

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Rodrigo Rocha dos Santos

**Implementação de uma abordagem de controle
preditivo em um pêndulo de duas rodas em
ambiente de simulação numérica**

Uberlândia, Brasil

2020

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Rodrigo Rocha dos Santos

**Implementação de uma abordagem de controle preditivo
em um pêndulo de duas rodas em ambiente de simulação
numérica**

Projeto de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Mecatrônica da Universidade Federal de Uberlândia, Minas Gerais, como parte dos requisitos para a obtenção do título de **BACHAREL** em **ENGENHARIA MECATRÔNICA**

Orientador: Prof. Dr. Pedro Augusto Queiroz de Assis

Universidade Federal de Uberlândia – UFU

Faculdade de Engenharia Mecânica

Bacharelado em Engenharia Mecatrônica

Uberlândia, Brasil

2020

Rodrigo Rocha dos Santos

Implementação de uma abordagem de controle preditivo em um pêndulo de duas rodas em ambiente de simulação numérica

Projeto de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Mecatrônica da Universidade Federal de Uberlândia, Minas Gerais, como parte dos requisitos para a obtenção do título de **BACHAREL** em **ENGENHARIA MECATRÔNICA**

Trabalho aprovado. Uberlândia, Brasil, 01 de outubro de 2020:

**Prof. Dr. Pedro Augusto Queiroz de
Assis**
Orientador

Prof. Dr. Rogério Sales Gonçalves

**Prof. Me. Felipe Machini Malachias
Marques**

Uberlândia, Brasil
2020

Dedico este trabalho à minha família.

Agradecimentos

Ao Pai eterno pelo dom da vida, por sua graça e misericórdia.

Aos meus pais Ricardo Joaquim e Maria Helena e aos meus irmãos Rafael Rocha e Rogério Rocha por todo o amor e incentivo.

Ao professor Pedro Augusto pela orientação, conhecimentos compartilhados e paciência.

Aos colegas e amigos adquiridos no decorrer dessa jornada.

À Universidade Federal de Uberlândia, em especial à Faculdade de Engenharia Mecânica por ter me proporcionado o curso.

Resumo

O controle preditivo baseado em modelo (MPC – *Model-based Predictive Control*) tem como vantagens o tratamento sistemático de restrições, o cálculo da ação de controle minimizando-se um índice de desempenho e a aplicabilidade em sistemas com entrada e saída única e em sistemas com múltiplas entradas e/ou saídas. Com o intuito de estudar tal estratégia de controle, o presente trabalho tem como objetivo estabilizar em simulação numérica um pêndulo de duas rodas na posição vertical empregando MPC. Para isso, em um primeiro momento, investigou-se uma formulação de MPC com garantia de factibilidade recursiva. Isto é, com garantia de que o problema de otimização terá solução em todos os instantes de amostragem, caso a condição inicial seja factível. Essa formulação foi implementada com sucesso no controle de um duplo integrador. Contudo, no caso do pêndulo, os resultados não foram satisfatórios. Mais especificamente, notou-se que o estado relacionado com a posição angular da roda do veículo divergia com o passar do tempo. Para resolver o problema, modificou-se a função de custo para obtenção de garantia de convergência, onde considerou-se apenas a minimização do esforço de controle preditivo. Em outras palavras, de modo a se garantir que a ação de controle leve o sistema para o ponto de equilíbrio. Essa última formulação apresentou bons resultados no controle do pêndulo invertido, inclusive em simulações no modelo não linear com ruídos de medida e descasamentos de modelo. Isso reforça a importância do emprego de técnicas de MPC com garantias teóricas de factibilidade recursiva e convergência, o que pode ser interpretado como garantia de estabilidade.

Palavras-chave: Controle preditivo baseado em modelo; pêndulo de duas rodas; garantia de estabilidade.

Abstract

The main advantages of the Model-based Predictive Control (MPC) strategy are constraint handling, computation of the control action by minimizing a performance index and applicability on SISO (Single-Input and Single-Output) and on MIMO (Multiple-Input and Multiple-Output) systems. In order to study such control strategy, this work involves the stabilization of a two-wheel pendulum in the vertical position using MPC. For this purpose, at first, a MPC formulation with recursive feasibility guarantee was investigated. Such guarantee ensures that, if the optimization problem is initially feasible, then it will be feasible at all the following sampling instants. This formulation was successfully implemented on a double integrator. However, the simulation results on the inverted pendulum were not satisfactory. More specifically, it was noted that angular position of the wheel diverged. The problem was solved by modifying (as proposed in the literature) the cost function to obtain convergence guarantee, where only the minimization of the predictive control effort was considered. In other words, to ensure that the control action enforces the system state to the equilibrium point. This last formulation presented good results on controlling the inverted pendulum, including in simulations on the non-linear model with measurement noise and model mismatches. This reinforces the importance of applying MPC techniques with nominal guarantees of recursive feasibility and convergence, which can be interpreted as guarantee of stability.

Key-words: Model-based predictive control; two-wheel pendulum; stability guarantee.

Lista de ilustrações

Figura 1.1 – Diagrama de blocos de um sistema em MF.	12
Figura 1.2 – Veículos do tipo pêndulo invertido: (a) Robô de duas rodas NXTway-G (YAMAMOTO, 2008), (b) Imagem veículo de duas rodas (SILVA, 2017)	14
Figura 2.1 – Diagrama de blocos de uma estrutura de controle preditivo com malha interna (AFONSO; GALVÃO, 2011). (Adaptada pelo autor).	15
Figura 3.1 – Resultados obtidos para a condição inicial $x(0) = [1 \ -1]^T$: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.	26
Figura 3.2 – Ações de controle para condição inicial $x(0) = [1 \ -1]^T$	26
Figura 3.3 – Resultados obtidos para a condição inicial $x(0) = [1,5 \ -0,5]^T$: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.	27
Figura 3.4 – Ações de controle para condição inicial $x(0) = [1,5 \ -0,5]^T$	27
Figura 3.5 – Resultados obtidos para a condição inicial $x(0) = [1,5 \ -0,5]^T$ sem a ação do MPC: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.	28
Figura 3.6 – Ações de controle para condição inicial $x(0) = [1,5 \ -0,5]^T$ sem a ação do MPC.	28
Figura 4.1 – Diagrama esquemático do pêndulo de duas rodas, em que θ e ψ são respectivamente a posição angular das rodas e a inclinação do centro de massa (YAMAMOTO, 2008) (Adaptada pelo autor).	29
Figura 4.2 – Resultados obtidos para a condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.	34
Figura 4.3 – Ações de controle para condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$	35
Figura 4.4 – Resultados obtidos para a condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.	36
Figura 4.5 – Ações de controle para condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$	36
Figura 4.6 – Resultados obtidos para a condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.	37
Figura 4.7 – Ações de controle para condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$	37
Figura 4.8 – Resultados obtidos para a condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$ para o modelo não linear do pêndulo: (a) Evolução dos estados, (b) Inclinação do pêndulo.	38
Figura 4.9 – Ações de controle para condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$ para o modelo não linear do pêndulo.	39

Lista de tabelas

Tabela 4.1 – Parâmetros do pêndulo invertido (YAMAMOTO, 2008).	31
--	----

Lista de abreviaturas e siglas

DC	Direct Current
DMC	Dynamic Matrix Control
LP	Linear Programming
LQR	Linear Quadratic Regulator
LTI System	Linear Time-Invariant System
MAS	Maximal Admissible Set
MA	Malha Aberta
MF	Malha Fechada
MIMO	Multiple-Input Multiple-Output
MPC	Model-based Predictive Control
MPT	Multi-Parametric Toolbox
MPHC	Model Predictive Heuristic Control
PID	Proportional Integral Derivative
PWM	Pulse Width Modulation
SISO	Single-Input Single-Output
ZOH	Zero Order Hold

Lista de símbolos

A	Matriz dos estados
B	Matriz das entradas
C	Matriz das saídas
D	Matriz de transmissão direta
k	Índice de tempo discreto
K	Vetor de ganhos do LQR
$\mathbb{N}$	Conjunto dos números naturais
N	Horizonte de predição
$\mathcal{O}_\infty$	Conjunto máximo admissível
$\mathbb{R}$	Conjunto dos números reais
Θ	Região de operação do controlador
T	Período de amostragem
$\mathcal{U}$	Conjunto admissível para variáveis de controle
u	Ação de controle
u_{lqr}	Ação de controle do LQR
u_{mpc}	Ação do controle preditivo
$\mathcal{X}$	Conjunto admissível para variáveis de estados
x	Vetor de estados
y	Vetor de saídas
$\hat{\bullet}$	Variável de predição. $\bullet$ representa uma variável qualquer
$\bullet_{max}$	Limitante superior para $\bullet$
$\bullet_{min}$	Limitante inferior para $\bullet$

Sumário

1	INTRODUÇÃO	12
2	FORMULAÇÃO DE CONTROLE PREDITIVO ADOTADA	15
2.1	Equação de predição de estados	17
2.2	Equação de predição para o controle total	18
2.3	Problema de programação quadrática	19
3	ESTUDO DE CASO NO DUPLO INTEGRADOR	25
4	ESTUDO DE CASO NO PÊNDULO INVERTIDO DE DUAS RODAS	29
5	CONCLUSÕES E TRABALHOS FUTUROS	40
	REFERÊNCIAS	41
	APÊNDICE A – CARACTERIZAÇÃO DO CONJUNTO DE ATRA- ÇÃO	43
	APÊNDICE B – LINEARIZAÇÃO DA DINÂMICA DO PÊNDULO INVERTIDO	44
	APÊNDICE C – CÓDIGOS DAS SIMULAÇÕES	47
C.1	Matrizes do MPC	47
C.2	MAS	49
C.3	Duplo integrador	50
C.4	Modelo linear do pêndulo	54
C.5	Modelo não linear do pêndulo	60
C.6	Linearização da dinâmica do pêndulo	66

1 Introdução

Sistemas de controle são formados por um conjunto de subsistemas e processos construídos para obter uma saída desejada de acordo com uma entrada específica e um comportamento pré-determinado (NISE, 2011). Em uma estrutura de controle em malha fechada (MF), ilustrada na Figura 1.1, a saída e a referência são comparadas, resultando em um erro de rastreamento. Então, a entrada do processo é calculada por meio de uma lei de controle (ou controlador). A adoção de um sistema em MF possibilita a estabilização de sistemas instáveis em malha aberta (MA), rejeição de perturbações e diminuição dos efeitos de descasamento de modelo (FRANKLIN; POWELL; EMAMI-NAEINI, 2009). Para isso, a lei de controle deve ser projetada adequadamente de acordo com os requisitos de desempenho e limitações do sistema.

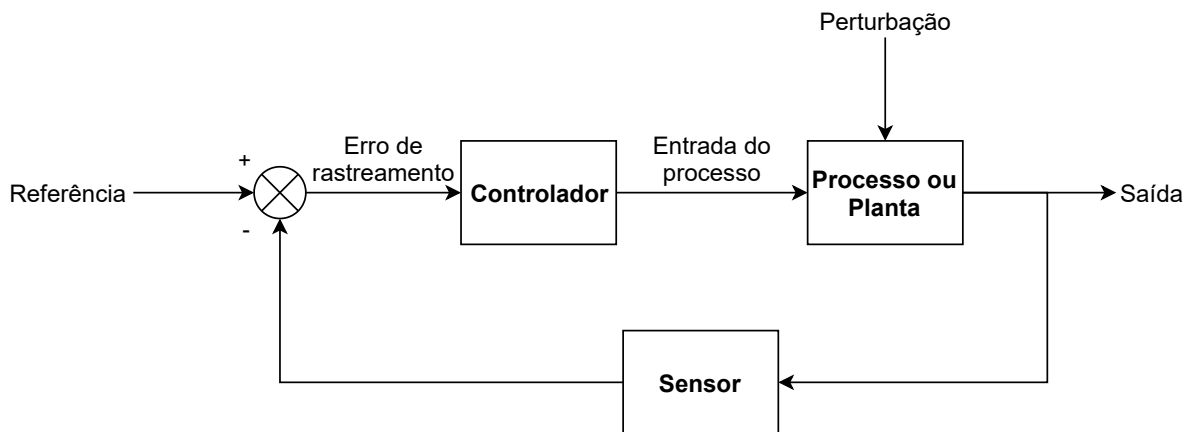


Figura 1.1 – Diagrama de blocos de um sistema em MF.

Uma possibilidade consiste em realizar o projeto utilizando técnicas clássicas (e.g. resposta em frequência, lugar geométrico das raízes) (NISE, 2011). Essas técnicas foram e são empregadas com sucesso no controle de sistemas com entrada e saída única (SISO – *Single-Input Single-Output*). Contudo, comumente processos reais têm múltiplas entradas e/ou saídas (MIMO – *Multiple-Input Multiple-Output*) acopladas entre si, o que pode dificultar o emprego de técnicas clássicas de controle (ROSSITER, 2004). Mais ainda, nessas técnicas não se consideram de maneira sistemática as restrições físicas do processo, isto é, os limitantes sobre as variáveis de entrada e saída. Como resultado, é possível que as ações de controle resultem em violações de tais limitantes, podendo causar danos aos atuadores e/ou colocando em risco a segurança da operação (MACIEJOWSKI, 2002). Assim, para o tratamento sistemático de restrições, pode-se adotar uma estratégia de controle preditivo baseado em modelo (MPC – *Model-based Predictive Control*). Nessa estratégia, a ação de controle é obtida resolvendo-se um problema de otimização. Mais especificamente, minimiza-se uma função de custo que envolve o comportamento futuro (predito)

do processo ao longo de um horizonte de predição. As restrições podem ser incluídas na formulação do problema de otimização, ou seja, são consideradas no projeto da lei de controle. O procedimento de minimização é repetido em cada instante de amostragem a partir das novas informações dos sensores, seguindo a estratégia de horizonte retrocedente. Desse modo, tem-se uma operação em MF (MACIEJOWSKI, 2002).

Os primeiros trabalhos de MPC datam da década de 80. Richalet et al. (1978) propuseram um controle preditivo heurístico (MPHC – *Model Predictive Heuristic Control*) para ser aplicado em problemas nos quais era inviável o emprego de um controlador PID. Cutler e Ramaker (1980) aplicaram uma estratégia de controle preditivo em uma planta sujeita à restrições, na qual o sinal de controle era calculado repetidamente resolvendo-se um problema de programação linear (LP – *Linear Programming*). A estratégia de Cutler e Ramaker (1980) ficou conhecida como DMC (*Dynamic Matrix Control*). Essas técnicas compartilhavam características similares como um modelo interno explícito, horizonte retrocedente e cálculo das ações de controle otimizando o comportamento predito da planta (MACIEJOWSKI, 2002). Com a popularização da abordagem, o MPC foi implementado com sucesso no controle de diferentes processos reais (QIN; BADGWELL, 2003). Pode-se afirmar que, dentre as técnicas de controle mais avançadas, o MPC é a que teve um impacto mais significativo na indústria (MACIEJOWSKI, 2002).

Com o intuito de estudar controle preditivo, o presente trabalho tem como objetivo implementar uma abordagem de MPC para estabilizar a inclinação de um pêndulo invertido de duas rodas. Para isso, uma técnica de MPC será projetada e o desempenho do sistema será avaliado por meio de simulações utilizando o modelo não linear.

Cabe salientar que diferentes trabalhos envolvendo o controle desse tipo de pêndulo podem ser encontrados na literatura. Isso se deve ao fato de que tal sistema é multivariável e instável em malha aberta (FRANKLIN; POWELL; EMAMI-NAEINI, 2009). Por exemplo, em Yamamoto (2008), utiliza-se uma técnica de realimentação de estados para fazer o controle do pêndulo invertido mostrado na Figura 1.2a. Já em Pereira, Sanches e Tavares (2017), adotou-se um controlador Neuro-Fuzzy para estabilização do pêndulo invertido mostrado na Figura 1.2b. Em Canale e Brunet (2013), implementou-se um MPC no controle do pêndulo para o tratamento de restrições de entrada e inclinação. Em particular, adotou-se um pêndulo similar ao mostrado na Figura 1.2a.

Assim, o presente trabalho visa reproduzir em simulação os resultados alcançados em Canale e Brunet (2013). Uma modificação será realizada na formulação do problema de otimização visando aumentar a região de operação do controlador, isto é, o conjunto dos estados no qual se garante que o pêndulo será estabilizado com o devido respeito às restrições (ROSSITER, 2004).

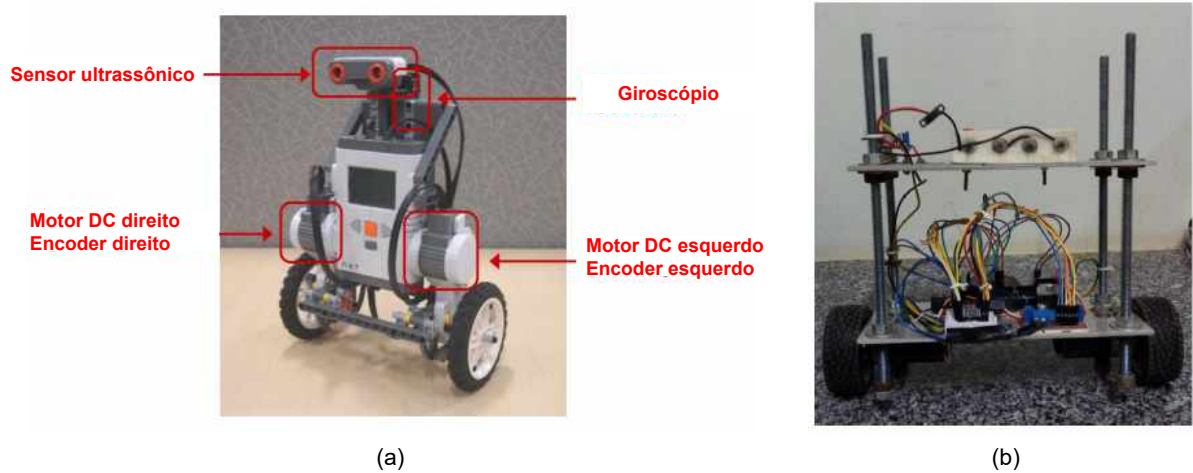


Figura 1.2 – Veículos do tipo pêndulo invertido: (a) Robô de duas rodas NXTway-G (YAMAMOTO, 2008), (b) Imagem veículo de duas rodas (SILVA, 2017)

O restante do trabalho está organizado como descrito a seguir. Na Seção 2 é apresentada a formulação de controle preditivo adotada. Na Seção 3 é realizado um estudo de caso de MPC no controle de um duplo integrador. A Seção 4 envolve os resultados obtidos no pêndulo. Finalmente, na Seção 5, são realizadas as conclusões e propostas para próximos trabalhos.

