for noncommercial ends.  If use of this code in
%part or in whole results in publication, proper citation must be
%included in that publication.  This code comes with no guarantees or
%support.
%Eric R. Westervelt & Jessy W. Grizzle
```

```

%2/15/01
%11/30/03 - updated for the CDC
if nargin < 3, ts = 1/20; end
[n,m] = size(x);
pH_horiz = zeros(n,1);
if n == 1 | m == 1
    te = t;
    xe = x;
    q=xe(1:6);
    n = 1;
else
    % Estimate hip horizontal position by estimating integral of hip
    % velocity
    vH = hip_vel(x); % convert angles to horizontal position of hips
    for j = 2:n
        pH_horiz(j) = pH_horiz(j-1)+(t(j)-t(j-1))*vH(j-1,1);
    end
    [te,pH_horiz] = even_sample(t,pH_horiz,1/ts);
    [te,xe] = even_sample(t,x,1/ts);
    [n,m] = size(xe);
    q = xe(1,1:4);
end
k=0;
out = limb_position(q,pH_horiz(1));
fig_hl=figure;
%set(fig_hl,'Position', [440 700 250 250]); % extra small
set(fig_hl,'Position', [100 100 320 420]);
set(fig_hl,'PaperPosition',[0 0 3 2.999]);
clf
anim_axis=[-.8 .8 -0.1 1.6];
axis off
axis(anim_axis)
axis equal
grid

```

```

% Use actual relations between masses in animation
modelParameters; % get model parameters
scl=0.022; % factor to scale masses
%mr_foot=M_foot^(1/3)*scl;
mr_tib=M_tib^(1/3)*scl; % radius of mass for tib
mr_femur=M_fem^(1/3)*scl; % radius of mass for femurs
mr_torso=M_torso^(1/3)*scl; % radius of mass for torso
leg1_color='g';
leg2_color='r';
torso_color='b';
ground_color='k'; % a.k.a. black

% Approximate circular shape of mass
param=linspace(0,2*pi+2*pi/50,50);
%xmass_foot=mr_foot*cos(param);
%ymass_foot=mr_foot*sin(param);
xmass_tib=mr_tib*cos(param);
ymass_tib=mr_tib*sin(param);
xmass_femur=mr_femur*cos(param);
ymass_femur=mr_femur*sin(param);
xmass_torso=mr_torso*cos(param);
ymass_torso=mr_torso*sin(param);

% Draw ground
buffer=5;
ground=line([-buffer pH_horiz(n)+buffer],[0 0]);
set(ground,'Color',ground_color,'LineWidth',2);
for k=-buffer:floor(pH_horiz(n)+buffer)
    ref_tick(k+buffer+1)=line([k k],[-0.02 0]);
    set(ref_tick(k+buffer+1),'Color',ground_color);
    ref_label(k+buffer+1)=text(-0.03+k,-0.18,num2str(k));
    set(ref_label(k+buffer+1),'FontSize',15);
end

% Draw stance leg

```

```

%foot1=line([out.pToe11 out.pHeel11],[out.pToe12 out.pHeel12]);
%foot1_mass=patch(xmass_foot+out.pFoot11,ymass_foot+out.pFoot12, ...
    %    leg1_color);
femtib=line([out.pFoot11 out.pH1],[out.pFoot12 out.pH2]);
femtib_mass=patch(xmass_femur+out.pFemtib1,ymass_femur+out.pFemtib2, ...
    leg1_color);
%set(foot1_mass,'EdgeColor',leg1_color)
%set(foot1,'LineWidth',2,'Color',leg1_color);
set(femtib_mass,'EdgeColor',leg1_color);
set(femtib,'LineWidth',2,'Color',leg1_color);

