UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE INFORMÁTICA

REINFORCEMENT LEARNING ON CONTROL SYSTEMS WITH UNOBSERVED STATES.

THESIS

A thesis submitted in partial fulfilment of the requirements for the degree of

DOCTOR EN INGENIERÍA INFORMÁTICA

by

MIGUEL ANDRÉS SOLÍS CID

In Valparaíso, Chile

June, 2017.

DISSERTATION TITLE:

REINFORCEMENT LEARNING ON CONTROL SYSTEMS WITH UNOBSERVED STATES

AUTHOR:

Miguel Andrés Solís Cid.

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor en Ingeniería Informática in Universidad Técnica Federico Santa María.

Dr. Héctor Allende _____

Dr. Manuel Olivares _____

Dr. Ricardo Ñanculef _____

Dr. Daniel Sbarbaro _____

Dr. Álvaro Pardo _____

Dr. Hernán Astudillo _____

Valparaíso, June 2017.

The thesis contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. I give consent to this copy of my thesis, when deposited in the University Library, being made available for loan and photocopying subject to the provisions of this entity.

RESUMEN

En el problema de hacer que un sistema de control siga una trayectoria dada, el principal objetivo es encontrar una señal de control apropiada (o acciones que el controlador debería ser capaz de ejecutar) para hacer que la variable a controlar del sistema siga los cambios realizados por el valor de referencia que se desea imitar. Existen dos opciones para abordar el problema: pudiendo ser tomado desde un enfoque basado en modelos paramétricos y un segundo enfoque adaptivo sin considerar la descripción detallada de la dinámica del sistema, donde la primera opción enunciada corresponde al enfoque clásico de la teoría de control, el cual contiene la natural desventaja que requiere de un conocimiento acabado del sistema, lo que es difícil de satisfacer debido a no-linealidades que deben ser aproximadas.

En el enfoque basado en modelos, se hace uso de un lazo de control realimentado por el estado del sistema a controlar. Dado que se desea mantener la complejidad estructural del controlador lo más baja posible, debido a que a mayor complejidad se requiere de un mayor esfuerzo por parte del diseñador o de la unidad de cómputo, es que habitualmente se utiliza una realimentación de estado de un grado de libertad. Sin embargo, el aumento de complejidad al añadir más grados de libertad al sistema de control no representa un problema cuando se considera la utilización de técnicas de aprendizaje automático para la sintonización de los parámetros correspondientes.

En esta tesis se propone el uso de una estructura alternativa para un controlador con realimentación de estado, agregando un grado de libertad adicional, lo que implica la incorporación de nuevos parámetros a considerar en la tarea de diseño, el que es abordado desde un enfoque de aprendizaje reforzado para así sintonizar los parámetros de manera automática. Los resultados obtenidos consideran la implementación de la propuesta en un entorno simulado y en el prototipo físico de un sistema subactuado. En dicho estudio se considera el péndulo rotatorio de Furuta, con resultados que muestran ventajas en la minimización de la función de costo en términos de la amplitud de la señal de control, obteniendo un desempeño mejor o igual en cuanto a la minimización del error de seguimiento de la trayectoria de referencia.

Palabras Clave:

aprendizaje reforzado, control adaptivo, iteración de política, péndulo de furuta, realimentación de estado

ABSTRACT

In the problem of making a control system that follows a given trajectory, the main objective is to find an appropriate control signal (or actions that the controller should be able to carry out) to ensure that the system variable for control follows the changes carried out by the reference value that one wishes to imitate. There are two options to tackle the problem: adopting an approach based on parametric models; or an adaptive approach without considering the detailed description of the system dynamics, where the former corresponds to the classic approach to control theory, which has the natural disadvantage of requiring a full knowledge of the system, which is difficult to fulfil due to nonlinear systems that must be approximated.

In the approach based on models, use is made of a control loop fed back by the state of the system to be controlled. Given the desire to maintain the controller's structural complexity as low as possible, since higher complexity requires more effort from the designer or the computing unit, usually a state feedback of a degree of freedom is used. However, the increase in complexity in adding more degrees of freedom to the control system does not represent a problem when one considers the use of automatic learning techniques for the tuning of the corresponding parameters.