2 Formulação de controle preditivo adotada

No controle preditivo determina-se uma sequência de ações de controle resolvendo-se um problema de otimização que considera o comportamento futuro e as restrições do processo. A Figura 2.1 mostra um diagrama de blocos de uma planta controlada por uma combinação de MPC e realimentação de estados. O modelo de predição é empregado para calcular o comportamento do processo ao longo de um horizonte N . Isto é, os N próximos estados, denotados por $\hat{x}(k+i|k)$, $i = 1, 2, \dots, N$, de acordo com uma sequência de ações de controle futuras $\hat{u}(k+i-1|k)$, $i = 1, 2, \dots, N$. Em particular, essa última sequência é calculada de modo a minimizar uma função de custo $J(k)$ com respeito às restrições. Como resultado, obtém-se uma sequência de controle ótima $\hat{u}^*(k+i-1|k)$, $i = 1, 2, \dots, N$. Apenas o primeiro termo dessa sequência, $\hat{u}^*(k|k)$, é aplicado ao processo. No próximo instante de amostragem, o procedimento de otimização é repetido. Essa estratégia é conhecida como horizonte retrocedente e introduz realimentação no sistema (MACIEJOWSKI, 2002).

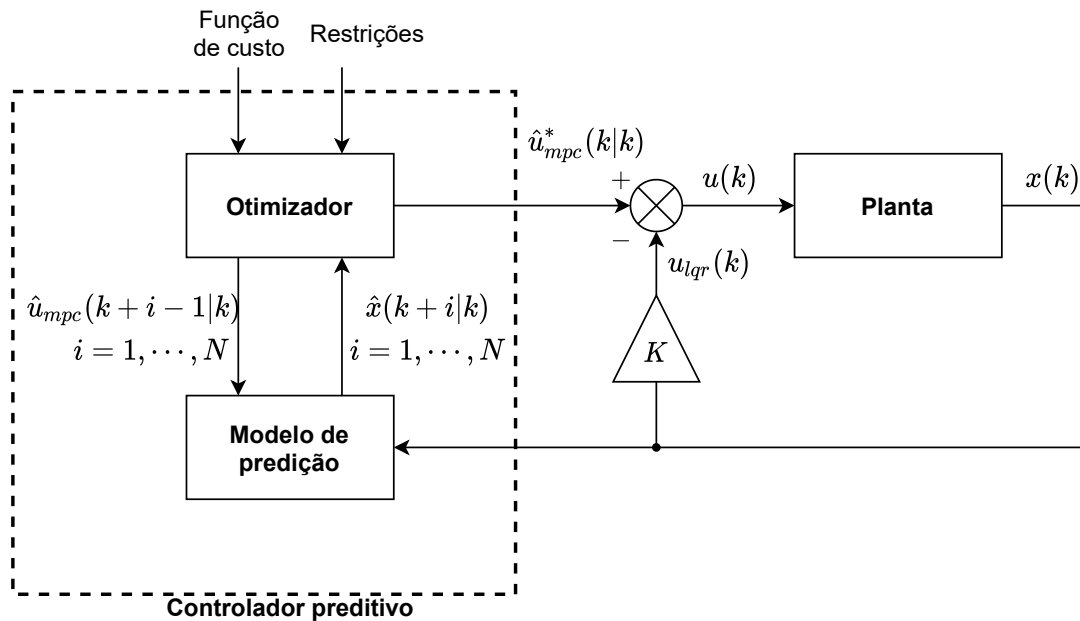


Figura 2.1 – Diagrama de blocos de uma estrutura de controle preditivo com malha interna (AFONSO; GALVÃO, 2011). (Adaptada pelo autor).

A inclusão da realimentação de estados na malha de controle traz algumas características interessantes para o problema, como a garantia de estabilidade e bom desempenho. Isso ocorre pois impõe-se que ao final do horizonte de predição, os estados adentrem em uma região admissível e invariante sob ação exclusiva da realimentação de estados (ROSSITER, 2004). Neste trabalho, supõe-se que a malha interna seja um regulador linear quadrático (LQR – *Linear Quadratic Regulator*).

Para projetar o controlador, considere um sistema discreto, linear e invariante no tempo representado por

$$x(k+1) = Ax(k) + Bu(k) \quad (2.1)$$

$$y(k) = Cx(k) \quad (2.2)$$

sendo

$x \in \mathbb{R}^n$ o vetor de estados

$y \in \mathbb{R}^m$ o vetor de saídas

$u \in \mathbb{R}^r$ o vetor de entradas

$A \in \mathbb{R}^{n \times n}$ a matriz dos estados

$B \in \mathbb{R}^{n \times r}$ a matriz das entradas

$C \in \mathbb{R}^{m \times n}$ a matriz das saídas

Observando a Figura 2.1, verifica-se que a entrada da planta é

$$u(k) = u_{lqr}(k) + u_{mpc}(k) \quad (2.3)$$

em que

$$u_{lqr}(k) = -Kx(k) \quad (2.4)$$

sendo K calculado a partir da solução da equação de Riccati (OGATA, 1994) e o termo u_{mpc} é obtido minimizando-se a seguinte função de custo:

$$J(k) = \sum_{i=1}^N \hat{x}^T(k+i|k)Q\hat{x}(k+i|k) + \hat{u}^T(k+i-1|k)R\hat{u}(k+i-1|k) \quad (2.5)$$

sendo $Q \in \mathbb{R}^{n \times n}$ e $R \in \mathbb{R}^{r \times r}$ matrizes de ajuste.

Mais precisamente, $\hat{u}_{mpc}$ é determinado resolvendo-se o problema de otimização a seguir

$$\min_{\hat{u}_{mpc}(k+i-1|k), i=1, \dots, N} J(k) \quad (2.6)$$

sujeito a

$$\hat{x}(k+i|k) \in \mathcal{X}, i = 1, 2, \dots \quad (2.7)$$

$$\hat{u}(k+i-1|k) \in \mathcal{U}, i = 1, 2, \dots \quad (2.8)$$

$$\hat{x}(k+N|k) \in \mathcal{O}_\infty \quad (2.9)$$

em que $\mathcal{X}$ e $\mathcal{U}$ representam os conjuntos admissíveis para as variáveis de estado e controle, respectivamente. Tais conjuntos são formados por limitantes inferiores e superiores para estado e controle denotados por x_{min} , u_{min} , x_{max} e u_{max} , respectivamente. Já $\mathcal{O}_\infty$ é um conjunto admissível e invariante (MAS – *Maximal Admissible Set*) para a dinâmica de

malha fechada com o LQR. A restrição terminal (2.9) é incluída no problema de otimização para garantia de factibilidade recursiva (ROSSITER, 2004).

O problema de otimização (2.6) – (2.8) pode ser transformado em um problema de programação quadrática. Para isso, primeiramente deve-se obter equações de predição de estados e controle em termos das ações de controle preditivo. Isso é realizado nas próximas seções.

2.1 Equação de predição de estados

Substituindo (2.4) em (2.3) e posteriormente em (2.1), segue que

$$\begin{aligned}
 x(k+1) &= Ax(k) + B(-Kx(k) + u_{mpc}(k)) \\
 &= Ax(k) - BKx(k) + Bu_{mpc}(k) \\
 &= \underbrace{(A - BK)}_{\Phi} x(k) + Bu_{mpc}(k) \\
 x(k+1) &= \Phi x(k) + Bu_{mpc}(k)
 \end{aligned} \tag{2.10}$$

Essa relação pode ser utilizada para escrever a predição do estado um passo à frente. Mais precisamente, tem-se que

$$\hat{x}(k+1|k) = \Phi \hat{x}(k|k) + B\hat{u}_{mpc}(k|k) \tag{2.11}$$

o símbolo “ $\hat{}$ ” aponta variáveis/vetores preditos e a notação $(k+1|k)$ indica que a predição um passo à frente está sendo realizada com base nas informações disponíveis em k . Supondo-se que o estado atual da planta é conhecido, tem-se que $\hat{x}(k|k) = x(k)$. Logo,

$$\hat{x}(k+1|k) = \Phi x(k) + B\hat{u}_{mpc}(k|k) \tag{2.12}$$

Seguindo um procedimento similar para dois passos à frente, é possível escrever

$$\begin{aligned}
 \hat{x}(k+2|k) &= \Phi \hat{x}(k+1|k) + B\hat{u}_{mpc}(k+1|k) \\
 &= \Phi [\Phi x(k) + B\hat{u}_{mpc}(k|k)] + B\hat{u}_{mpc}(k+1|k) \\
 \hat{x}(k+2|k) &= \Phi^2 x(k) + \Phi B\hat{u}_{mpc}(k|k) + B\hat{u}_{mpc}(k+1|k)
 \end{aligned} \tag{2.13}$$

Já para três passos à frente, tem-se

$$\begin{aligned}
 \hat{x}(k+3|k) &= \Phi \hat{x}(k+2|k) + B\hat{u}_{mpc}(k+2|k) \\
 &= \Phi [\Phi^2 x(k) + \Phi B\hat{u}_{mpc}(k|k) + B\hat{u}_{mpc}(k+1|k)] + B\hat{u}_{mpc}(k+2|k) \\
 \hat{x}(k+3|k) &= \Phi^3 x(k) + \Phi^2 B\hat{u}_{mpc}(k|k) + \Phi B\hat{u}_{mpc}(k+1|k) + B\hat{u}_{mpc}(k+2|k)
 \end{aligned} \tag{2.14}$$

Por fim, a predição para N é dada por

$$\hat{x}(k+N|k) = \Phi^N x(k) + \Phi^{N-1} B \hat{u}_{mpc}(k|k) + \Phi^{N-2} B \hat{u}_{mpc}(k+1|k) + \dots + B \hat{u}_{mpc}(k+N-1|k) \quad (2.15)$$

Organizando (2.12) – (2.15) na forma matricial, obtém-se

$$\underbrace{\begin{bmatrix} \hat{x}(k+1|k) \\ \hat{x}(k+2|k) \\ \hat{x}(k+3|k) \\ \vdots \\ \hat{x}(k+N|k) \end{bmatrix}}_{\hat{X}} = \underbrace{\begin{bmatrix} \Phi \\ \Phi^2 \\ \Phi^3 \\ \vdots \\ \Phi^N \end{bmatrix}}_{P_x} x(k) + \underbrace{\begin{bmatrix} B & 0 & 0 & \dots & 0 \\ \Phi B & B & 0 & \dots & 0 \\ \Phi^2 B & \Phi B & B & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Phi^{N-1} B & \Phi^{N-2} B & \Phi^{N-3} B & \dots & B \end{bmatrix}}_{H_x} \underbrace{\begin{bmatrix} \hat{u}_{mpc}(k|k) \\ \hat{u}_{mpc}(k+1|k) \\ \hat{u}_{mpc}(k+2|k) \\ \vdots \\ \hat{u}_{mpc}(k+N-1|k) \end{bmatrix}}_{\hat{U}_{mpc}} \quad (2.16)$$

Resumidamente,

$$\hat{X} = P_x x(k) + H_x \hat{U}_{mpc} \quad (2.17)$$

2.2 Equação de predição para o controle total

A equação para o cálculo da ação de controle do LQR foi dada em (2.4). Por essa expressão, a predição um passo à frente é dada por

$$\hat{u}_{lqr}(k|k) = -Kx(k) \quad (2.18)$$

Para dois passos à frente, tem-se

$$\hat{u}_{lqr}(k+1|k) = -K\hat{x}(k+1|k) \quad (2.19)$$

Similarmente para três passos à frente, segue que

$$\hat{u}_{lqr}(k+2|k) = -K\hat{x}(k+2|k) \quad (2.20)$$

Finalmente, N passos é dada por

$$\hat{u}_{lqr}(k+N-1|k) = -K\hat{x}(k+N-1|k) \quad (2.21)$$

Organizando (2.18) – (2.21) na forma matricial, obtém-se

$$\underbrace{\begin{bmatrix} \hat{u}_{lqr}(k|k) \\ \hat{u}_{lqr}(k+1|k) \\ \hat{u}_{lqr}(k+2|k) \\ \vdots \\ \hat{u}_{lqr}(k+N-1|k) \end{bmatrix}}_{\hat{U}_{lqr}} = \underbrace{\begin{bmatrix} -K \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{P_{lqr}} x(k) + \underbrace{\begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ -K & 0 & \dots & 0 & 0 \\ 0 & -K & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -K & 0 \end{bmatrix}}_{H_{lqr}} \underbrace{\begin{bmatrix} \hat{x}(k+1|k) \\ \hat{x}(k+2|k) \\ \hat{x}(k+3|k) \\ \vdots \\ \hat{x}(k+N|k) \end{bmatrix}}_{\hat{X}} \quad (2.22)$$

Em um formato mais conciso,

$$\hat{U}_{lqr} = P_{lqr}x(k) + H_{lqr}\hat{X} \quad (2.23)$$

Então, de (2.17) e (2.23) é possível escrever

$$\begin{aligned} \hat{U}_{lqr} &= P_{lqr}x(k) + H_{lqr} [P_x x(k) + H_x \hat{U}_{mpc}] \\ \hat{U}_{lqr} &= (P_{lqr} + H_{lqr}P_x)x(k) + H_{lqr}H_x \hat{U}_{mpc} \end{aligned} \quad (2.24)$$

Como mostrado em (2.3), o controle total aplicado à planta é o resultado das ações do LQR e MPC. Desse modo, a ação de controle total N passos à frente é dada por

$$\underbrace{\begin{bmatrix} \hat{u}(k|k) \\ \hat{u}(k+1|k) \\ \hat{u}(k+2|k) \\ \vdots \\ \hat{u}(k+N-1|k) \end{bmatrix}}_{\hat{U}} = \underbrace{\begin{bmatrix} \hat{u}_{lqr}(k|k) \\ \hat{u}_{lqr}(k+1|k) \\ \hat{u}_{lqr}(k+2|k) \\ \vdots \\ \hat{u}_{lqr}(k+N-1|k) \end{bmatrix}}_{\hat{U}_{lqr}} + \underbrace{\begin{bmatrix} \hat{u}_{mpc}(k|k) \\ \hat{u}_{mpc}(k+1|k) \\ \hat{u}_{mpc}(k+2|k) \\ \vdots \\ \hat{u}_{mpc}(k+N-1|k) \end{bmatrix}}_{\hat{U}_{mpc}} \quad (2.25)$$

Resumidamente,

$$\hat{U} = \hat{U}_{lqr} + \hat{U}_{mpc} \quad (2.26)$$

Substituindo (2.24) em (2.26), pode-se escrever

$$\begin{aligned} \hat{U} &= [(P_{lqr} + H_{lqr}P_x)x(k) + H_{lqr}H_x \hat{U}_{mpc}] + \hat{U}_{mpc} \\ &= \underbrace{(P_{lqr} + H_{lqr}P_x)x(k)}_{P_u} + \underbrace{(H_{lqr}H_x + I)\hat{U}_{mpc}}_{H_u} \\ \hat{U} &= P_u x(k) + H_u \hat{U}_{mpc} \end{aligned} \quad (2.27)$$

sendo $I \in \mathbb{R}^{rN \times rN}$ a matriz identidade.

Na próxima seção, reescreve-se o problema de otimização (2.6) – (2.8) na forma de programação quadrática utilizando (2.17) e (2.27).

2.3 Problema de programação quadrática

A função de custo definida em (2.5), pode ser escrita em termos de $\hat{X}$ e $\hat{U}$ por

$$J(k) = \hat{X}^T \bar{Q} \hat{X} + \hat{U}^T \bar{R} \hat{U} \quad (2.28)$$

sendo

$$\bar{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & Q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q \end{bmatrix} \quad (2.29)$$

$$\bar{R} = \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R \end{bmatrix} \quad (2.30)$$

Então, substituindo (2.17) e (2.27) em (2.28), obtém-se

$$\begin{aligned} J(k) &= \left(P_x x(k) + H_x \hat{U}_{mpc} \right)^T \bar{Q} \left(P_x x(k) + H_x \hat{U}_{mpc} \right) + \\ &\quad \left(P_u x(k) + H_u \hat{U}_{mpc} \right)^T \bar{R} \left(P_u x(k) + H_u \hat{U}_{mpc} \right) \\ &= x^T(k) P_x^T \bar{Q} P_x x(k) + x^T(k) P_x^T \bar{Q} H_x \hat{U}_{mpc} + \hat{U}_{mpc}^T H_x^T \bar{Q} P_x x(k) + \\ &\quad \hat{U}_{mpc}^T H_x^T \bar{Q} H_x \hat{U}_{mpc} + x^T(k) P_u^T \bar{R} P_u x(k) + x^T(k) P_u^T \bar{R} H_u \hat{U}_{mpc} + \\ &\quad \hat{U}_{mpc}^T H_u^T \bar{R} P_u x(k) + \hat{U}_{mpc}^T H_u^T \bar{R} H_u \hat{U}_{mpc} \end{aligned}$$

rearranjando os termos, tem-se

$$\begin{aligned} J(k) &= \hat{U}_{mpc}^T H_x^T \bar{Q} H_x \hat{U}_{mpc} + \hat{U}_{mpc}^T H_u^T \bar{R} H_u \hat{U}_{mpc} + \\ &\quad x^T(k) P_x^T \bar{Q} H_x \hat{U}_{mpc} + \hat{U}_{mpc}^T H_x^T \bar{Q} P_x x(k) + x^T(k) P_u^T \bar{R} H_u \hat{U}_{mpc} + \\ &\quad \hat{U}_{mpc}^T H_u^T \bar{R} P_u x(k) + x^T(k) P_x^T \bar{Q} P_x x(k) + x^T(k) P_u^T \bar{R} P_u x(k) \\ &= \hat{U}_{mpc}^T \left(H_x^T \bar{Q} H_x + H_u^T \bar{R} H_u \right) \hat{U}_{mpc} + \\ &\quad 2x^T(k) P_x^T \bar{Q} H_x \hat{U}_{mpc} + 2x^T(k) P_u^T \bar{R} H_u \hat{U}_{mpc} + \\ &\quad cte \\ &= \hat{U}_{mpc}^T \underbrace{\left(H_x^T \bar{Q} H_x + H_u^T \bar{R} H_u \right)}_{H_{qp}} \hat{U}_{mpc} + 2x^T(k) \underbrace{\left(P_x^T \bar{Q} H_x + P_u^T \bar{R} H_u \right)}_{f_{qp}} \hat{U}_{mpc} + cte \\ J(k) &= \hat{U}_{mpc}^T H_{qp} \hat{U}_{mpc} + 2x^T(k) f_{qp} \hat{U}_{mpc} + cte \end{aligned} \quad (2.31)$$

sendo *cte* termos que não dependem de $\hat{U}_{mpc}$ e, portanto, não alteram a solução do problema de otimização (ROSSITER, 2004).