% Draw swing leg
%foot2=line([out.pToe21 out.pHeel21],[out.pToe22 out.pHeel22]);
%foot2_mass=patch(xmass_foot+out.pFoot21,ymass_foot+out.pFoot22, ...
    %    leg2_color);
tibia2=line([out.pFoot21 out.pKnee21],[out.pFoot22 out.pKnee22]);
tibia_mass2=patch(xmass_tib+out.pTib21,ymass_tib+out.pTib22, ...
    leg2_color);
femur2=line([out.pKnee21 out.pH1],[out.pKnee22 out.pH2]);
femur_mass2=patch(xmass_femur+out.pFem21,ymass_femur+out.pFem22, ...
    leg2_color);
%set(foot2_mass,'EdgeColor',leg2_color)
%set(foot2,'LineWidth',2,'Color',leg2_color);
set(tibia_mass2,'EdgeColor',leg2_color);
set(tibia2,'LineWidth',2,'Color',leg2_color);
set(femur_mass2,'EdgeColor',leg2_color);
set(femur2,'LineWidth',2,'Color',leg2_color);

% Draw torso
torso=line([out.pH1 out.pT1],[out.pH2 out.pT2]);
torso_mass=patch(xmass_torso+out.pT1,ymass_torso+out.pT2, ...
    torso_color);
set(torso_mass,'EdgeColor',torso_color);
set(torso,'LineWidth',2,'Color',torso_color);
hip_mass=patch(xmass_torso+out.pH1,ymass_torso+out.pH2, ...

```

```

torso_color);
set(hip_mass,'EdgeColor',torso_color);
for k = 1:n
    if n == 1 | m == 1, q=xe(1:6); else, q=xe(k,1:6); end
    out = limb_position(q,pH_horiz(k));
    % stance leg
    % set(foot1,'XData',[out.pToe11 out.pHeel11],...
    %   'YData',[out.pToe12 out.pHeel12]);
    %set(foot1_mass,'XData',xmass_foot+out.pFoot11,...
    %   'YData',ymass_foot+out.pFoot12);
    set(femtib,'XData',[out.pFoot11 out.pH1],...
        'YData',[out.pFoot12 out.pH2]);
    set(femtib_mass,'XData',xmass_femur+out.pFemtib1,...
        'YData',ymass_femur+out.pFemtib2);

    % swing leg
    % set(foot2,'XData',[out.pToe21 out.pHeel21],...
    %   'YData',[out.pToe22 out.pHeel22]);
    % set(foot2_mass,'XData',xmass_foot+out.pFoot21,...
    %   'YData',ymass_foot+out.pFoot22);
    %
    set(tibia2,'XData',[out.pFoot21 out.pKnee21],...
        'YData',[out.pFoot22 out.pKnee22]);
    set(tibia_mass2,'XData',xmass_tib+out.pTib21,...
        'YData',ymass_tib+out.pTib22);
    set(femur2,'XData',[out.pKnee21 out.pH1],...
        'YData',[out.pKnee22 out.pH2]);
    set(femur_mass2,'XData',xmass_femur+out.pFem21,...
        'YData',ymass_femur+out.pFem22);

    % torso
    set(torso,'XData',[out.pH1 out.pT1],...
        'YData',[out.pH2 out.pT2]);
    set(torso_mass,'XData',xmass_torso+out.pT1,...
        'YData',ymass_torso+out.pT2);

```

```

    set(hip_mass,'XData',xmass_torso+out.pH1,...
'YData',ymass_torso+out.pH2);
if n ~= 1 & m ~= 1
    hdl=title(['T_{est} = ',num2str(te(k),'%.2f')]);
    set(hdl,'FontSize',15);
end
new_axis=anim_axis+[out.pH1 out.pH1 0 0];
axis(new_axis);
for j = 1:length(ref_label)
    if (j-buffer-1<new_axis(1)) | (j-buffer-1>new_axis(2))
        set(ref_label(j),'Visible','off')
        set(ref_tick(j),'Visible','off')
    else
        set(ref_label(j),'Visible','on');
        set(ref_tick(j),'Visible','on')
    end
end
drawnow;
end

```

A.9 Arquivos auxiliares

Outros arquivos são chamados no decorrer da simulação, conforme pode ser visto nos códigos exibidos anteriormente. Estes arquivos são de parâmetros e de modelo do robô bípede sem e com incertezas paramétricas (`modelParameters.m`, `model_absolute_spongb.m` e `model_absolute_spongb_inc.m`), além dos arquivos que implementam os controladores descritos no Capítulo 3: `regulador_robusto.m` e `hinf.m` e de cálculos tais como da altura da perna.