In this thesis, the use of an alternative structure for a controller with state feedback is proposed, adding an additional degree of freedom, which implies the incorporation of new parameters to be considered in the task design, which is tackled from a reinforcement learning approach, in order to tune the parameters automatically. The results obtained consider the implementation of the proposal in a simulated environment and in the physical prototype of an underactuated system. The study considers the rotational (Furuta) pendulum, with results that reveal advantages with the minimisation of cost function in terms of the range of the control signal, obtaining a better or equal performance with regard to tracking error minimization when following the reference trajectory.

Keywords:

adaptive control, Furuta pendulum, policy iteration, reinforcement learning, state feedback

SYMBOLS AND ACRONYMS

The following is a list of symbols and acronyms commonly used throughout this thesis. Further details in some definitions will be given on its appropiate moment.

ARE	Algebraic Riccati Equation
BIBO	Bounded-Input Bounded-Output
i.i.d.	identically distributed and independent
LQR	Linear Quadratic Regulation
LQT	Linear Quadratic Tracking
MDP	Markovian Decision Process
MIMO	Multi-Input Multi-Output
MMSE	Minimum Mean Square Error
MSS	Mean Square Stable
PDF	Probability Density Function
PI	Policy Iteration
POMDP	Partially Observable Markovian Decision Process
RL	Reinforcement Learning
TD	Temporal Difference
VI	Value Iteration
WSS	Wide-Sense Stationary

$eig(A)$	Eigenvalues of matrix A
$det(A)$	Determinant of matrix A
$tr\{A\}$	Trace of matrix A
A^{-1}	(Multiplicative) Inverse of A
$A^{\intercal}$	Transpose of matrix A
$E_F[x]$	Expectation of stochastic variable x from a probability distribution F
$\mathbb{N}_0^+$	Set of (positive) natural numbers, including zero
$Prob\{\cdot\}$	Probability of $\cdot$

CONTENTS

Chapter 1

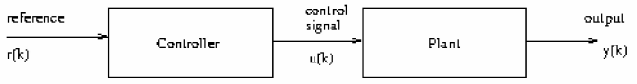
INTRODUCTION

When tackling the problem of making a system follow a given trajectory, the main objective is to find the appropiate control signals (or actions that should be generated) for making a variable of the system to be controlled to keep track of the desired reference value. To this end, and according to Figure 1.1, a **plant** is defined as the process being observed (or not) in order to change its output through an appropiate signal sent from the **controller**.

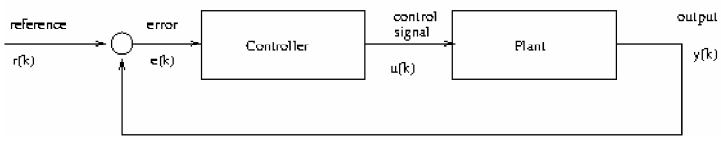
Tipically, there are two architectures or control loops, shown in Figure 1.1

- Open-loop control: the controller is not able to measure the current output of the plant, so it is not possible to correct this signal and change its behavior as desired. Also, there is no feedback, so it is not possible to stabilize this plant. Moreover, the model on which the design of the controller has been based must be a very good representation of the plant, since disturbances are negligible.
- Closed-loop control: the controller measures the output of the plant, and use it for influencing the control signal. In this architecture, several approaches can be found, as state feedback or output feedback which could be specially useful when there are states not availables.

Then, according to the notation introduced in Figure 1.1, a control signal $u[k]$ must be generated and used as input of the controlled system, which has an output $y(k)$ that should



(a) Open loop scheme



(b) Closed loop architecture

Figure 1.1: Open and closed loop comparison

keep track of the reference $r[k]$, which is minimizing control error $e[k]$ at time k. Therefore, a model is needed.

One of the typical options is to assume a given structure for the model, and then tune its parameters until the model and real system dynamics match. Another option, could be to use physical laws for building a model and set relations between all the variables of the system.

Both areas tackled in this thesis, Reinforcement Learning and classical Control Theory assume there is a system description based on states, control variables (actions) and a model which describes the state transitions. Nevertheless, the main disadvantage of optimal control theory approaches, is that there is a perfect model assumption which is naturally hard to satisfy due to non-linearities that have to be approximated.