De maneira análoga, as restrições do problema de otimização também podem ser escritas em termos do vetor de variáveis de decisão $\hat{U}_{mpc}$. Esse procedimento é detalhado a seguir.

- Restrições de estados

A restrição (2.7) pode ser escrita na seguinte forma:

$$x_{min} \leq \hat{x}(k+1|k) \leq x_{max} \quad (2.32)$$

$$x_{min} \leq \hat{x}(k+2|k) \leq x_{max} \quad (2.33)$$

$$\begin{aligned} & \vdots \\ x_{min} & \leq \hat{x}(k+N|k) \leq x_{max} \end{aligned} \quad (2.34)$$

Na forma matricial, tem-se que

$$\underbrace{\begin{bmatrix} x_{min} \\ x_{min} \\ \vdots \\ x_{min} \end{bmatrix}}_{X_{min}} \leq \underbrace{\begin{bmatrix} \hat{x}(k+1|k) \\ \hat{x}(k+2|k) \\ \vdots \\ \hat{x}(k+N|k) \end{bmatrix}}_{\hat{X}} \leq \underbrace{\begin{bmatrix} x_{max} \\ x_{max} \\ \vdots \\ x_{max} \end{bmatrix}}_{X_{max}} \quad (2.35)$$

$$X_{min} \leq \hat{X} \leq X_{max} \quad (2.36)$$

Substituindo (2.17) em (2.36), obtém-se

$$\begin{aligned} X_{min} & \leq P_x x(k) + H_x \hat{U}_{mpc} \leq X_{max} \\ X_{min} - P_x x(k) & \leq H_x \hat{U}_{mpc} \leq X_{max} - P_x x(k) \end{aligned} \quad (2.37)$$

Esses limitantes sobre as ações de controle preditivo podem ser escritos na seguinte forma matricial

$$\underbrace{\begin{bmatrix} H_x \\ -H_x \end{bmatrix}}_{A_{xqp}} \hat{U}_{mpc} \leq \underbrace{\begin{bmatrix} X_{max} - P_x x(k) \\ -X_{min} + P_x x(k) \end{bmatrix}}_{b_{xqp}} + \underbrace{\begin{bmatrix} -P_x \\ P_x \end{bmatrix}}_{b_{xqp}} x(k) \quad (2.38)$$

- Restrições de controle

Analogamente, a restrição (2.8) pode ser escrita na seguinte forma:

$$u_{min} \leq \hat{u}(k|k) \leq u_{max} \quad (2.39)$$

$$u_{min} \leq \hat{u}(k+1|k) \leq u_{max} \quad (2.40)$$

$$\begin{aligned} & \vdots \\ u_{min} & \leq \hat{u}(k+N-1|k) \leq u_{max} \end{aligned} \quad (2.41)$$

No formato matricial, tem-se que

$$\underbrace{\begin{bmatrix} u_{min} \\ u_{min} \\ \vdots \\ u_{min} \end{bmatrix}}_{U_{min}} \leq \underbrace{\begin{bmatrix} \hat{u}(k|k) \\ \hat{u}(k+1|k) \\ \vdots \\ \hat{u}(k+N-1|k) \end{bmatrix}}_{\hat{U}} \leq \underbrace{\begin{bmatrix} u_{max} \\ u_{max} \\ \vdots \\ u_{max} \end{bmatrix}}_{U_{max}} \quad (2.42)$$

Substituindo (2.27) em (2.42), obtém-se

$$\begin{aligned} U_{min} &\leq P_u x(k) + H_u \hat{U}_{mpc} \leq U_{max} \\ U_{min} - P_u x(k) &\leq H_u \hat{U}_{mpc} \leq U_{max} - P_u x(k) \end{aligned} \quad (2.43)$$

As inequações de (2.43) podem ser reorganizadas no formato matricial como se segue

$$\begin{aligned} \begin{bmatrix} H_u \\ -H_u \end{bmatrix} \hat{U}_{mpc} &\leq \begin{bmatrix} U_{max} - P_u x(k) \\ -U_{min} + P_u x(k) \end{bmatrix} \\ \underbrace{\begin{bmatrix} H_u \\ -H_u \end{bmatrix}}_{A_{uqp}} \hat{U}_{mpc} &\leq \underbrace{\begin{bmatrix} U_{max} \\ -U_{min} \end{bmatrix}}_{b_{uqp}} + \underbrace{\begin{bmatrix} -P_u \\ P_u \end{bmatrix}}_{b_{uqp}} x(k) \end{aligned} \quad (2.44)$$

- Restrição terminal

A restrição terminal (2.9) é incluída no problema de otimização para obtenção de garantia de factibilidade recursiva. Isto é, para garantir que, partindo-se de uma condição inicial factível, o problema de otimização tenha solução nos demais instantes de amostragem, e que os estados sejam conduzidos para a origem (ROSSITER, 2004). Como resultado, a ação de controle não resultará em violação das restrições. Então, o conjunto de condições iniciais factíveis pode ser interpretado como região de operação do controlador. Essa região doravante será denotada por Θ cuja caracterização encontra-se no Apêndice A. Cabe salientar que esse tipo de garantia é de crucial importância para a implementação do MPC, uma vez que o problema de otimização é resolvido repetidamente durante a operação da planta.

Para imposição da restrição terminal, primeiramente deve-se caracterizar um conjunto invariante e admissível sob a ação da realimentação de estados. Essa caracterização é realizada considerando a dinâmica em MF (2.4), as restrições de estado e controle, que podem ser escritas da seguinte forma:

$$x_{min} \leq x(k) \leq x_{max} \quad (2.45)$$

$$u_{min} \leq -Kx(k) \leq u_{max} \quad (2.46)$$

As restrições (2.45) e (2.46) podem ser escritas na forma matricial a seguir

$$\underbrace{\begin{bmatrix} I \\ -I \\ -K \\ K \end{bmatrix}}_{A_x} x(k) \leq \underbrace{\begin{bmatrix} x_{max} \\ -x_{min} \\ u_{max} \\ -u_{min} \end{bmatrix}}_{b_x} \quad (2.47)$$

A partir da dinâmica (2.10) e a restrição (2.47), o MAS pode ser determinado conforme o procedimento detalhado em (GILBERT; TAN, 1991). Vale comentar que, como todos os autovalores de Φ estão dentro do círculo unitário, o MAS é finitamente determinado. Considerar-se-á que esse conjunto pode ser representado por

$$\mathcal{O}_\infty = \{x(k) \in \mathbb{R}^n | A_{O_\infty} x(k) \leq b_{O_\infty}\} \quad (2.48)$$

Então, a restrição terminal é imposta fazendo-se com que o estado $\hat{x}(k + N|k)$ esteja dentro do MAS, ou seja,

$$A_{O_\infty} \hat{x}(k + N|k) \leq b_{O_\infty} \quad (2.49)$$

Mais ainda, (2.16) por

$$\hat{x}(k + N|k) = \Phi^N x(k) + \underbrace{\begin{bmatrix} \Phi^{N-1} B & \Phi^{N-2} B & \dots & B \end{bmatrix}}_{H_B} \underbrace{\begin{bmatrix} \hat{u}_{mpc}(k|k) \\ \hat{u}_{mpc}(k + 1|k) \\ \vdots \\ \hat{u}_{mpc}(k + N - 1|k) \end{bmatrix}}_{\hat{U}_{mpc}} \quad (2.50)$$

De (2.49) e (2.50), a restrição terminal é imposta fazendo-se

$$\begin{aligned} A_{O_\infty} \hat{x}(k + N|k) &\leq b_{O_\infty} \\ A_{O_\infty} (\Phi^N x(k) + H_B \hat{U}_{mpc}) &\leq b_{O_\infty} \\ A_{O_\infty} \Phi^N x(k) + A_{O_\infty} H_B \hat{U}_{mpc} &\leq b_{O_\infty} \\ \underbrace{A_{O_\infty} H_B}_{A_{oqp}} \hat{U}_{mpc} &\leq \underbrace{b_{O_\infty} - A_{O_\infty} \Phi^N x(k)}_{b_{oqp}} \end{aligned} \quad (2.51)$$

As restrições (2.38), (2.44) e (2.51) podem ser combinadas da forma mostrada a seguir.

$$\underbrace{\begin{bmatrix} A_{xqp} \\ A_{uqp} \\ A_{oqp} \end{bmatrix}}_{A_{qp}} \hat{U}_{mpc} \leq \underbrace{\begin{bmatrix} b_{xqp} \\ b_{uqp} \\ b_{oqp} \end{bmatrix}}_{b_{qp}} \quad (2.52)$$

Portanto, o problema de programação quadrática que deve ser resolvido a cada instante de amostragem é

$$\min_{\hat{U}_{mpc}} J(k) = \hat{U}_{mpc}^T H_{qp} \hat{U}_{mpc} + 2x^T(k) f_{qp} \hat{U}_{mpc} \quad (2.53)$$

sujeito a

$$A_{qp} \hat{U}_{mpc} \leq b_{qp} \quad (2.54)$$

3 Estudo de caso no duplo integrador

Para fins de exemplo a lei de controle descrita no Capítulo 2 foi empregada para controlar um duplo integrador. Esse sistema é bastante utilizado na avaliação do desempenho de controladores, pois pode representar diferentes processos, como o movimento de translação ou rotação com um único grau de liberdade (RAO; BERNSTEIN, 2001). O modelo matemático do duplo integrador é

$$\ddot{y}(t) = u(t) \quad (3.1)$$

em que y e u podem, por exemplo, representar posição e aceleração, respectivamente.

Definindo $x_1(t) = y(t)$ e $x_2(t) = \dot{y}(t)$, é possível escrever a representação do sistema no espaço de estados, isto é,

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \quad (3.2)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad (3.3)$$

Discretizando (3.2) e (3.3) utilizando o método do segurador de ordem zero (ZOH – *Zero Order Hold*) para um período de amostragem de $T = 0,01$ s (OGATA, 1994), obtém-se

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0,01 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 5 \times 10^{-5} \\ 0,01 \end{bmatrix} u(k) \quad (3.4)$$

$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \quad (3.5)$$

As simulações foram realizadas no MATLAB, considerando-se diferentes estados iniciais. Adicionalmente, o MAS foi obtido utilizando o *toolbox* MPT (*Multi-Parametric Toolbox*) de Herceg et al. (2013). O horizonte de predição foi fixado em $N = 5$ para todos os casos a serem apresentados. As restrições de estados e controle adotadas foram $x_{max} = [1,5 \ 1,5]^T$, $x_{min} = [-2 \ -2]^T$ e $u_{max} = 10$, $u_{min} = -10$. As matrizes do MPC e do LQR foram ajustadas da seguinte maneira:

$$Q_{mpc} = \begin{bmatrix} 1 \times 10^6 & 0 \\ 0 & 5 \times 10^3 \end{bmatrix}, R_{mpc} = 1, Q_{lqr} = \begin{bmatrix} 100 & 0 \\ 0 & 10 \end{bmatrix} \text{ e } R_{lqr} = 1.$$

Como resultado, o vetor de ganhos do LQR é $K = [9,7299 \ 5,3784]$.

Vale destacar que os autovalores de Φ são $0,9729 \pm 0,0154i$. Portanto, o conjunto $\mathcal{O}_\infty$ é finitamente determinado. Os resultados obtidos são apresentados a seguir.

• Resultados para $x(0) = [1 \ -1]^T$

Como se pode notar na Figura 3.1a, a condição inicial está dentro do MAS. Verifica-se nas Figuras 3.1 e 3.2 que os estados são guiados para a origem (regulação) com o devido respeito às restrições de estado e controle. Em particular o estado x_2 e o controle total u atingem as restrições impostas, porém não há violação.

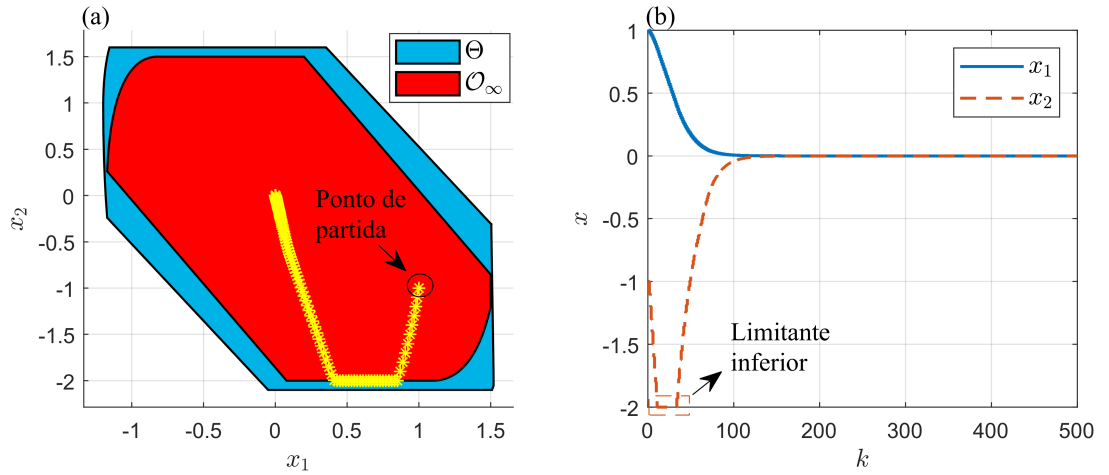


Figura 3.1 – Resultados obtidos para a condição inicial $x(0) = [1 \ -1]^T$: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.

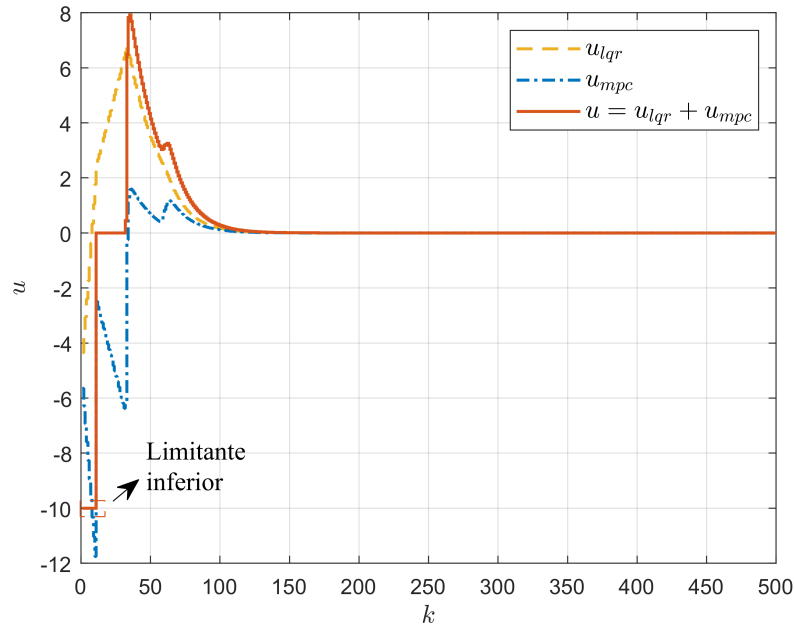


Figura 3.2 – Ações de controle para condição inicial $x(0) = [1 \ -1]^T$.

- Resultados para $x(0) = [1,5 \ -0,5]^T$

Nesse caso, a condição inicial pertence a Θ , porém não a $\mathcal{O}_\infty$. Então, a ação do controlador também resulta em convergência dos estados para a origem com atendimento às restrições.

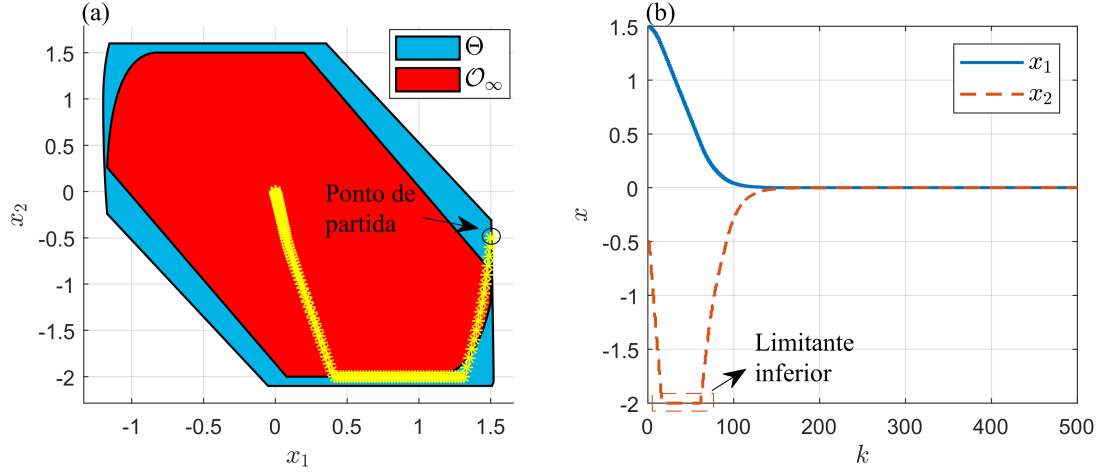


Figura 3.3 – Resultados obtidos para a condição inicial $x(0) = [1,5 \ -0,5]^T$: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.

As ações de controle podem ser visualizadas na Figura 3.4.

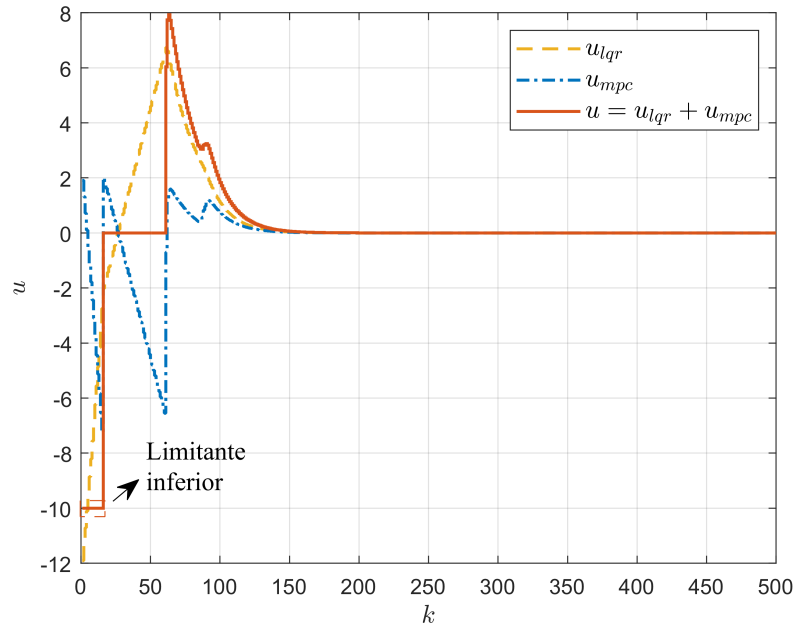


Figura 3.4 – Ações de controle para condição inicial $x(0) = [1,5 \ -0,5]^T$.

- Resultados para $x(0) = [1,5 \ -0,5]^T$ sem a ação do MPC ($u_{mpc}(k) = 0$)

A condição inicial é a mesma do exemplo anterior, porém a ação do MPC foi retirada para evidenciar que esse termo é responsável por garantir respeito às restrições. Conforme pode ser visualizado na Figura 3.5b, os estados atingem a origem sem violação dos limitantes. No entanto, na Figura 3.6 percebe-se que o houve violação dos limitantes sobre a ação de controle total.

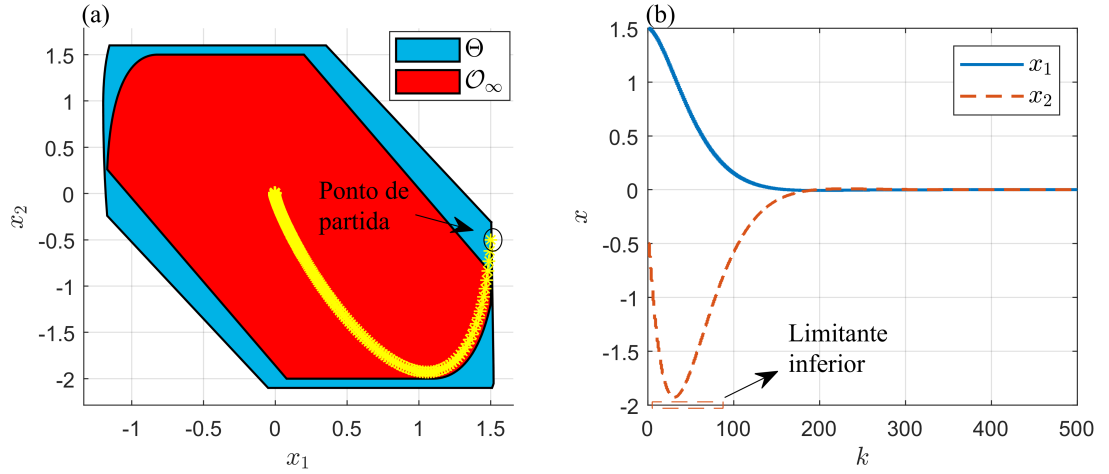


Figura 3.5 – Resultados obtidos para a condição inicial $x(0) = [1,5 \ -0,5]^T$ sem a ação do MPC: (a) Evolução dos estados no plano de fase, (b) Evolução dos estados a tempo discreto.

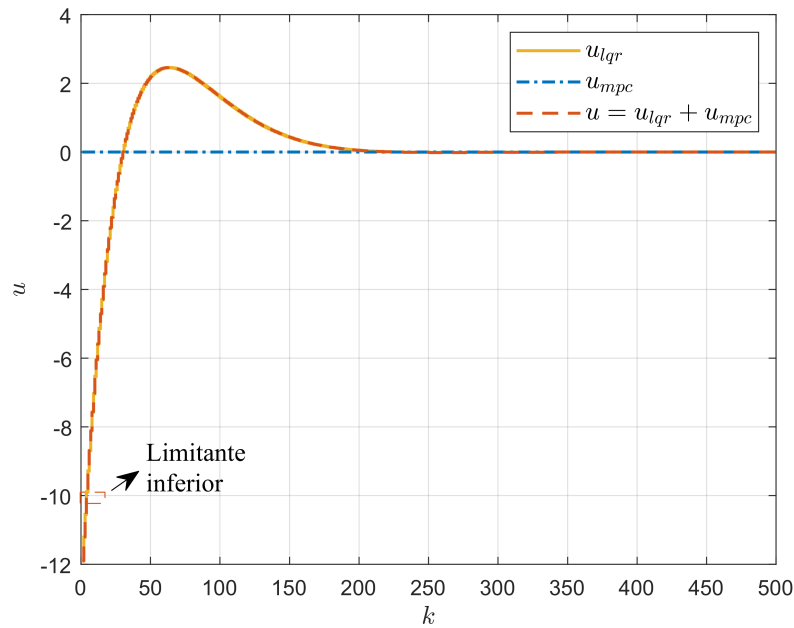


Figura 3.6 – Ações de controle para condição inicial $x(0) = [1,5 \ -0,5]^T$ sem a ação do MPC.

4 Estudo de caso no pêndulo invertido de duas rodas

Neste capítulo investiga-se o emprego do controle preditivo no veículo de duas rodas construído com o kit LEGO Mindstorms NXT[®] mostrado na Figura 1.2a. Além da composição estrutural, isto é, rodas, chassis, corpo, o sistema é equipado com motores DC e sensores ultrassônico, giroscópio e encoders. Denominar-se-á tal veículo como pêndulo invertido, pois a modelagem matemática é realizada considerando a dinâmica de um pêndulo (YAMAMOTO, 2008).

A tarefa de controle consiste em equilibrar o pêndulo, isto é, regular os estados no ponto de equilíbrio associado à posição vertical, respeitando restrições de estado e controle.

As variáveis de entrada para esse sistema são as tensões aplicadas aos motores esquerdo e direito. Os motores, por sua vez, são os atuadores responsáveis por aplicar torque ao sistema. Os encoders permitem a medição da posição angular das rodas e velocidade. Por fim, o sensor giroscópio permite a mensuração da taxa de variação da inclinação do pêndulo.

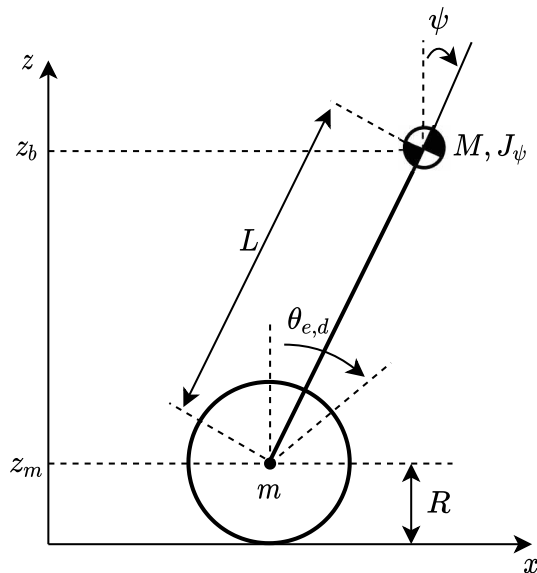


Figura 4.1 – Diagrama esquemático do pêndulo de duas rodas, em que θ e ψ são respectivamente a posição angular das rodas e a inclinação do centro de massa (YAMAMOTO, 2008) (Adaptada pelo autor).

A partir do diagrama esquemático da Figura 4.1, é possível obter as equações diferenciais que descrevem a dinâmica do veículo. Mais precisamente, obtêm-se o seguinte

modelo matemático (YAMAMOTO, 2008; CANALE; BRUNET, 2013):

$$T_\theta(t) = \left((2m + M)R^2 + 2n^2 J_m \right) \ddot{\theta}(t) - MLR \dot{\psi}^2(t) \sin\psi(t) + \left(MLR \cos\psi(t) - 2n^2 J_m \right) \ddot{\psi}(t) \quad (4.1)$$

$$T_\psi(t) = \left(MLR \cos\psi(t) - 2n^2 J_m \right) \ddot{\theta}(t) - MgL \sin\psi(t) + \left(ML^2 + J_\psi + 2n^2 J_m \right) \ddot{\psi}(t) \quad (4.2)$$

em que T_θ e T_ψ são os torques aplicados à roda e ao centro de massa, respectivamente. O significado dos demais símbolos encontra-se na Tabela 4.1.

As relações entre os torques e as tensões v_e e v_d aplicadas respectivamente aos motores esquerdo e direito são

$$T_\theta(t) = \alpha(v_e(t) + v_d(t)) - 2(\beta + f_w)\dot{\theta}(t) + 2\beta\dot{\psi}(t) \quad (4.3)$$

$$T_\psi(t) = -\alpha(v_e(t) + v_d(t)) + 2\beta\dot{\theta}(t) - 2\beta\dot{\psi}(t) \quad (4.4)$$

em que

$$\alpha = \frac{nK_t}{R_m} \quad (4.5)$$

$$\beta = \frac{nK_t K_b}{R_m} + f_m \quad (4.6)$$

$$v_e(t) = v_d(t) = G_u(\mu V_b - V_o)u(t) \quad (4.7)$$

sendo a variável controlada u o *duty cycle* do sinal PWM (*Pulse Width Modulation*) aplicado a ambos os motores.

Por conveniência, doravante omitir-se-á a dependência com o tempo das variáveis. É possível reescrever (4.1) e (4.2) de forma mais compacta por

$$T_\theta = c_1 \ddot{\theta} - c_2 \dot{\psi}^2 \sin\psi + (c_2 \cos\psi - c_3) \ddot{\psi} \quad (4.8)$$

$$T_\psi = (c_2 \cos\psi - c_3) \ddot{\theta} - c_4 \sin\psi + c_5 \ddot{\psi} \quad (4.9)$$

sendo

$$c_1 = (2m + M)R^2 + 2n^2 J_m \quad (4.10)$$

$$c_2 = MLR \quad (4.11)$$

$$c_3 = 2n^2 J_m \quad (4.12)$$

$$c_4 = MgL \quad (4.13)$$

$$c_5 = ML^2 + J_\psi + 2n^2 J_m \quad (4.14)$$

O modelo matemático dado por (4.3), (4.4), (4.8) e (4.9) é não linear. Para implementar o MPC descrito no Capítulo 2 é necessário linearizar tal modelo em torno da posição

Tabela 4.1 – Parâmetros do pêndulo invertido (YAMAMOTO, 2008).

Símbolo e valor adotado	Descrição
$g = 9,81 \text{ m/s}^2$	Aceleração da gravidade
$m = 0,03 \text{ kg}$	Massa de cada roda
$R = 0,04 \text{ m}$	Raio da roda
$M = 0,6 \text{ kg}$	Massa considerada no centro de massa
$L = 0,072 \text{ m}$	Distância do centro de massa ao eixo de giro da roda
$J_\psi = 0,001 \text{ kg}\cdot\text{m}^2$	Momento de inércia em relação a ψ
$J_m = 10^{-5} \text{ kg}\cdot\text{m}^2$	Momento de inércia do motor DC
$R_m = 6,69 \Omega$	Resistência elétrica do motor
$K_b = 0,468 \text{ V}\cdot\text{s/rad}$	Constante de força eletromotriz do motor
$K_t = 0,317 \text{ N}\cdot\text{m/A}$	Constante de torque do motor
$f_m = 0,0022 \text{ N}\cdot\text{m}\cdot\text{rad/s}$	Coefficiente de fricção do motor
$V_b = 8,00 \text{ V}$	Tensão de alimentação
$V_o = 0,625 \text{ V}$	Offset de tensão
$\mu = 1,089$	Fator de ganho da alimentação
$G_u = 10^{-2}$	Fator de ganho do PWM
$n = 1$	Relação de transmissão do motor
$f_w = 0$	Coefficiente de atrito entre a roda e o solo

vertical. A linearização é simplificada caso a dependência entre os termos de segunda ordem em (4.8) e (4.9) seja removida. Para isso, pode-se isolar $\ddot{\psi}$ em (4.9), obtendo-se

$$\ddot{\psi} = \frac{1}{c_5} (T_\psi - (c_2 \cos \psi - c_3)\ddot{\theta} + c_4 \sin \psi) \quad (4.15)$$

Substituindo (4.15) em (4.8), seque que

$$\begin{aligned}
T_\theta &= c_1 \ddot{\theta} - c_2 \dot{\psi}^2 \sin \psi + \underbrace{\frac{c_2 \cos \psi - c_3}{c_5}}_{h_1} (T_\psi - (c_2 \cos \psi - c_3)\ddot{\theta} + c_4 \sin \psi) \\
&= c_1 \ddot{\theta} - c_2 \dot{\psi}^2 \sin \psi + h_1 T_\psi - h_1 (c_2 \cos \psi - c_3)\ddot{\theta} + h_1 c_4 \sin \psi \\
&= (c_1 - h_1 (c_2 \cos \psi - c_3))\ddot{\theta} + h_1 (T_\psi + c_4 \sin \psi) - c_2 \dot{\psi}^2 \sin \psi \\
\ddot{\theta} &= \frac{1}{c_1 - h_1 (c_2 \cos \psi - c_3)} (T_\theta - h_1 (T_\psi + c_4 \sin \psi) + c_2 \dot{\psi}^2 \sin \psi) \quad (4.16)
\end{aligned}$$

De (4.3), (4.4) e (4.16), é possível escrever

$$\begin{aligned}
\ddot{\theta} &= \frac{1}{c_1 - h_1 (c_2 \cos \psi - c_3)} \left[(\alpha(v_e + v_d) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi}) \right. \\
&\quad \left. - h_1(-\alpha(v_e + v_d) + 2\beta\dot{\theta} - 2\beta\dot{\psi} + c_4 \sin \psi) + c_2 \dot{\psi}^2 \sin \psi \right] \\
&= \frac{1}{c_1 - h_1 (c_2 \cos \psi - c_3)} \left(\alpha(v_e + v_d) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi} + h_1 \alpha(v_e + v_d) \right. \\
&\quad \left. - h_1 2\beta\dot{\theta} + h_1 2\beta\dot{\psi} - h_1 c_4 \sin \psi + c_2 \dot{\psi}^2 \sin \psi \right)
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{c_1 - h_1(c_2 \cos \psi - c_3)} \left[\left(-2(\beta + f_w) - h_1 2\beta \right) \dot{\theta} + (2\beta + h_1 2\beta) \dot{\psi} \right. \\
&\quad \left. + (-h_1 c_4 + c_2 \dot{\psi}^2) \operatorname{sen} \psi + \alpha(v_e + v_d) + h_1 \alpha(v_e + v_d) \right] \\
\ddot{\theta} &= \frac{1}{c_1 - h_1(c_2 \cos \psi - c_3)} \left[-2(\beta(1 + h_1) + f_w) \dot{\theta} + 2\beta(1 + h_1) \dot{\psi} \right. \\
&\quad \left. + (c_2 \dot{\psi}^2 - h_1 c_4) \operatorname{sen} \psi + \alpha(1 + h_1)(v_e + v_d) \right]
\end{aligned} \tag{4.17}$$

Seguindo um procedimento similar para $\ddot{\psi}$, pode-se obter

$$\begin{aligned}
\ddot{\psi} &= \frac{1}{c_5 - h_2(c_2 \cos \psi - c_3)} \left[2(\beta + h_2(\beta + f_w)) \dot{\theta} - 2\beta(1 + h_2) \dot{\psi} \right. \\
&\quad \left. + (c_4 - h_2 c_2 \dot{\psi}^2) \operatorname{sen} \psi - \alpha(1 + h_2)(v_e + v_d) \right]
\end{aligned} \tag{4.18}$$

Definindo-se $x_1 = \theta$, $x_2 = \psi$, $x_3 = \dot{\theta}$ e $x_4 = \dot{\psi}$, $u_1 = v_e$, $u_2 = v_d$ é possível representar a dinâmica do veículo no espaço de estados por

$$\dot{x}_1 = x_3 \tag{4.19}$$

$$\dot{x}_2 = x_4 \tag{4.20}$$

$$\begin{aligned}
\dot{x}_3 &= \frac{1}{c_1 - h_1(c_2 \cos x_2 - c_3)} \left[-2(\beta(1 + h_1) + f_w) x_3 + 2\beta(1 + h_1) x_4 \right. \\
&\quad \left. + (c_2 x_4^2 - h_1 c_4) \operatorname{sen} x_2 + \alpha(1 + h_1)(u_1 + u_2) \right]
\end{aligned} \tag{4.21}$$

$$\begin{aligned}
\dot{x}_4 &= \frac{1}{c_5 - h_2(c_2 \cos x_2 - c_3)} \left[2(\beta + h_2(\beta + f_w)) x_3 - 2\beta(1 + h_2) x_4 \right. \\
&\quad \left. + (c_4 - h_2 c_2 x_4^2) \operatorname{sen} x_2 - \alpha(1 + h_2)(u_1 + u_2) \right]
\end{aligned} \tag{4.22}$$

em que

$$h_1 = \frac{c_2 \cos x_2 - c_3}{c_5} \tag{4.23}$$

$$h_2 = \frac{c_2 \cos x_2 - c_3}{c_1} \tag{4.24}$$

Cabe salientar que a mudança de v_e para u_1 e v_d para u_2 se dá apenas por conveniência da nomenclatura das entradas na literatura de controle.