Reinforcement Learning (RL) implies a cause and effect relationship between actions and reinforcement signals given as reward or punishment. It involves a goal directed behavior at least while the agent has an understanding of reward versus lack of reward or punishment. The RL algorithms are built on the idea that successful control decisions should be remembered, by means of a reinforcement signal, such that they become more likely to be used at subsequent times. Although the idea comes from experimental animal learning, RL is strongly connected from a theoretical point of view with direct and indirect adaptive optimal control methods. On the other hand, Feedback Control Theory could be defined as the study of alternatives of developing control systems with guaranteed performance and/or safety. Applications could be found in aircrafts, industrial processes, robotics and many more. It is often of interest to mimic nature and design control systems that are optimal in some sense of effectively achieving required performance without using excessive amounts of resources.

The first RL applications on control systems have been found in [44, 45], where the regulation problem was tackled, whose objective is to design a controller for a given process, such that the internal state of this process approaches zero as time increases. Then, an immediate extension was to apply policy iteration (PI) algorithms to solve the linear quadratic regulator (LQR) problem [11].

The LQR, i.e., the regulator problem when the system is assumed to be linear, and the performance index is given in terms of a quadratic function [2], is particularly appealing given that its solution is obtained by solving an algebraic Riccati equation (ARE). Then, PI algorithms basically start with an admissible control policy and then iterate between policy evaluation and policy improvement steps until variations on the policy or the specified value function are negligible, as seen in [11, 26, 41].

On the other hand, the linear quadratic tracking (LQT) problem also assumes a linear model for the process dynamics and a quadratic function for the performance index, but the main objective is to design a controller such that the measured output of the process to be controlled, follows an exogenous reference signal, so the LQR could be considered as a particular case of the LQT problem. Although, as mentioned before, RL algorithms have been extensively applied for solving the LQR problem, the LQT has not received much attention in the literature mainly because for most reference signals the infinite horizon cost becomes unbounded [7]. Work in [30] tackles the problem in the continuous time domain by solving an augmented ARE obtained from the original system dynamics and the reference

trajectory dynamics, while [22] takes a similar approach for the discrete-time case, where a Q-learning algorithm is obtained for solving the LQT problem without any model knowledge.

Then, when considering noisy systems, the performance index and notions of stability have to be modified accordingly. This problem has been extensively treated in literature from the classical control, or model-based approach [13, 18, 47], unlike on the learning paradigm. Work in [24] used neural networks for reducing calculus efforts on providing optimal control for the stochastic LQR, while other works focus on relaxing assumptions in the ARE under different scenarios, but still requiring knowledge of the system dynamics [12, 49].

Work in [22] could be considered as the closest to our approach, given the LQT setup and the absence of model knowledge. Nevertheless, unlike the work therein, we consider the stochastic LQT problem, and we extend the structure of the (linear) state feedback controller to be of a more general form. Then, when analyzing experimental results, RL will prove to be specially useful for the case when the model is unknown, but it is still useful when dynamics are assumed to be given, since hand-tuning of controller parameters could represent a time consuming task due to their number of degrees of freedom and their constraints.

This thesis tackles the trajectory tracking problem in control systems assuming there exists some (sub)-optimal policy in terms of the corresponding (state) value function, by using statistically guided exploration reinforcement learning techniques including a priori knowledge, in terms of partial model or structural knowledge, whose convergence solves the reference trajectory tracking problem for a given class of non-linear and unstable systems through state-feedback controllers with partial information and indirect measurements.

This thesis aims to study the characteristics of a control system that would allow the design of a stabilizing controller, through reinforcement learning techniques, without requiring full knowledge of the process dynamics.

In particular, the problems to be tackled would be arise from

- Obtaining conditions which allows modeling the reference tracking problem in control theory, for an unstable plant (with unobserved states) through a MDP.
- Analyzing which forms of adding a-priori knowledge would be better for a given problem. This knowledge could be expressed in terms of the structure of the controller, or partial knowledge about model parameters.
- To design a method to find a policy for a reinforcement learning problem modelled by a MDP of a system with unknown dynamics.
- The analysis is obtained in real system applications, like the Furuta Pendulum, an underactuated system whose control signal performs movements on the horizontal plane for stabilizing an inverted pendulum on the vertical plane.