O modelo matemático (4.19) – (4.22) é não linear. Aplicando o procedimento de linearização descrito no Apêndice B e substituindo os valores das constantes apresentadas na Tabela 4.1, é possível escrever o seguinte modelo no espaço de estados:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -474,01 & -186,39 & 186,39 \\ 0 & 298,62 & 88,88 & -88,88 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 181,17 & 181,17 \\ -86,39 & -86,39 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \quad (4.25)$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ y_4(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} \quad (4.26)$$

Discretizando (4.25) e (4.26) utilizando o método do ZOH para um período de amostragem de $T = 4 \text{ ms}$ (OGATA, 1994), obtém-se

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ x_4(k+1) \end{bmatrix} = \begin{bmatrix} 1 & -0,0026 & 0,0029 & 0,0011 \\ 0 & 1,0018 & 0,0005 & 0,0035 \\ 0 & -1,0729 & 0,5478 & 0,4496 \\ 0 & 0,8021 & 0,2157 & 0,7861 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \end{bmatrix} + \begin{bmatrix} 0,0010 & 0,0010 \\ -0,0005 & -0,0005 \\ 0,4395 & 0,4395 \\ -0,2096 & -0,2096 \end{bmatrix} \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix} \quad (4.27)$$

$$\begin{bmatrix} y_1(k) \\ y_2(k) \\ y_3(k) \\ y_4(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \end{bmatrix} \quad (4.28)$$

A lei de controle descrita no Capítulo 2 foi empregada em simulação para regular os estados do sistema composto por (4.27) e (4.28) utilizando o MATLAB e sendo considerado diferentes condições iniciais. Adicionalmente, o MAS foi caracterizado por meio do *toolbox* MPT de Herceg et al. (2013). Limitou-se superiormente e inferiormente o estado x_2 em 20° e o controle inferiormente e superiormente em 8 V. As matrizes do MPC e do LQR foram ajustadas da seguinte maneira:

$$Q_{mpc} = \begin{bmatrix} 1 \times 10^2 & 0 & 0 & 0 \\ 0 & 4,5 \times 10^6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad R_{mpc} = \begin{bmatrix} 3,5 & 0 \\ 0 & 3,5 \end{bmatrix},$$

$$Q_{lqr} = \begin{bmatrix} 4,2 \times 10^2 & 0 & 0 & 0 \\ 0 & 6,5 \times 10^3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{e} \quad R_{lqr} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}.$$

Como resultado, o vetor de ganhos do LQR é

$$K = \begin{bmatrix} -4,2511 & -49,7203 & -2,2298 & -4,8531 \\ -4,2511 & -49,7203 & -2,2298 & -4,8531 \end{bmatrix}.$$

Os resultados são apresentados e discutidos a seguir.

- Resultados para $x(0) = [0 \ 5 \ 0 \ 0]^T$

O horizonte de predição adotado para esse caso foi $N = 10$. A condição inicial considerada equivale a de 5° do centro de massa do veículo.

A Figura 4.2 apresenta os resultados para essa condição inicial. Observa-se que os estados atingem a origem rapidamente, porém o estado x_1 tende a divergir à medida que o tempo passa. Esse comportamento foi observado mesmo para valores de k maiores que 500. Na Figura 4.2b destaca-se a inclinação do pêndulo. Nota-se que esse estado é mantido em equilíbrio.

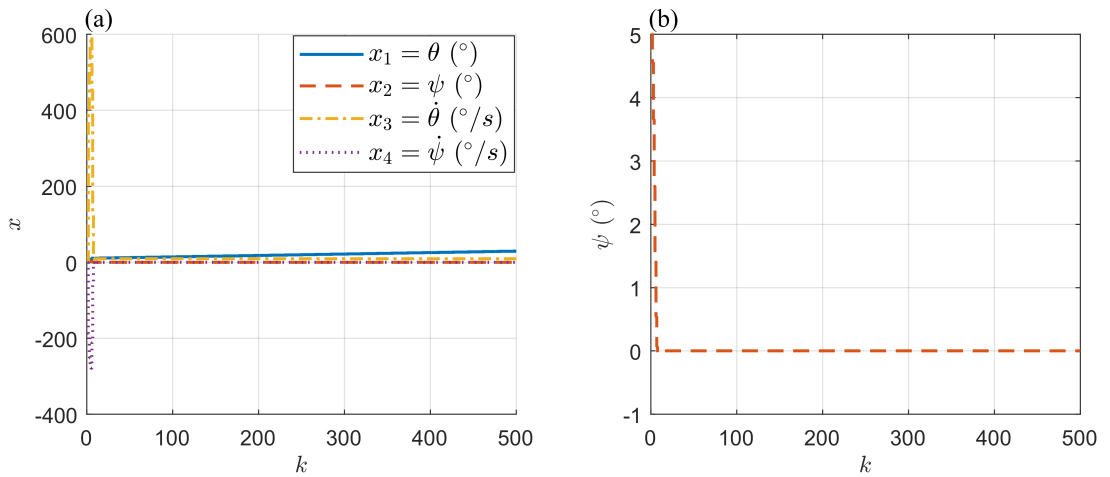


Figura 4.2 – Resultados obtidos para a condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.

As ações de controle são apresentadas na Figura 4.3. Percebe-se que o controle total aplicado à planta atinge o limitante superior no início da simulação, porém não há violação das restrições. Vale destacar que as ações individuais do LQR e MPC divergem, mas o controle total, calculado somando-se essas ações, converge para aproximadamente 0,1 V.

Esses resultados são intrigantes, pois esperava-se que houvesse regulação de todos os estados e sinais de controle até o último passo da simulação, assim como ocorreu nos resultados para o duplo integrador. O problema é que a formulação apresentada no Capítulo 2 não tem garantia de convergência dos estados para a origem (ROSSITER, 2004). Para contornar esse problema, adotou-se a formulação de controle preditivo descrita em Afonso e Galvão (2011). A principal diferença é que a função de custo envolve somente

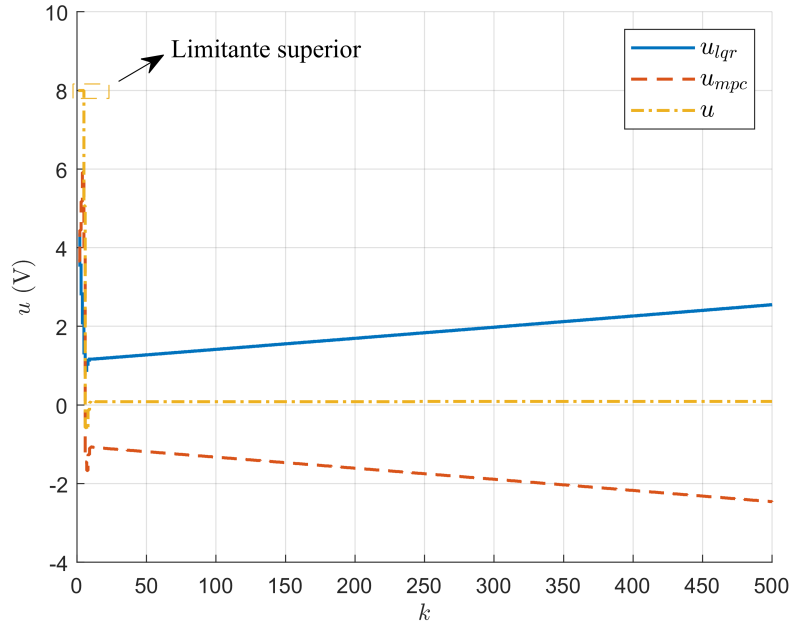


Figura 4.3 – Ações de controle para condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$.

as ações do MPC. Mais precisamente, adota-se

$$J(k) = \sum_{i=1}^N \hat{u}_{mpc}^T(k+i-1|k) \Psi \hat{u}_{mpc}(k+i-1|k) \quad (4.29)$$

em que

$$\Psi = R_{lqr} + B^T P B \quad (4.30)$$

e sendo R_{lqr} a matriz de ajuste das entradas do LQR, P a única solução simétrica não-negativa da equação algébrica de Riccati e B a matriz das entradas definida em (2.1). Essa definição de Ψ é vantajosa pois equivale a trabalhar com uma função de custo de horizonte de predição infinito (AFONSO; GALVÃO, 2011). Como resultado, pode-se demonstrar que a ação de controle resultará em convergência dos estados para a origem (ROSSITER, 2004). Vale ressaltar que devido à forma de construção de (4.29), o MPC só atua para garantir respeito às restrições.

Então, o problema de programação quadrática a ser resolvido é

$$\min_{\hat{U}_{mpc}} J(k) = \hat{U}_{mpc}^T \bar{\Psi} \hat{U}_{mpc} \quad (4.31)$$

sujeito a (2.54), em que

$$\bar{\Psi} = \begin{bmatrix} \Psi & 0 & \cdots & 0 \\ 0 & \Psi & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Psi \end{bmatrix} \quad (4.32)$$

Os resultados para essa formulação são apresentados e discutidos a seguir.

- Resultados para $x(0) = [0 \ 5 \ 0 \ 0]^T$

A inclinação inicial e o horizonte de predição são os mesmos do exemplo anterior. Como pode ser notado na Figura 4.4, todos os estados foram regulados com o devido atendimento às restrições. Em especial, a Figura 4.4b destaca a inclinação do pêndulo. As restrições de controle também foram atendidas, assim como mostrado na Figura 4.5. Nota-se ainda que durante todo o decorrer da simulação, a ação de controle do MPC é nula. Isso ocorreu pois as restrições foram devidamente atendidas somente sob a ação do LQR.

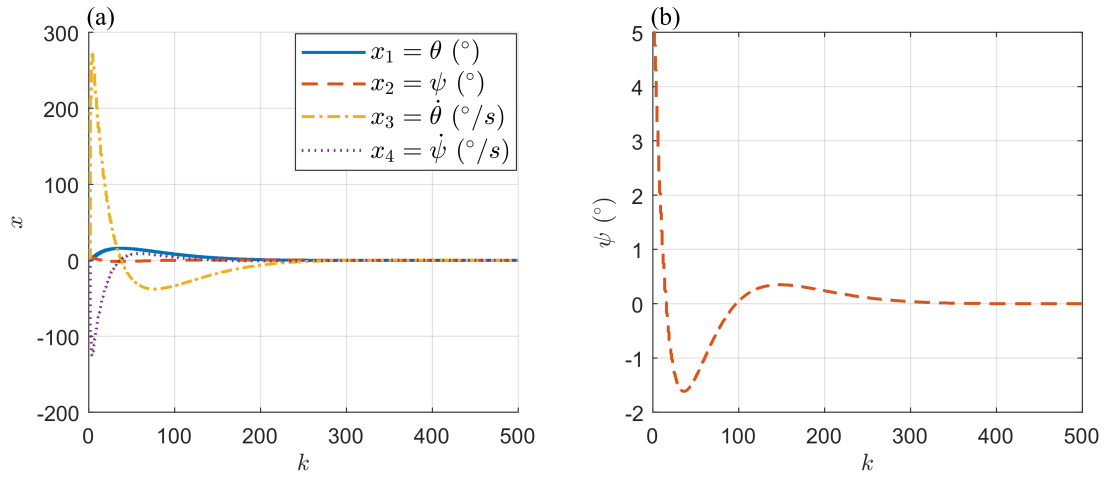


Figura 4.4 – Resultados obtidos para a condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.

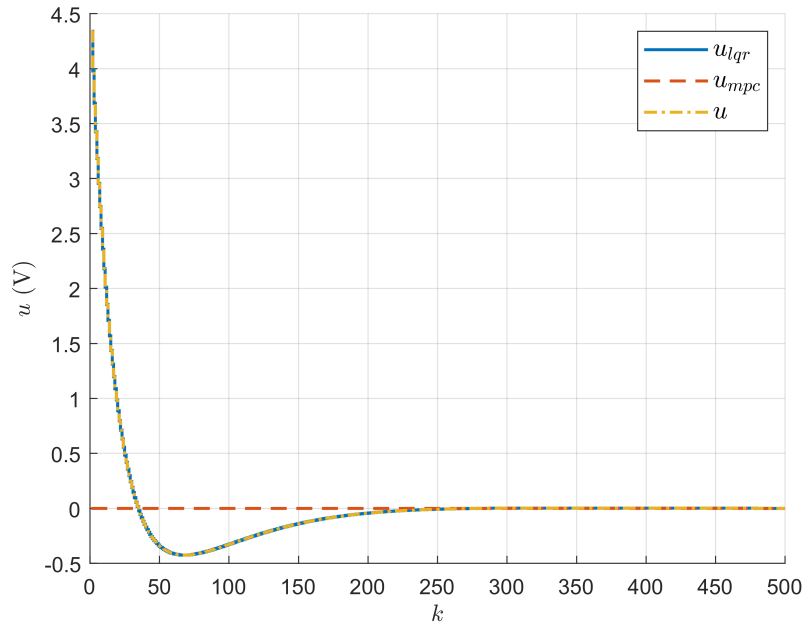


Figura 4.5 – Ações de controle para condição inicial $x(0) = [0 \ 5 \ 0 \ 0]^T$.

• Resultados para $x(0) = [0 \ 15 \ 0 \ 0]^T$

Nesse cenário, o veículo tem uma inclinação inicial de 15° . Além da restrição colocada anteriormente sobre x_2 , para esse caso também foi imposto $|x_3| \leq 450^\circ/\text{s}$. Aumentou-se horizonte de predição para $N = 25$ a fim de evitar problemas de factibilidade inicial.

Observando as Figuras 4.6 e 4.7, nota-se que ação do controlador também resultou em convergência dos estados para a origem com atendimento às restrições. Nota-se ainda na Figura 4.6a que o estado x_3 atinge seu limite superior.

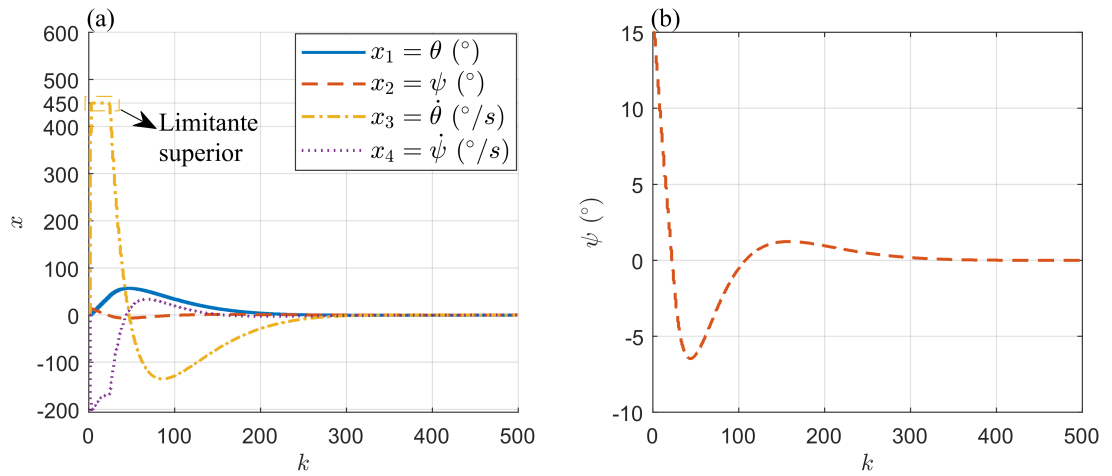


Figura 4.6 – Resultados obtidos para a condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$: (a) Evolução dos estados, (b) Inclinação do pêndulo.

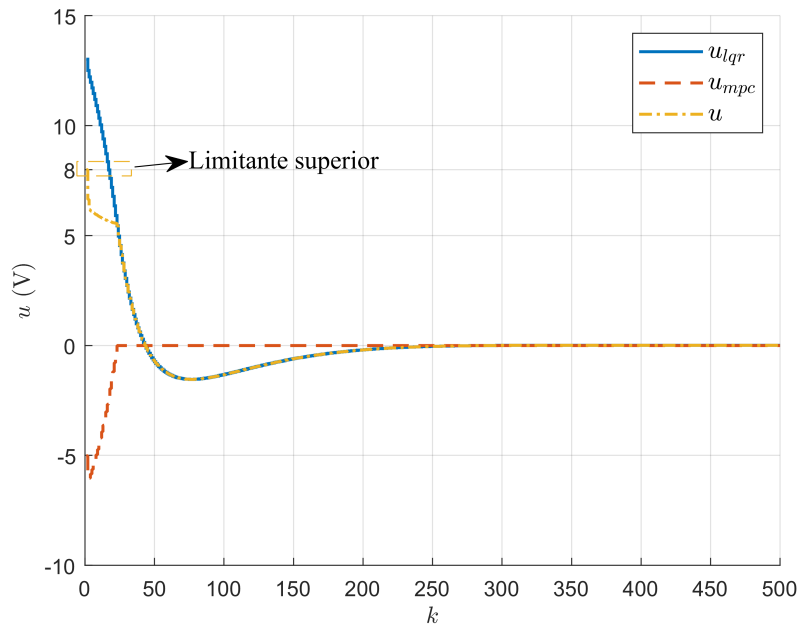


Figura 4.7 – Ações de controle para condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$.

Vale notar na Figura 4.7 que foi necessária a ação do MPC para impedir a violação das restrições de estado e controle.

- Resultados para $x(0) = [0 \ 15 \ 0 \ 0]^T$ para o modelo não linear do pêndulo

As condições de simulação são as mesmas do exemplo anterior, porém adotou-se o modelo não linear do pêndulo e adicionaram-se ruídos de medida nas leituras dos estados. Mais especificamente, adicionou-se em todos os estados um sinal aleatório limitado em $\pm 1^\circ$ de distribuição uniforme. Mais ainda, para considerar um descasamento de modelo, reduziu-se a ação de controle em 10 % antes de se evoluir a dinâmica da planta. Isso equivale a um descasamento de 10 % na matriz de entrada B .

As Figuras 4.8 e 4.9 apresentam os resultados obtidos. Observa-se que a tarefa de controle foi bem-sucedida. Também não houve violação das restrições de controle e estado. Isso ilustra que a formulação de MPC adotada apresenta certa robustez contra descasamentos de modelo e ruídos de medida, ainda que esses efeitos sejam desconsiderados no projeto do controlador.

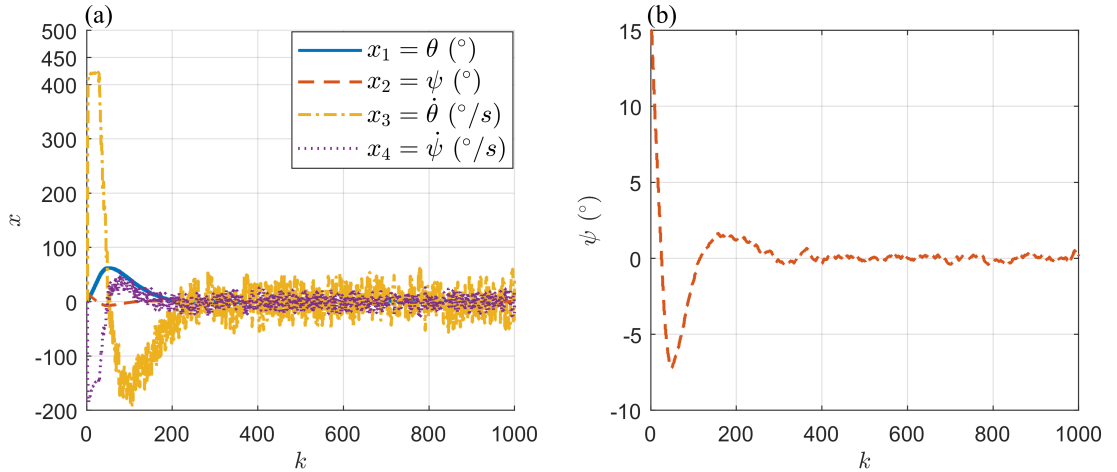


Figura 4.8 – Resultados obtidos para a condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$ para o modelo não linear do pêndulo: (a) Evolução dos estados, (b) Inclinação do pêndulo.