1.1 Overview of Thesis Contents and Contributions

This thesis is organized in 6 chapters (including the present one) and one appendix. The most important contribution of this thesis is included in Chapter 4, which corresponds to solving the Linear Quadratic Tracking problem, namely, the control problem where the

output has to keep track of the trajectory of some exogenous reference signal assuming a linear model and a quadratic cost function. To this end, a (linear) dynamic state feedback controller is used, whose parameters are chosen by means of applying reinforcement learning techniques, proving to be specially useful when the model of the plant to be controlled is unknown or inaccurate, but still useful when dynamics are assumed to be given, since hand-tuning of controller parameters could represent a time consuming task due to its number of degrees of freedom and constraints.

In particular, content and contributions of each chapter can be summarized as follows [1]

- Chapter 2: This is a review chapter where the notation and conventions used throughout the thesis are presented. This chapter also shows known results regarding the stability of stochastic control systems.
- Chapter 3: This chapter deals with design and stability analysis of state-feedback controllers. We will focus on linear controllers for keeping complexity of control signals to be computed as low as possible. Related to the plant, first the fully-observable case is analyzed, that is the design of the controller assuming that the state is available and there are no missing elements. Then, the partially-observable case is introduced, along with state estimation based on Kalman Filter.
- Chapter 4: This chapter makes use of the fully-observable state-feedback control shown in Chapter 3, formulating this control problem as an MDP and then deriving a new approach for adaptive control using reinforcement learning, which is often interesting due to its practical applications, since it does not require complete knowledge of system dynamics.
- Chapter 5: This chapter shows the implementation results on the Furuta pendulum, under a simulated environment and also on the physical prototype, including an appropiate discussion.
- Chapter 6: This chapter wraps up and summarizes the thesis. Directions for future research are also discussed.
- Appendix A: This appendix includes a review of some useful matrix properties to be used throughout this thesis.

This thesis has made a direct and indirect contribution to the reinforcement learning area, in terms of tackling problems from this thesis or contributing to RL applications knowledge respectively. For instance, [36] propose the formulation and implementation of a state-feedback control architecture which is not frequently used, mainly due to the additional parameters introduced that make the hand-tuning task of obtaining an stabilizable controller more complex, but this limitation can be lifted if parameters are initially chosen and updated such that a performance index is optimized in terms of a reinforcement learning problem.

Another work directly related to this thesis, currently under review, corresponds to **A switched control strategy for swing-up and state regulation for the rotary**

[1] Chapter 2 and Appendix present known results and, accordingly, no contributions.

inverted pendulum, which presents the experimental results obtained on the stabilization problem of the physical rotary inverted pendulum, a subactuated system with acquired signals limited to position (angle) measurements from the two degrees of freedom, estimating velocities from a soft-sensor and using this information for generating an appropiate control signal in a state-feedback loop.

Concerning contributions to the reinforcement learning area of applications, the work entitled [1] refers to one of the initial experiments conducted by the author in this area, extending the application's spectrum of reinforcement learning algorithms to the learning task of a goalkeeper in a certain league (small-size league) of the Robot Soccer World Cup, since the student had participated in the development and organization of the corresponding team at this institution, and other leagues had already proposed this approach, noting of course, there are several applications of this learning approach to the wider area of robotics itself.

Then, the last application's work corresponds to a book chapter [33], presenting a robotic hexapod plattform development, and presents early results about this robot walking pattern learning. The curse of dimensionality problem that faces this particular reinforcement learning problem, given the high number of degrees of freedom of this robotic crab, is tackled with an artificial neural network as function approximator, whose evolution is not limited to update the network weights but it is also concerned with evolving topology of this neural network.

Chapter 2

NOTATION AND PRELIMINARIES

2.1 Introduction

The purpose of this chapter is to present definitions and basic results that are used repeatedly throughout this thesis. Minors definitions and known results whose use is restricted to isolated sections, will be presented in the Appendix at the end of this thesis, and referred to when appropiate.

2.2 State space models

When designing a control system, that is, deciding the best structure and/or tuning parameters for making the system to be controlled (from now on referred to as *plant*) to behave as desired, it is often required to have a model. This model describes a phenomenological interaction between all the relevant variables that affects the signals of interest.

When the model corresponds to a continuous-time approximation for these physical interactions, it describes a set of differential equations, while in the case of discrete-time models we have difference equations. A linear version of a high order difference equation model would be of the form

$$y[k+n] + a_{n-1}y[k+n-1] + ... + a_0 y[k] = b_m u[k+m] + ... + b_0 u[k], \tag{2.2.1}$$

where a_i and b_j correspond to real-valued coefficients for relating the measured output y excited by the input u, or real-valued coefficient vectors in the multi-input multi-output case, with $i \in [0, n-1]$, $j \in [0, m]$ and $i, j, k \in \mathbb{N}_0^+$.

This linear difference equation can also be described by introducing a number of internal variables $x_1, x_2, ..., x_{n_x}$ that can be written as vector notation given by the internal state vector $x[k]$ at time k:

$$x[k] = \left[\; x_1[k], ..., x_{n_x}[k] \; \right]^\intercal, \tag{2.2.2}$$

so difference equation can be written in terms of this internal state vector that evolves as

$$x[k+1] = f\left(x[k], u[k]\right), \tag{2.2.3}$$

where $f(x,u)$, $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$, is a state transition function such that:

$$f(x,u) = \left[\; f_1(x,u), ..., f_{n_x}(x,u) \; \right]^\intercal . \tag{2.2.4}$$

The state is exactly the information that has to be stored and updated in order to be able to calculate the output, by using a given function $h(x,u)$, $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$, so the following model is obtained:

$$x[k+1] = f\left(x[k], u[k]\right), \tag{2.2.5a}$$
$$y[k] = h\left(x[k], u[k]\right). \tag{2.2.5b}$$

Noting that as n_x is used for the number of internal states of the system, n_u and n_y denote the number of inputs and outputs respectively. Then, a state-space model is said to be linear if $f(x,u)$ and $h(x,u)$ are linear functions of x and u:

$$f(x,u) = Ax + Bu, \tag{2.2.6a}$$
$$h(x,u) = Cx + Du, \tag{2.2.6b}$$

which directly leads to

$$x[k+1] = Ax[k] + Bu[k], \tag{2.2.7a}$$
$$y[k] = Cx[k] + Du[k], \tag{2.2.7b}$$

with $x[k]$,$y[k]$ and $u[k]$ corresponding to the internal state, measured output and control signal respectively at time k, and A,B,C and D are (usually known and constants for the time invariant case) matrices of dimensions $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $C \in \mathbb{R}^{n_y \times n_x}$ and $D \in \mathbb{R}^{n_y \times n_u}$, where n_x, n_y, n_u correspond to the number of internal states, number of outputs and inputs respectively.

In addition, it is often interesting to analyze the stability of the system to be controlled, that is analyze boundness of the signal to be controlled under different circumstances, since depending on the criteria used for controlling a given process it could be possible that the measured output follows the desired behavior in the long-term, but causing the output of the system to diverge at several times before achieving this, which in practical applications could even mean that the physical plant crashes before achieving the desired behavior.

Although in the case of linear models, eigenvalues of matrix A or eigenvalues of an expression involving this matrix A and parameters of the controller are often studied, the next chapter will introduce some suitable definitions of stability for the problem of interest, introducing stochasticity in the model (2.2.7) by adding noise in the process.

Discrete-time systems under study in this thesis, assume a phenomenological relation between their variables described by a model given by difference equations. A linear version of a high order difference equation model would be of the form

$$y[k+n] + a_{n-1}y[k+n-1] + ... + a_0 y[k] = b_m u[k+m] + ... + b_0 u[k], \tag{2.2.8}$$

where a_i and b_j correspond to real-valued coefficients for relating the measured output y excited by the input u, or real-valued coefficients vectors in the multi-input multi-output case, with $i \in [0, n-1]$, $j \in [0, m]$ and $i, j, k \in \mathbb{N}_0^+$.

In order to obtain a transfer function, that is a a relation between the input and output of a linear time-invariant system with zero initial conditions at a given fixed point equilibrium, it is convenient to define the forward shift operator, q, defined by

$$\begin{aligned} q(\cdot): \quad & \mathbb{R}^{\mathbb{Z}} \to \mathbb{R}^{\mathbb{Z}}, \\ & q\left(f[k]\right) \triangleq f[k+1], \end{aligned} \tag{2.2.9a}$$

for a given differentiable and bounded function f. Then, the model (2.2.8) in terms of this operator would be given by

$$q^n y[k] + a_{n-1} q^{n-1} y[k] + ... + a_0 y[k] = b_m q^m u[k] + ... + b_0 u[k]. \tag{2.2.10}$$

Then, it is appropiate to introduce the Zeta-transform for obtaining the frequency response of the system with a Kronecker impulse as input at time k, with zero initial conditions (null derivatives of $y[k]$ and $u[k]$ for all k), i.e.,

$$H(z) = \mathcal{Z}\{y[k]\}|_{u[k]=\delta[k]}, \tag{2.2.11}$$

which is useful for studying features of the control system such as stability, and designing a controller also described by a transfer function. For Multi-Input Multi-Output (MIMO) systems, the same feedback control structure remains, but instead transfer matrices are obtained.