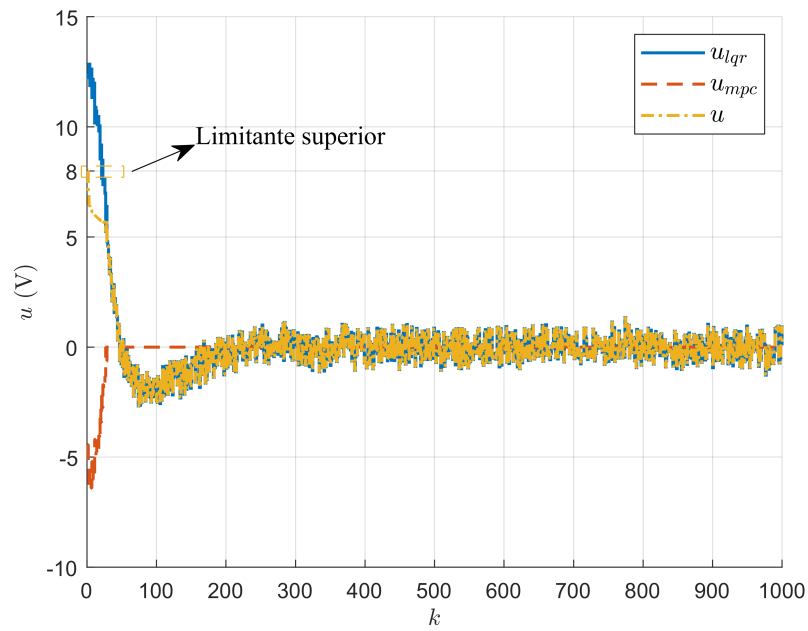


Figura 4.9 – Ações de controle para condição inicial $x(0) = [0 \ 15 \ 0 \ 0]^T$ para o modelo não linear do pêndulo.

Assim como para o caso linear, nota-se na Figura 4.9 que foi necessária a ação do MPC para impedir a violação das restrições de estado e controle.

5 Conclusões e Trabalhos Futuros

Este trabalho teve como objetivo implementar uma lei de controle preditivo em um pêndulo invertido de duas rodas em ambiente de simulação. A tarefa de controle consistiu em equilibrar o pêndulo na posição vertical garantindo o devido atendimento às restrições do sistema. Cabe salientar que essa dinâmica é instável em malha aberta, o que torna desafiador o projeto da lei de controle.

Para atender tal tarefa, adotaram-se formulações de MPC com malha interna. Isto é, considerando uma realimentação de estados estabilizante. Como resultado, as predições foram realizadas a partir de um modelo estável, o que é vantajoso em termos computacionais. Considerando essa estrutura de controle, foi possível caracterizar uma região invariante na qual o sistema em MF apenas com a malha interna garante o respeito às restrições de controle e estado. Essa região denominada MAS (*Maximal Admissible Set*) foi incluída no problema de otimização como restrição terminal. Mais ainda, determinou-se uma região Θ no espaço de estados na qual o problema de otimização é inicialmente factível. Caso a formulação de MPC tenha garantia de factibilidade recursiva e convergência dos estados para a origem, Θ pode ser interpretada como a região de operação do controlador.

No Capítulo 2 descreveu-se uma formulação de MPC com garantia de factibilidade recursiva, porém sem garantia de convergência dos estados para a origem. Isto é, não se garante que a ação de controle preditivo guiará o estado para a origem. Essa formulação apresentou um desempenho adequado no controle do duplo integrador. Entretanto, nas simulações no pêndulo invertido, observou-se divergência das ações individuais de controle (u_{lqr} e u_{mpc}) e do estado θ . Para eliminar o problema, modificou-se a função de custo para obtenção da garantia de convergência. Em particular, a função empregada envolve somente a minimização da ação de controle do MPC e considera horizonte de predição infinito. Com efeito, foi possível manter o pêndulo na posição invertida respeitando-se as restrições sobre o canal de controle. Isso reforça a importância do emprego de formulações de MPC com garantia de estabilidade e convergência. Por fim, foram realizadas simulações no modelo não linear, considerando descasamento de modelo e ruídos de medida. O desempenho do controlador foi adequado, o que indica robustez da formulação de controle.

Como trabalhos futuro sugere-se a implementação do controlador no sistema real ou ainda, modificar a formulação para permitir rastreamento de referências de inclinação. Isso pode ser utilizado para movimentar o veículo.

Referências

- AFONSO, R. J. M.; GALVÃO, R. K. Uso de conjuntos terminais parametrizados para tratamento de problemas de não-factibilidade em controle preditivo. *Revista Controle & Automação*, v. 23, n. 6, p. 653–666, 2011. Citado 4 vezes nas páginas 7, 15, 34 e 35.
- CANALE, M.; BRUNET, S. C. A lego mindstorms nxt experiment for model predictive control education. In: *2013 European Control Conference (ECC)*. [S.l.: s.n.], 2013. p. 2549–2554. Citado 2 vezes nas páginas 13 e 30.
- CUTLER, C. R.; RAMAKER, B. L. Dynamic matrix control - a computer control algorithm. *Joint Automatic Control Conference*, v. 17, p. 72, 1980. Citado na página 13.
- FRANKLIN, G. F.; POWELL, J. D.; EMAMI-NAEINI, A. *Feedback Control of Dynamic Systems*. 6. ed. Upper Saddle River: Pearson, 2009. Citado 2 vezes nas páginas 12 e 13.
- GILBERT, E. G.; TAN, K. T. Linear systems with state and control constraints: the theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, v. 36, n. 9, p. 1008–1020, 1991. Citado na página 23.
- HERCEG, M. et al. Multi-Parametric Toolbox 3.0. In: *Proc. of the European Control Conference*. Zürich, Switzerland: [s.n.], 2013. p. 502–510. <<http://control.ee.ethz.ch/~mpt>>. Citado 2 vezes nas páginas 25 e 33.
- MACIEJOWSKI, J. M. *Predictive Control with Constraints*. 1. ed. England: Prentice Hall, 2002. Citado 3 vezes nas páginas 12, 13 e 15.
- NISE, N. S. *Control Systems Engineering*. 6. ed. New York: LTC, 2011. Citado na página 12.
- OGATA, K. *Discrete-Time Control Systems*. 2. ed. New Jersey: Prentice-Hall, Inc., 1994. Citado 4 vezes nas páginas 16, 25, 33 e 44.
- PEREIRA, B. L.; SANCHES, L.; TAVARES, J. J.-P. Z. de S. Neuro-fuzzy control and differential evolution in the compensation of the actuators' dead zone of a two-wheeled vehicle. In: . [S.l.: s.n.], 2017. Citado na página 13.
- QIN, S.; BADGWELL, T. A. A survey of industrial model predictive control technology. *Control Engineering Practice*, v. 11, n. 7, p. 733 – 764, 2003. ISSN 0967-0661. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0967066102001867>>. Citado na página 13.
- RAO, V.; BERNSTEIN, D. Naive control of the double integrator. *Control Systems, IEEE*, v. 21, p. 86 – 97, 11 2001. Citado na página 25.
- RICHALET, J. et al. Model predictive heuristic control: Applications to industrial processes. *Autom.*, v. 14, p. 413–428, 1978. Citado na página 13.
- ROSSITER, J. A. *Model-based Predictive Control: A Practical Approach*. 1. ed. Boca Raton: CRC Press LLC, 2004. Citado 8 vezes nas páginas 12, 13, 15, 17, 20, 22, 34 e 35.

SILVA, G. D. *Projeto, modelagem, simulação, construção e controle de um veículo de duas rodas pêndulo invertido*. 2017. <<https://repositorio.ufu.br/handle/123456789/25442>>. Online; acessado em 29 de Junho de 2020. Citado 2 vezes nas páginas 7 e 14.

STEWART, J. *Cálculo*: Tradução da 7ª edição norte-americana. 7. ed. São Paulo: CENGAGE Learning, 2014. I. ISBN 978-85-221-1461-0. Citado na página 44.

YAMAMOTO, Y. *NXTway-GS Model-Based Design - Control of self-balancing two-wheeled robot built with LEGO Mindstorm NXT*. [S.l.], 2008. Disponível em: <http://www.pages.drexel.edu/~dml46/Tutorials/BalancingBot/files/NXTway-GS%20Model-Based_Design.pdf>. Acesso em: 02 jun. 2020. Citado 7 vezes nas páginas 7, 8, 13, 14, 29, 30 e 31.

APÊNDICE A – Caracterização do conjunto de atração

As restrições do problema de otimização dadas por (2.54), podem ser escritas da seguinte forma:

$$A_{qp}\hat{U}_{mpc} \leq \begin{bmatrix} X_{max} \\ -X_{min} \\ U_{max} \\ -U_{min} \\ b_{O_\infty} \end{bmatrix} + \begin{bmatrix} -P_x \\ P_x \\ -P_u \\ P_u \\ -A_{O_\infty}\Phi^N \end{bmatrix} x(k) \quad (A.1)$$

Então, é possível obter

$$\begin{aligned} A_{qp}\hat{U}_{mpc} + \underbrace{\begin{bmatrix} P_x \\ -P_x \\ P_u \\ -P_u \\ A_{O_\infty}\Phi^N \end{bmatrix}}_{A_f} x(k) &\leq \underbrace{\begin{bmatrix} X_{max} \\ -X_{min} \\ U_{max} \\ -U_{min} \\ b_{O_\infty} \end{bmatrix}}_{b_{xu}} \\ \underbrace{\begin{bmatrix} A_{qp} & A_f \end{bmatrix}}_{A_{xu}} \begin{bmatrix} \hat{U}_{mpc} \\ x(k) \end{bmatrix} &\leq b_{xu} \end{aligned} \quad (A.2)$$

As matrizes A_{xu} e b_{xu} definem o conjunto de soluções factíveis para o problema de otimização no espaço $\hat{U}_{mpc} \times x(k)$. Projetando-se tal conjunto no espaço de estados, tem-se a região para os quais o problema de otimização tem solução, isto é, o conjunto de operação Θ . Cabe salientar que esse conjunto pode ser representado por

$$\Theta = \{x(k) \in \mathbb{R}^n | A_\Theta x(k) \leq b_\Theta\} \quad (A.3)$$

APÊNDICE B – Linearização da dinâmica do pêndulo invertido

Um sistema não linear e invariante a tempo contínuo pode ser representando no espaço de estados por (OGATA, 1994)

$$\dot{x}(t) = f(x(t), u(t)) \quad (\text{B.1})$$

$$y(t) = g(x(t), u(t)) \quad (\text{B.2})$$

em que $x \in \mathbb{R}^n$, $u \in \mathbb{R}^r$ e f e g são funções que descrevem a dinâmica do sistema.

Com o devido procedimento de linearização, é possível representar a dinâmica (B.1) e (B.2) em torno do par de equilíbrio (x_0, u_0) por

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (\text{B.3})$$

$$y(t) = Cx(t) \quad (\text{B.4})$$

em que A , B e C têm dimensões adequadas.

Mais precisamente, (B.3) e (B.4) podem ser obtidas utilizando a linearização por série de Taylor da seguinte forma (STEWART, 2014):

$$\underbrace{\begin{bmatrix} \delta \dot{x}_1 \\ \delta \dot{x}_2 \\ \vdots \\ \delta \dot{x}_n \end{bmatrix}}_{\delta \dot{x}} = \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}}_A \underbrace{\begin{bmatrix} \delta x_1 \\ \delta x_2 \\ \vdots \\ \delta x_n \end{bmatrix}}_{\delta x} + \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_r} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_r} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_r} \end{bmatrix}}_B \underbrace{\begin{bmatrix} \delta u_1 \\ \delta u_2 \\ \vdots \\ \delta u_r \end{bmatrix}}_{\delta u} \quad (B.5)$$

$x = x_0$
 $u = u_0$

$$\underbrace{\begin{bmatrix} \delta y_1 \\ \delta y_2 \\ \vdots \\ \delta y_n \end{bmatrix}}_{\delta y} = \underbrace{\begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_n}{\partial x_1} & \frac{\partial g_n}{\partial x_2} & \cdots & \frac{\partial g_n}{\partial x_n} \end{bmatrix}}_C \underbrace{\begin{bmatrix} \delta x_1 \\ \delta x_2 \\ \vdots \\ \delta x_n \end{bmatrix}}_{\delta x} \quad (B.6)$$

$x = x_0$
 $u = u_0$

Resumidamente,

$$\delta \dot{x}(t) = A \delta x(t) + B \delta u(t) \quad (B.7)$$

$$\delta y(t) = C \delta x(t) \quad (B.8)$$

Por simplicidade, omite-se o termo δ em (B.7) e (B.8), resultando em (B.3) e (B.4).

No caso do pêndulo invertido, a condição de operação é dada por

$$\theta_0 = \psi_0 = \dot{\theta}_0 = \dot{\psi}_0 = v_{0e} = v_{0d} = 0 \quad (B.9)$$

e,

$$x_{10} = x_{20} = x_{30} = x_{40} = u_{10} = u_{20} = 0 \quad (B.10)$$

De (4.19) – (4.22), (B.5) e (B.6), é possível obter

$$\begin{aligned}
 \begin{bmatrix} \delta \dot{x}_1 \\ \delta \dot{x}_2 \\ \delta \dot{x}_3 \\ \delta \dot{x}_4 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{c_4(c_2-c_3)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & -\frac{2(\beta(c_2-c_3+c_5)+c_5f_w)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & -\frac{2\beta(c_2-c_3+c_5)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} \\ 0 & \frac{c_1c_4}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & \frac{2(\beta(c_1+c_2-c_3)+f_w(c_2-c_3))}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & -\frac{2\beta(c_1+c_2-c_3)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \delta x_2 \\ \delta x_3 \\ \delta x_4 \end{bmatrix} + \\
 &\quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{\alpha(c_2-c_3+c_5)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & \frac{\alpha(c_2-c_3+c_5)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} \\ -\frac{\alpha(c_1+c_2-c_3)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} & -\frac{\alpha(c_1+c_2-c_3)}{-c_2^2+2c_2c_3-c_3^2+c_1c_5} \end{bmatrix} \begin{bmatrix} \delta u_1 \\ \delta u_2 \end{bmatrix} \quad (B.11)
 \end{aligned}$$

$$\begin{aligned}
 \begin{bmatrix} \delta y_1 \\ \delta y_2 \\ \delta y_3 \\ \delta y_4 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \delta x_2 \\ \delta x_3 \\ \delta x_4 \end{bmatrix} \quad (B.12)
 \end{aligned}$$

Cabe salientar que (B.11) e (B.12) representam (4.19) – (4.22) para pequenas variações dos ângulos θ e ψ em torno da posição vertical.

APÊNDICE C – Códigos das simulações

C.1 Matrizes do MPC

```

1 % arquivo: matricesMPC.m
2
3 function [Phi, Hqp, fqp, Hx, Px, Hu, Pu] = matricesMPC(A, B, Q, R, ...
    K, N)
4     % Funcao que retorna as matrizes do MPC desenvolvidas no ...
        Capitulo 2
5     % para um modelo linear  $x(k+1) = Ax(k) + Bu(k)$ .
6     %
7     % [Phi, Hqp, fqp, Hx, Px, Hu, Pu] = matricesMPC(A, B, Q, R, K, N)
8     %   A -> matriz dos estados
9     %   B -> matriz das entradas
10    %   Q -> matriz MPC de custos dos estados
11    %   R -> matriz MPC de custos das entradas
12    %   K -> vetor de ganhos do LQR
13    %   N -> horizonte de predicao
14    %
15    %   Phi = A - B*K
16    %   Hqp = Hx'*Q_*Hx + Hu'*R_*Hu
17    %   fqp = Px'*Q_*Hx + Pu'*R_*Hu
18    %   Hx -> formada por  $\Phi^i \cdot B$ . (i = 0 ... N-1)
19    %   Px -> empilhamento de  $\Phi^i$ . (i = 1 ... N)
20    %   Pu = Plqr + Hlqr*Px
21    %   Hu = Hlqr*Hx + eye(m*N)
22
23
24    %%
25    Phi = A - B*K;
26    [m, n] = size(Phi);
27    Px = zeros(m*N, n);
28    exp = 1;
29    for i = 1:m:m*N
30        Px(i:i+m-1, 1:n) = Phi^exp;
31        exp = exp + 1;
32    end
33
34    %%
35    [m, n] = size(B);
36    Hx = zeros(m*N, n*N);

```

```

37     Hx(1:m, 1:n) = B;
38     expr = 2;
39     for i = m+1:m:m*N
40         expc = 1;
41         for j = 1:n:n*N
42             Hx(i:i+m-1, j:j+n-1) = Phi^(expr-expc)*B; % Phi^exp*B
43             expc = expc + 1;
44             if (expr-expc) < 0
45                 break
46             end
47         end
48         expr = expr + 1;
49     end
50
51     %%
52     [m, n] = size(K);
53     Plqr = zeros(m*N, n);
54     Plqr(1:m, 1:n) = -K;
55
56     Hlqr = zeros(m*N, n*N);
57     j = 1;
58     for i = m+1:m:m*N
59         Hlqr(i:i+m-1, j:j+n-1) = -K;
60         j = j + n;
61     end
62
63     Pu = Plqr + Hlqr*Px;
64     Hu = Hlqr*Hx + eye(m*N);
65
66     %%
67     [m, n] = size(Q);
68     Q_ = zeros(m*N, n*N);
69     j = 1;
70     for i = 1:m:m*N
71         Q_(i:i+m-1, j:j+n-1) = Q;
72         j = j + n;
73     end
74
75     %%
76     [m, n] = size(R);
77     R_ = zeros(m*N, n*N);
78     j = 1;
79     for i = 1:m:m*N
80         R_(i:i+m-1, j:j+n-1) = R;
81         j = j + n;
82     end
83

```

```

84      %%
85      Hqp = Hx'*Q_*Hx + Hu'*R_*Hu;
86      fqp = Px'*Q_*Hx + Pu'*R_*Hu;
87  end