Given a discrete-time function $f[k]$, there is a bilateral and unilateral $\mathcal{Z}$-transform respectively given by $\mathcal{Z}_b$ and $\mathcal{Z}_u$:

$$\mathcal{Z}_b\{f[k]\} = F[z] = \sum_{k=-\infty}^{\infty} f[k] z^{-k}, \tag{2.2.12a}$$

$$\mathcal{Z}_u\{f[k]\} = F[z] = \sum_{k=0}^{\infty} f[k] z^{-k}. \tag{2.2.12b}$$

Note that throughout this thesis, $\mathcal{Z}$-transform will be used as short-hand for $\mathcal{Z}_u$, that is the unilateral $\mathcal{Z}$-transform, and it is also possible to get back to time domain by using the inverse transform, given by:

$$\mathcal{Z}^{-1}\{F[z]\} = f[k] = \frac{1}{2\pi j} \oint_{\Gamma} F[z] z^{k-1} dz, \tag{2.2.13}$$

where the contour limit Γ of the closed path integral is chosen inside the region of convergence $|z| > \rho$, so the $\mathcal{Z}$-transform and its inverse are well-defined (they exist) if there is some $\rho > 0$ and a positive constant $k < \infty$ such that

$$|f[k]| \leq k\rho^k; \qquad k \geq 0, \tag{2.2.14}$$

i.e., it suffices that $f[k]$ belongs to an exponential order in discrete-time.

Note that z in (2.2.12) is, in general, a complex number:

$$z = Ae^{j\phi},$$

$$= A\left(cos\phi + jsin\phi\right), \tag{2.2.15}$$

where A is the magnitude of z, j is the imaginary unit, and ϕ is the complex argument in radians.

Then, making an algebraic arrangement and applying $\mathcal{Z}$-transform to (2.2.10):

$$Y(z)\left(z^n + a_{n-1}z^{n-1} + ... + a_0\right) = U(z)\left(b_m z^m + ... + b_0\right), \tag{2.2.16}$$

so defining polynomials:

$$A(z) = z^n + a_{n-1}z^{n-1} + ... + a_1 z + a_0, \tag{2.2.17a}$$
$$B(z) = b_m z^m + ... + b_1 z + b_0, \tag{2.2.17b}$$
$$\tag{2.2.17c}$$

which replacing into (2.2.16) and solving for $Y(z)/U(z)$ lead to the transfer function $H(z)$:

$$\begin{aligned} H(z) &= \frac{Y(z)}{U(z)}, \\ &= \frac{b_m z^m + ... + b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + ... + a_1 z + a_0}. \end{aligned}$$

If a dummy variable, $E(z)$, is defined in order to split (2.2.16) into two parts:

$$E(z) = \frac{1}{z^n + a_{n-1}z^{n-1} + ... + a_1 z + a_0}U(z), \tag{2.2.18a}$$
$$Y(z) = \frac{b_m z^m + ... + b_1 z + b_0}{E(z)}, \tag{2.2.18b}$$

expression in (2.2.18a) may be solved for $U(z)$ in terms of

$$U(z) = \left(z^n + a_{n-1}z^{n-1} + ... + a_1 z + a_0\right)X(z), \tag{2.2.19}$$

and rearranged to generate a feedback structure that can be used as the basis for a block composition as:

$$E(z) = \frac{1}{a_0}U(z) - \left(\frac{a_1}{a_0}\frac{1}{z} + ... + \frac{a_{n-1}}{a_0}\frac{1}{z^{n-1}} + \frac{a_n}{a_0}\frac{1}{z^n}\right)E(z). \tag{2.2.20}$$