```

C.2 MAS

```

1  % arquivo: MAS.m
2
3  function [Aoinf, boinf, MASObject] = MAS(Phi, Ts, KLQR, xMax, xMin, ...
      uMax, uMin)
4      % Funcao que retorna as matrizes do MAS (Maximal Admissible Set) ...
      Aoinfo
5      % e boinf, em que
6      %      Aoinf*x(k) ≤ boinf.
7      % Sistema discreto representado no espaco de estados:
8      %      x(k+1) = A*x(k)+B*u(k)
9      %
10     % [Aoinf, boinf, MASObject] = MAS(Phi, Ts, KLQR, xMax, xMin, ...
      uMax, uMin)
11     %   Phi  -> matriz de malha fechada: Phi =  A - B*KLQR
12     %   Ts   -> [s] - tempo de amostragem
13     %   KLQR -> vetor de ganhos do LQR
14     %   xMax -> vetor coluna de limitantes superiores dos estados
15     %   xMin -> "      "      "      "      inferiores "      "
16     %   uMax -> vetor coluna de limitantes superiores das entradas
17     %   uMin -> "      "      "      "      inferiores "      "
18     %
19     %   Aoinf      = MASObject.A
20     %   boinf      = MASObject.B
21     %   MASObject -> invariantSet
22
23
24     %% Definindo matrizes que respeitam as restricoes impostas
25     % Ax*x(k) ≤ bx
26     [m, n] = size(KLQR);
27     Ax = [eye(n); -eye(n); -KLQR; KLQR];
28     bx = [xMax; -xMin; uMax; -uMin];
29
30     X = Polyhedron('H', [Ax bx]);
31     system = LTISystem('A', Phi, 'Ts', Ts);
32     MASObject = system.invariantSet('X', X);
33
34     %% Retornando matrizes do MAS

```

```
35     Aoinf = MASObject.A;  
36     boinf = MASObject.b;  
37 end
```

C.3 Duplo integrador

```
1 % arquivo: main_duplo_integrador.m  
2  
3 clear  
4 close all  
5 clc  
6  
7 %%  
8 % Implementacao de uma lei de controle preditivo em um duplo  
9 % integrador  
10 %  
11 % Implementacao de uma lei de controle do tipo  
12 %      $u(k) = u_{lqr}(k) + u_{mpc}(k)$ ,  
13 % em que  
14 %      $u_{LQR}(k) = -K_{lqr} * x(k)$   
15 %      $u_{MPC}(k)$  = e dado a partir da resolucao de um problema  
16 %         de otimizacao  
17  
18 %% Condicoes iniciais de simulacao  
19 x0 = [1 -1]';  
20 kMax = 500; % numero maximo de iteracoes  
21 Ts = 1e-2; % [s] -> periodo de amostragem  
22 N = 5; % horizonte de predicao  
23 xMax = [1.5 1.5]'; % limitantes superiores para os estados  
24 xMin = [-2 -2]'; % limitantes inferiores  
25 uMax = 10; % limitante superior para o controle  
26 uMin = -10; % limitante inferior  
27  
28 %% Scopes  
29 uLQR = zeros(1, kMax);  
30 uMPC = zeros(1, kMax);  
31 u = zeros(1, kMax); %  $u = u_{LQR} + u_{MPC}$   
32 x = zeros(2, kMax);  
33 x(:, 1) = x0; % condicao inicial  
34  
35 XMax = repmat(xMax, N, 1); % empilha as matrizes N vezes  
36 XMin = repmat(xMin, N, 1);  
37 UMax = repmat(uMax, N, 1);  
38 UMin = repmat(uMin, N, 1);
```

```

39 exitFlags = zeros(1, kMax); % flags da funcao quadprog
40
41 options = optimset('Display','off'); % desabilita logs da funcao ...
    quadprog
42
43 %% Obtendo K do LQR
44 QLQR = [100 0; 0 10]; % matriz de custo dos estados
45 RLQR = 1; % matriz de custo da entrada
46
47 % Matrizes do sistema
48 sysc = doubleIntegrator(); % tempo contínuo
49 sysd = c2d(sysc, Ts, 'zoh'); % tempo discreto
50
51 % Calculo do vetor ganhos
52 [KLQR, S, e] = dlqr(sysd.A, sysd.B, QLQR, RLQR);
53
54 %% Obtendo matrizes do MPC
55 QMPC = [1e6 0; 0 5e3]; % matriz de custo dos estados
56 RMPC = 1; % matriz de custo da entrada
57 [Phi, Hqp, fqp, Hx, Px, Hu, Pu] = matricesMPC(sysd.A, sysd.B, QMPC, ...
    RMPC, KLQR, N);
58
59
60 %% Matrizes de restricao dos estados, controle e terminal
61 disp('Determinando o MAS...')
62
63 % Matrizes que respeitam as restricoes do conjunto Oinf
64 [Aoinf, boinf, MASObject] = MAS(Phi, Ts, KLQR, xMax, xMin, uMax, uMin);
65
66 Hb = Phi^(N-1)*sysd.B;
67 for i = N-2:-1:0
68     Hb = horzcat(Hb, Phi^i*sysd.B);
69 end
70
71 Aoqp = Aoinf*Hb; % restricao terminal
72 Axqp = [Hx; -Hx]; % restricao de estados
73 Auqp = [Hu; -Hu]; % restricao de controle
74 Aqp = [Axqp; Auqp; Aoqp]; % resumo das restricoes
75
76 %% Definicao da regio de factibilidade
77 disp('Calculando regio de factibilidade...')
78
79 Af = [Px; -Px; Pu; -Pu; Aoinf*Phi^N];
80 Axu = [Aqp Af];
81 bxu = [xMax; -xMin; uMax; -uMin; boinf];
82
83 Theta = Polyhedron('H', [Axu bxu]);

```

```

84 ThetaProjection = Theta.projection(N+1:N+2);
85
86 %%
87 Hqp = (Hqp + Hqp')/2; % Para retirar esse warning na funcao quadprog ->
88 % Warning: Your Hessian is not symmetric. ...
89 % Resetting H=(H+H')/2.
90
91 %% Simulando acoes de controle
92 disp('Simulando acoes de controle...')
93
94 for k = 1:kMax
95     bxqp = [XMax; -XMin] + [-Px; Px]*x(:, k);
96     buqp = [UMax; -UMin] + [-Pu; Pu]*x(:, k);
97     boqp = boinf - Aoinf*Phi^N*x(:, k);
98     bqp = [bxqp; buqp; boqp]; % resumo das restricoes
99
100     [UMPC, fval, exitflag] = quadprog(2*Hqp, 2*x(:, k)'*fqp, Aqp, ...
101     bqp, [], [], [], [], [], options);
102     exitFlags(k) = exitflag;
103
104     uMPC(k) = UMPC(1); % acao do MPC
105     uLQR(k) = -KLQR*x(:, k); % acao do LQR
106     u(k) = uLQR(k) + uMPC(k); % controle total
107
108     x(:, k+1) = sysd.A*x(:, k) + sysd.B*u(k); % atualizando estados
109
110     fprintf('k: %d\n', k);
111 end
112
113 %% Resultados
114 disp('Apresentando resultados...')
115
116 % Configura as linhas de todas as figuras, exibicao dos simbolos em ...
117 % latex
118 % e tamanho de fonte
119 set(groot, 'defaultStairLineWidth', 1.4)
120 set(groot, 'defaultTextFontSize', 12)
121 set(groot, 'defaultLegendFontSize', 12)
122 set(groot, 'defaultTextInterpreter', 'latex')
123 set(groot, 'defaultLegendInterpreter', 'latex');
124
125 % Grafico de Theta (Regiao do controlador)
126 figure('Renderer', 'painters', 'Position', [100 100 800 300])
127 subplot(1, 2, 1)
128 ThetaProjection.plot('Color', [0, 0.7, 0.9])
129

```

```

128 % Grafico do MAS
129 hold on
130 MASObject.plot()
131
132 % Plotando estados no grafico do MAS
133 plot(x(1, 1:end-1), x(2, 1:end-1), 'y*')
134 ylabel('$x_2$')
135 xlabel('$x_1$')
136 grid on
137 legend('$\Theta$', '$\mathcal{O}_{\infty}$', 'FontSize', 12)
138
139 % Estados
140 subplot(1, 2, 2)
141 hold on
142 stairs(1:kMax, x(1, 1:end-1))
143 stairs(1:kMax, x(2, 1:end-1), '--')
144 ylabel('$x$')
145 xlabel('$k$')
146 grid on
147 legend('$x_1$', '$x_2$', 'FontSize', 12)
148
149 % Entradas
150 figure
151 hold on
152 stairs(1:kMax, uLQR, '--');
153 stairs(1:kMax, uMPC, '-. ');
154 stairs(1:kMax, u);
155 ylabel('$u$')
156 xlabel('$k$')
157 grid on
158 legend('$u_{lqr}$', '$u_{mpc}$', '$u = u_{lqr} + u_{mpc}$', ...
        'FontSize', 12)
159
160 % Flags de quadprog
161 figure
162 scatter(1:kMax, exitFlags, '.')
163 ylabel('Quadprog flags'), xlabel('$k$'), grid on

```

```

1 % arquivo: doubleIntegrator.m
2
3 function sysc = doubleIntegrator()
4     % Funcao que retorna as matrizes de estados do sistema a tempo
5     % continuo.
6     %
7     % sysc = doubleIntegrator()
8     % sysc -> sistema a tempo continuo

```

```
9
10     %% Matrizes da dinamica do sistema
11     A = [0 1
12          0 0];
13     B = [0
14          1];
15     C = [1 0];
16     D = 0;
17
18     %% Sistema no espaco de estados no tempo continuo
19     sysc = ss(A, B, C, D);
20 end
```

C.4 Modelo linear do pêndulo

```
1 % arquivo: main_pendulo_linear.m
2
3 clear
4 close all
5 clc
6
7 %%
8 % Implementacao de uma lei de controle preditivo nos modelos linear do
9 % pendulo invertido de duas rodas
10 %
11 % Implementacao de uma lei de controle do tipo
12 %      $u(k) = u_{LQR}(k) + u_{MPC}(k)$ 
13 % em que
14 %      $u_{LQR}(k) = -K_{LQR} \cdot x(k)$ 
15 %      $u_{MPC}(k)$  = calculado a partir da resolucao de um problema de ...
16 %     otimizacao
17
18 %% Condicoes iniciais de simulacao
19 x0 = [0 5*pi/180 0 0]'; % [rad] -> theta, psi, dotTheta, dotPsi
20 kMax = 500; % numero maximo de iteracoes da simulacao
21 Ts = 4e-3; % [s] -> periodo de amostragem
22 N = 10; % horizonte de predicao
23 xMax = [999 20*pi/180 8 999]'; % [rad] limitantes superiores para os ...
24 %     estados
25 xMin = -xMax; % [rad] -> limitantes inferiores
26 uMax = [8 8]'; % [V] -> limitantes superiores para o controle
27 uMin = -uMax; % [V] -> limitantes inferiores
28
```

```

28 %% Scopes
29 uLQR = zeros(2, kMax); % ve, vd
30 uMPC = zeros(2, kMax);
31 u     = zeros(2, kMax); % u = uLQR + uMPC
32 x     = zeros(4, kMax);
33 x(:, 1) = x0; % condicao inicial
34
35 XMax = repmat(xMax, N, 1); % empilha as matrizes N vezes
36 XMin = repmat(xMin, N, 1);
37 UMax = repmat(uMax, N, 1);
38 UMin = repmat(uMin, N, 1);
39 exitFlags = zeros(1, kMax); % flags da funcao quadprog
40
41 options = optimset('Display','off'); % desabilita logs da funcao ...
    quadprog
42
43 %% Obtendo K do LQR
44 % Matriz de custos dos estados
45 QLQR = [4.2e2 0      0 0
46         0      6.5e3 0 0
47         0      0      1 0
48         0      0      0 1];
49 % Matriz de custos das entradas
50 RLQR = 10*eye(2);
51
52 % Matrizes do sistema
53 sysc = linearPendulum(); % tempo continuo
54 sysd = c2d(sysc, Ts, 'zoh'); % tempo discreto
55
56 % Calculo do vetor ganhos
57 [KLQR, S, e] = dlqr(sysd.A, sysd.B, QLQR, RLQR);
58
59
60 %% Obtendo matrizes do MPC
61 QMPC = [1e2 0      0 0
62         0      4.5e6 0 0
63         0      0      1 0
64         0      0      0 1];
65 RMPC = 3.5*eye(2);
66
67 [Phi, Hqp, fqp, Hx, Px, Hu, Pu] = matricesMPC(sysd.A, sysd.B, QMPC, ...
    RMPC, KLQR, N);
68
69
70 %% Matrizes de restricoes dos estados, controle e terminal
71 disp('Determinando o MAS...')
72

```

```

73 [Aoinf, boinf, MASObject] = MAS(Phi, Ts, KLQR, xMax, xMin, uMax, uMin);
74
75 Hb = Phi^(N-1)*sysd.B;
76 for i = N-2:-1:0
77     Hb = horzcat(Hb, Phi^i*sysd.B);
78 end
79
80 Aoqp = Aoinf*Hb; % restricao terminal
81 Axqp = [Hx; -Hx]; % restricao de estados
82 Auqp = [Hu; -Hu]; % restricao de controle
83 Aqp = [Axqp; Auqp; Aoqp]; % resumo das restricoes
84
85 % Para uso da funcao de custo apresentada no Capitulo 4
86 Psi = RLQR + sysd.B'*S*sysd.B;
87 Psi_ = Psi;
88 for i = 1:N-1
89     Psi_ = blkdiag(Psi_, Psi);
90 end
91
92
93 %% Definicao da regio de factibilidade
94 % disp('Calculando regio Theta...')
95 % Af = [Px; -Px; Pu; -Pu; Aoinf*Phi^N];
96 % Axu = [Aqp Af];
97 % bxu = [XMax; -XMin; UMax; -UMin; boinf];
98 % Theta = Polyhedron('H', [Axu bxu]);
99 % ThetaProjection = Theta.projection(N+1:N+4);
100
101
102 %%
103 Hqp = (Hqp + Hqp')/2; % Para retirar esse warning na funcao quadprog ->
104 % Warning: Your Hessian is not symmetric. ...
105 % Resetting H=(H+H')/2.
106
107 %% Simulando acoes de controle
108 disp('Simulando acoes de controle...')
109
110 for k = 1:kMax
111     bxqp = [XMax; -XMin] + [-Px; Px]*x(:, k);
112     buqp = [UMax; -UMin] + [-Pu; Pu]*x(:, k);
113     boqp = boinf - Aoinf*Phi^N*x(:, k);
114     bqp = [bxqp; buqp; boqp]; % resumo das restricoes
115
116     % Calculo do MPC
117
118     % Formulacao do Capitulo 4

```

```

119     [UMPC, fval, exitflag] = quadprog(2*Psi_, [], Aqp, bqp, [], [], ...
120         [], [], [], options);
121     % Formulacao do Capitulo 2
122     %[UMPC, fval, exitflag] = quadprog(2*Hqp, 2*x(:, k)'\*fqp, Aqp, ...
123         bqp, [], [], [], [], [], options);
124     exitFlags(k) = exitflag;
125     uMPC(:, k) = UMPC(1:2); % acao do MPC
126     uLQR(:, k) = -KLQR*x(:, k); % acao do LQR
127     u(:, k) = uLQR(:, k) + uMPC(:, k); % controle total
128
129     x(:, k+1) = sysd.A*x(:, k) + sysd.B*u(:, k); % atualizando estados
130
131     fprintf('k: %d\n', k);
132 end
133
134 %% Resultados
135 disp('Apresentando resultados...')
136
137 % Configura as linhas de todas as figuras, exibicao dos simbolos em ...
138 % e tamanho de fonte
139 set(groot, 'defaultStairLineWidth', 1.4)
140 set(groot, 'defaultTextFontSize', 12)
141 set(groot, 'defaultLegendFontSize', 12)
142 set(groot, 'defaultTextInterpreter', 'latex')
143 set(groot, 'defaultLegendInterpreter', 'latex');
144
145 % Estados
146 lineStyle = {'-', '--', '-.', ':'};
147 figure('Renderer', 'painters', 'Position', [288 210 800 300])
148 subplot(1, 2, 1)
149 hold on
150 for i = 1:4
151     stairs(1:kMax, x(i, 1:end-1)*180/pi, lineStyle{i})
152 end
153 ylabel('$x$')
154 xlabel('$k$')
155 grid on
156 legend('$x_1=\theta \ (\ ^{\circ})$', '$x_2=\psi \ (\ ^{\circ})$', ...
157     '$x_3=\dot{\theta} \ (\ ^{\circ}/s)$', '$x_4=\dot{\psi} \ (\ ^{\circ}/s)$', ...
158     'FontSize', 12)
159
160 % Grafico de psi
161 subplot(1, 2, 2)

```

```

162 stairs(1:kMax, x(2, 1:end-1)*180/pi, '--')
163 ylabel('$\psi \ (\^\circ)$')
164 xlabel('$k$')
165 grid on
166
167
168 % Entradas
169 figure
170 hold on
171 stairs(1:kMax, uLQR(1, :))
172 stairs(1:kMax, uMPC(1, :), '--')
173 stairs(1:kMax, u(1, :), '-.')
174 ylabel('$u$ (V)')
175 xlabel('$k$')
176 grid on
177 legend('$u_{lqr}$', '$u_{mpc}$', '$u$', 'FontSize', 12)
178
179
180 % Flags de quadprog
181 figure
182 scatter(1:kMax, exitFlags, '.')
183 ylabel('Quadprog flags'), xlabel('$k$'), grid on

```

```

1 % arquivo: linearPendulum.m
2
3 function sysc = linearPendulum()
4     % Funcao que retorna as matrizes de estados do sistema a tempo ...
5     % a partir do modelo linearizado da planta.
6     %
7     % sysc = linearPendulum()
8     % sysc -> sistema a tempo contínuo
9
10
11 %% Parametros da planta
12 g = 9.81; % [m/s^2] -> aceleracao da gravidade
13 m = 0.03; % [kg] -> massa das rodas
14 R = 0.04; % [m] -> raio das rodas
15 M = 0.6; % [kg] -> massa do centro de massa
16 L = 0.072; % [m] -> distancia do centro de massa ao eixo de ...
17     % giro da roda
18 Jpsi = 0.001; % [kg.m^2] -> momento de inercia em relacao ao Psi
19 Jm = 1e-5; % [kg.m^2] -> momento de inercia do motor DC
20 Rm = 6.69; % [Ohm] -> resistencia eletrica do motor
21 Kb = 0.468; % [V.s/rad] -> cte de foraca eletromotriz do motor
22 Kt = 0.317; % [N.m/A] -> cte de torque do motor

```

```

22     fm    = 0.0022; % [N.m.rad/s]-> coeficiente de friccao do motor
23     Vb    = 8.00;   % [V]          -> tensao de alimentacao
24     Vo    = 0.625;  % [V]          -> offset de tensao
25     mi    = 1.089;  % []           -> fator de ganho da alimentacao
26     Gu    = 1e-2;   % []           -> fator de ganho do PWM
27     n     = 1;      % []           -> gearbox ratio
28     fw    = 0;      % []           -> coeficiente de atrito entre a ...
        roda e o chao.
29
30     %% Constantes a serem utilizadas nas equacoes
31     alpha = n*Kt/Rm;
32     beta  = n*Kt*Kb/Rm + fm;
33
34     c1 = (2*m + M)*R^2 + 2*n^2*Jm;
35     c2 = M*L*R;
36     c3 = 2*n^2*Jm;
37     c4 = M*g*L;
38     c5 = M*L^2 + Jpsi + 2*n^2*Jm;
39
40     %% Processo de linearizacao (Apendice B)
41     df1dx1 = 0;
42     df1dx2 = 0;
43     df1dx3 = 1;
44     df1dx4 = 0;
45
46     df2dx1 = 0;
47     df2dx2 = 0;
48     df2dx3 = 0;
49     df2dx4 = 1;
50
51     df3dx1 = 0;
52     df3dx2 = -c4*(c2 - c3)/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
53     df3dx3 = -2*(beta*(c2 - c3 + c5) + c5*fw)/(-c2^2 + 2*c2*c3 - ...
        c3^2 + c1*c5);
54     df3dx4 = 2*beta*(c2 - c3 + c5)/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
55
56     df4dx1 = 0;
57     df4dx2 = c1*c4/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
58     df4dx3 = 2*(beta*(c1 + c2 - c3) + fw*(c2 - c3))/(-c2^2 + 2*c2*c3 ...
        - c3^2 + c1*c5);
59     df4dx4 = -2*beta*(c1 + c2 - c3)/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
60
61     df1du1 = 0;
62     df1du2 = 0;
63
64     df2du1 = 0;
65     df2du2 = 0;

```

```

66
67     df3du1 = alpha*(c2 - c3 + c5)/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
68     df3du2 = df3du1;
69
70     df4du1 = -alpha*(c1 + c2 - c3)/(-c2^2 + 2*c2*c3 - c3^2 + c1*c5);
71     df4du2 = df4du1;
72
73     %% Matrizes da dinamica do sistema
74     A = [df1dx1 df1dx2 df1dx3 df1dx4
75          df2dx1 df2dx2 df2dx3 df2dx4
76          df3dx1 df3dx2 df3dx3 df3dx4
77          df4dx1 df4dx2 df4dx3 df4dx4];
78
79     B = [df1du1 df1du2
80          df2du1 df2du2
81          df3du1 df3du2
82          df4du1 df4du2];
83
84     C = eye(4);
85
86     D = 0;
87
88     %% Sistema no espaco de estados a tempo continuo
89     sysc = ss(A, B, C, D);
90 end

```

C.5 Modelo não linear do pêndulo

```

1 % arquivo: main_pendulo_naolinear.m
2
3 clear
4 close all
5 clc
6
7 %%
8 % Implementacao de uma lei de controle preditivo no modelo nao ...
   linear do
9 % pendulo invertido de duas rodas
10 %
11 % Implementacao de uma lei de controle do tipo
12 %     u(k) = uLQR(k) + uMPC(k)
13 % em que
14 %     uLQR(k) = -KLQR*x(k)
15 %     uMPC(k) = calculado a partir da resolucao de um problema de

```

```

16 %                                otimizacao
17
18
19 %% Condições iniciais de simulacao
20 x0 = [0 15*pi/180 0 0]'; % [rad] -> theta, psi, dotTheta, dotPsi
21 kMax = 1000; % numero maximo de iteracoes da simulacao
22 Ts = 4e-3; % [s] -> periodo de amostragem
23 N = 25; % horizonte de predicao
24 xMax = [999 20*pi/180 8 999]'; % [rad] limitantes superiores para os ...
    estados
25 xMin = -xMax; % [rad] -> limitantes inferiores
26 uMax = [8 8]'; % [V] -> limitantes superiores para o controle
27 uMin = -uMax; % [V] -> limitantes inferiores
28
29 %% Scopes
30 uLQR = zeros(2, kMax); % ve, vd
31 uMPC = zeros(2, kMax);
32 u = zeros(2, kMax); % u = uLQR + uMPC
33 x = zeros(4, kMax);
34 x(:, 1) = x0; % condicao inicial
35
36 XMax = repmat(xMax, N, 1); % empilha as matrizes N vezes
37 XMin = repmat(xMin, N, 1);
38 UMax = repmat(uMax, N, 1);
39 UMin = repmat(uMin, N, 1);
40 exitFlags = zeros(1, kMax); % flags da funcao quadprog
41
42 options = optimset('Display','off'); % desabilita logs da funcao ...
    quadprog
43
44 %% Obtendo K do LQR
45 % Matriz de custos dos estados
46 QLQR = [4.2e2 0 0 0
47         0 6.5e3 0 0
48         0 0 1 0
49         0 0 0 1];
50 % Matriz de custos das entradas
51 RLQR = 10*eye(2);
52
53 % Matrizes do sistema
54 sysc = linearPendulum(); % tempo contínuo
55 sysd = c2d(sysc, Ts, 'zoh'); % tempo discreto
56
57 % Calculo do vetor ganhos
58 [KLQR, S, e] = dlqr(sysd.A, sysd.B, QLQR, RLQR);
59
60

```

```

61 %% Obtendo matrizes do MPC
62 QMPC = [1e2 0      0 0
63         0   4.5e6 0 0
64         0    0    1 0
65         0    0    0 1];
66 RMPC = 3.5*eye(2);
67
68 [Phi, Hqp, fqp, Hx, Px, Hu, Pu] = matricesMPC(sysd.A, sysd.B, QMPC, ...
        RMPC, KLQR, N);
69
70
71 %% Matrizes de restricoes dos estados, controle e terminal
72 disp('Determinando o MAS...')
73
74 [Aoinf, boinf, MASObject] = MAS(Phi, Ts, KLQR, xMax, xMin, uMax, uMin);
75
76 Hb = Phi^(N-1)*sysd.B;
77 for i = N-2:-1:0
78     Hb = horzcat(Hb, Phi^i*sysd.B);
79 end
80
81 Aoqp = Aoinf*Hb; % restricao terminal
82 Axqp = [Hx; -Hx]; % restricao de estados
83 Auqp = [Hu; -Hu]; % restricao de controle
84 Aqp = [Axqp; Auqp; Aoqp]; % resumo das restricoes
85
86
87 % Para uso da funcao de custo apresentada no Capitulo 4
88 Psi = RLQR + sysd.B'*S*sysd.B;
89 Psi_ = Psi;
90 for i = 1:N-1
91     Psi_ = blkdiag(Psi_, Psi);
92 end
93
94
95 %% Definicao da regio de factibilidade
96 % disp('Calculando regio Theta...')
97 % Af = [Px; -Px; Pu; -Pu; Aoinf*Phi^N];
98 % Axu = [Aqp Af];
99 % bxu = [XMax; -XMin; UMax; -UMin; boinf];
100 % Theta = Polyhedron('H', [Axu bxu]);
101 % ThetaProjection = Theta.projection(N+1:N+4);
102
103
104 %%
105 Hqp = (Hqp + Hqp')/2; % Para retirar esse warning na funcao quadprog ->

```

```

106                                     % Warning: Your Hessian is not symmetric. ...
                                     Resetting H=(H+H')/2.
107
108 %% Simulando acoes de controle
109 disp('Simulando acoes de controle...')
110
111 for k = 1:kMax
112
113     x_med = x(:, k) + (-1 + 2*rand(4, 1))*pi/180; % simulando sinal ...
               com ruido (-1 a 1 deg)
114
115     bxqp = [XMax; -XMin] + [-Px; Px]*x_med;
116     buqp = [UMax; -UMin] + [-Pu; Pu]*x_med;
117     boqp = boinf - Aoinf*Phi^N*x_med;
118     bqp = [bxqp; buqp; boqp]; % resumo das restricoes
119
120     % Calculo do MPC
121
122     % Formulacao do Capitulo 4
123     [UMPC, fval, exitflag] = quadprog(2*Psi_, [], Aqp, bqp, [], [], ...
               [], [], [], options);
124
125     % Formulacao do Capitulo 2
126     %[UMPC, fval, exitflag] = quadprog(2*Hqp, 2*x(:, k)'*fqp, Aqp, ...
               bqp, [], [], [], [], [], options);
127     exitFlags(k) = exitflag;
128
129     uMPC(:, k) = UMPC(1:2); % acao do MPC
130     uLQR(:, k) = -KLQR*x_med; % acao do LQR
131     u(:, k) = uLQR(:, k) + uMPC(:, k); % controle total
132
133     % u(:, k)*0.9 -> para simular um descasamento de modelo. Apenas ...
               90 % da
134     % tensao eh de fato aplicada aos motores
135     [t, x_] = ode45(@(t, x_) nonLinearPendulum(t, x_, u(:, k)*0.9), ...
               [0 Ts], x(:, k)); % evoluindo dinamica da planta
136     x(:, k+1) = x_(end, :); % atualizando estados
137
138     fprintf('k: %d\n', k);
139 end
140
141 %% Resultados
142 disp('Apresentando resultados...')
143
144 % Configura as linhas de todas as figuras, exibicao dos simbolos em ...
               latex
145 % e tamanho de fonte

```

```

146 set(groot, 'defaultStairLineWidth', 1.4)
147 set(groot, 'defaultTextFontSize', 12)
148 set(groot, 'defaultLegendFontSize', 12)
149 set(groot, 'defaultTextInterpreter', 'latex')
150 set(groot, 'defaultLegendInterpreter', 'latex');
151
152 % Estados
153 lineStyle = {'-', '--', '-.', ':'};
154 figure('Renderer', 'painters', 'Position', [288 210 800 300])
155 subplot(1, 2, 1)
156 hold on
157 for i = 1:4
158     stairs(1:kMax, x(i, 1:end-1)*180/pi, lineStyle{i})
159 end
160 ylabel('$x$')
161 xlabel('$k$')
162 grid on
163 legend('$x_1=\theta \ (\ ^{\circ})$', '$x_2=\psi \ (\ ^{\circ})$', ...
164     '$x_3=\dot{\theta} \ (\ ^{\circ}/s)$', '$x_4=\dot{\psi} \ (\ ^{\circ}/s)$', ...
165     'FontSize', 12)
166
167 % Grafico de psi
168 subplot(1, 2, 2)
169 stairs(1:kMax, x(2, 1:end-1)*180/pi, '--')
170 ylabel('$\psi \ (\ ^{\circ})$')
171 xlabel('$k$')
172 grid on
173
174
175 % Entradas
176 figure
177 hold on
178 stairs(1:kMax, uLQR(1, :))
179 stairs(1:kMax, uMPC(1, :), '--')
180 stairs(1:kMax, u(1, :), '-.')
181 ylabel('$u$ (V)')
182 xlabel('$k$')
183 grid on
184 legend('$u_{lqr}$', '$u_{mpc}$', '$u$', 'FontSize', 12)
185
186
187 % Flags de quadprog
188 figure
189 scatter(1:kMax, exitFlags, '.')
190 ylabel('Quadprog flags'), xlabel('$k$'), grid on

```

```

1 % arquivo: nonLinearPendulum.m
2
3 function dotx = nonLinearPendulum(t, x, u)
4     % Funcao da dinamica nao linear do sistema a tempo continuo no ...
5     espaco
6     %
7     % dotx = nonLinearPendulum(t, x, u)
8     % dotx -> vetor (4x1) de derivadas dos estados
9     % t -> tempo (nao utilizado na funcao)
10    % x -> vetor (4x1) de estados
11    % u -> vetor (1x2) de entrada
12
13
14    %% Parametros da planta
15    g = 9.81; % [m/s^2] -> aceleracao da gravidade
16    m = 0.03; % [kg] -> massa das rodas
17    R = 0.04; % [m] -> raio das rodas
18    M = 0.6; % [kg] -> massa do centro de massa
19    L = 0.072; % [m] -> distancia do centro de massa ao eixo de ...
20    giro da roda
21    Jpsi = 0.001; % [kg.m^2] -> momento de inercia em relacao ao Psi
22    Jm = 1e-5; % [kg.m^2] -> momento de inercia do motor DC
23    Rm = 6.69; % [Ohm] -> resistencia eletrica do motor
24    Kb = 0.468; % [V.s/rad] -> cte de foraca eletromotriz do motor
25    Kt = 0.317; % [N.m/A] -> cte de torque do motor
26    fm = 0.0022; % [N.m.rad/s] -> coeficiente de friccao do motor
27    Vb = 8.00; % [V] -> tensao de alimentacao
28    Vo = 0.625; % [V] -> offset de tensao
29    mi = 1.089; % [] -> fator de ganho da alimentacao
30    Gu = 1e-2; % [] -> fator de ganho do PWM
31    n = 1; % [] -> gearbox ratio
32    fw = 0; % [] -> coeficiente de atrito entre a ...
33    roda e o chao.
34
35    %% Constantes a serem utilizadas nas equacoes
36
37    alpha = n*Kt/Rm;
38    beta = n*Kt*Kb/Rm + fm;
39
40    c1 = (2*m + M)*R^2 + 2*n^2*Jm;
41    c2 = M*L*R;
42    c3 = 2*n^2*Jm;
43    c4 = M*g*L;
44    c5 = M*L^2 + Jpsi + 2*n^2*Jm;
45
46    h1 = (c2*cos(x(2)) - c3)/c5;
47    h2 = (c2*cos(x(2)) - c3)/c1;

```

```

45
46     %% Vetor de estados
47     dotx = zeros(4, 1);
48
49     % x(1) = theta
50     % x(2) = psi
51     % x(3) = derivada de theta
52     % x(4) = derivada de psi
53
54     dotx(1) = x(3);
55     dotx(2) = x(4);
56     dotx(3) = (1/(c1 - h1*(c2*cos(x(2))-c3)))*( -2*(beta*(1 + h1) + ...
57         fw)*x(3) + 2*beta*(1 + h1)*x(4) ...
58         + (c2*x(4)^2 - h1*c4)*sin(x(2)) ...
59         + alpha*(1 + h1)*(u(1) + ...
60         u(2)) );
61     dotx(4) = (1/(c5 - h2*(c2*cos(x(2))-c3)))*( 2*(beta + h2*(beta + ...
62         fw))*x(3) - 2*beta*(1 + h2)*x(4) ...
63         + (c4 - h2*c2*x(4)^2)*sin(x(2)) ...
64         - alpha*(1 + h2)*(u(1) + ...
65         u(2)) );
66 end

```

C.6 Linearização da dinâmica do pêndulo

```

1 % arquivo: linearizacao.mlx
2
3 close
4 clear
5 clc
6
7 % Turn off abbreviated output format
8 sympref('AbbreviateOutput',false);
9
10 % definindo variaveis de estado e entradas
11 syms x_1 x_2 x_3 x_4 u_1 u_2 real
12
13 % definindo parametros da planta
14 syms m M R n J_m L g J_psi beta f_w alpha real
15
16 % definindo constantes
17 syms c1 c2 c3 c4 c5 real
18
19 % definindo funcoes

```

```

20 h1 = (c2*cos(x_2) - c3)/c5
21 h2 = (c2*cos(x_2) - c3)/c1
22
23 % definindo equacoes de estado
24 dot_x1 = x_3
25 dot_x2 = x_4
26 dot_x3 = (1/(c1 - h1*(c2*cos(x_2) - c3)))*(-2*(beta*(1 + h1) + ...
    f_w)*x_3 ...
27     + 2*beta*(1 + h1)*x_4 + (c2*x_4^2 - h1*c4)*sin(x_2) ...
28     + alpha*(1 + h1)*(u_1 + u_2))
29 dot_x4 = (1/(c5 - h2*(c2*cos(x_2) - c3)))*(2*(beta + h2*(beta + ...
    f_w))*x_3 ...
30     - 2*beta*(1 + h2)*x_4 + (c4 - h2*c2*x_4^2)*sin(x_2) ...
31     - alpha*(1 + h2)*(u_1 + u_2))
32
33 % definindo as funcoes
34 f1 = dot_x1;
35 f2 = dot_x2;
36 f3 = dot_x3
37 f4 = dot_x4
38
39 % definindo um vetor de estados e entrada
40 x = [x_1, x_2, x_3, x_4]
41 u = [u_1, u_2]
42
43 % condicao normal de operacao
44 x0 = [0, 0, 0, 0]; % theta, psi, dot_theta, dot_psi
45 u0 = [0, 0];      % vl, vr
46
47 % calculando as derivadas parciais
48 df1dx1 = diff(f1, x_1)
49 df1dx2 = diff(f1, x_2)
50 df1dx3 = diff(f1, x_3)
51 df1dx4 = diff(f1, x_4)
52
53 df2dx1 = diff(f2, x_1)
54 df2dx2 = diff(f2, x_2)
55 df2dx3 = diff(f2, x_3)
56 df2dx4 = diff(f2, x_4)
57
58 % e substituindo o valores das condicao normal de operacao
59 df3dx1 = subs(diff(f3, x_1), [x, u], [x0, u0])
60 df3dx2 = simplifyFraction(subs(diff(f3, x_2), [x, u], [x0, u0]))
61 df3dx3 = simplifyFraction(subs(diff(f3, x_3), [x, u], [x0, u0]))
62 df3dx4 = simplifyFraction(subs(diff(f3, x_4), [x, u], [x0, u0]))
63
64 df4dx1 = subs(diff(f4, x_1), [x, u], [x0, u0])

```

```
65 df4dx2 = simplifyFraction(subs(diff(f4, x_2), [x, u], [x0, u0]))
66 df4dx3 = simplifyFraction(subs(diff(f4, x_3), [x, u], [x0, u0]))
67 df4dx4 = simplifyFraction(subs(diff(f4, x_4), [x, u], [x0, u0]))
68
69 df1du1 = diff(f1, u_1)
70 df1du2 = diff(f1, u_2)
71 df2du1 = diff(f2, u_1)
72 df2du2 = diff(f2, u_2)
73
74 df3du1 = simplifyFraction(subs(diff(f3, u_1), [x, u], [x0, u0]))
75 df3du2 = simplifyFraction(subs(diff(f3, u_2), [x, u], [x0, u0]))
76 df4du1 = simplifyFraction(subs(diff(f4, u_1), [x, u], [x0, u0]))
77 df4du2 = simplifyFraction(subs(diff(f4, u_2), [x, u], [x0, u0]))
```