Then it is clear to observe that the dummy variable $E(z)$ is specified in terms of the system input $u[k]$ and a weighted sum of successive terms of itself. A set of state equations may be found from these expressions, assigning the state variables $x_i[k]$ (for $i = 1, 2, ..., n$) to the outputs of the n previous equations, which by inspection takes the form:

$$x_1[k+1] = x_2[k], \tag{2.2.21a}$$
$$x_2[k+1] = x_3[k], \tag{2.2.21b}$$
$$\vdots = \vdots$$
$$x_{n-1}[k+1] = x_n[k], \tag{2.2.21c}$$

$$x_n[k] = -\frac{a_n}{a_0}x_1[k] - \frac{a_{n-1}}{a_0}x_2[k] - ... - \frac{a_1}{a_0}x_n[k] + \frac{1}{a_0}u[k]. \tag{2.2.21d}$$

Expressions in (2.2.21) written in matricial notation, lead to an alternative derivation of the state space model described in Section 1.1, which allows a deeper analysis of the system features, given that transfer functions are often limited by their controlable and observable components. Also, given the initial value of the state of the plant and the value of signals that act as inputs, it is possible to predict the output values as a combination of the previously mentioned signals.

For discrete-time models, recall (2.2.7) and its matrices of dimensions $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $C \in \mathbb{R}^{n_y \times n_x}$, where n_x, n_y, n_u correspond to the number of internal states, number of outputs and inputs respectively. Then, if we incorporate noisy measurement and process disturbances, the state space representation is given by

$$x[k+1] = Ax[k] + Bu[k] + v[k], \tag{2.2.22a}$$
$$y[k] = Cx[k] + w[k], \tag{2.2.22b}$$

where $x[k]$, $u[k]$ and $y[k]$ correspond to the internal state of the plant, control signal and measured output of the plant respectively at time k, just like in (2.2.22), but on the other hand, now we introduced uncorrelated (gaussian, and white, concept developed in the subsequent paragraphs) noises $v[k]$ and $w[k]$, namely process and measurement noise with zero-mean and constant variance P_v and P_w respectively. Introducing these kinds of noise is necessary for dealing with uncertainty and disturbances unlike the model introduced in (2.2.7).

Remark 2.1. *In practice, control signals on physical systems often do not cause an instantaneous effect on their outputs, which means that the numerator of the transfer function will have higher order than the denominator, and in the state space representation this implies* $D = 0$ *in* (2.2.22)*, which is the case for the model to consider in the remainder of this thesis, as shown in* (2.2.22). □

In this thesis, we use the simplest notion of random process, that is, we understand the random process $x[k]$ with $k \in \mathbb{N}_0^+$ as a sequence of (possibly vector valued) real random variables that have well defined (joint) PDFs (see [5]).

The following definitions are standard, and can be found in the previously referenced work.

Definition 2.1 (**Notation on stochastic processes**)**.**
Consider two discrete and complex-valued random processes $x[k]$ *and* $y[k]$*, and let* h *be a function defined in the real numbers. Then,*

1. *The mean of* $h(x)$*, for* x *discrete, denoted by* $\mu_x[k]$*, is defined by*

$$\begin{aligned} \mu_x[k] &\triangleq E_F\{x[k]\}, \\ &= \sum_k h(k) \cdot F(k). \end{aligned} \tag{2.2.23}$$

where $E_F\{\cdot\}$ *denotes the expectation operator over the probability mass function* F*.*

2. *The cross-variance function between x and y, denoted by $R_{xy}[k+\tau, k]$, is defined by*

$$R_{xy}[k+\tau, k] \triangleq E_F\{(x[k+\tau] - \mu_x[k+\tau])\,(y[k] - \mu_y[k])^\intercal\}, \tag{2.2.24}$$

for all k and τ such that $x[k+\tau]$ and $y[k]$ are defined, that is $k+\tau \geq 0$.
Moreover, if $x[k] = y[k]$, for all k, then the covariance function of x is given by

$$R_x[k+\tau, k] \triangleq R_{xx}[k+\tau, k]. \tag{2.2.25}$$

3. *The variance matrix of $x[k]$, denoted by $P_x[k]$, is defined by*

$$P_x[k] \triangleq R_x[k, k], \tag{2.2.26}$$

while the variance $\sigma_x^2[k]$ of x is defined by

$$\sigma_x^2[k] \triangleq tr\{P_x[k]\}, \tag{2.2.27}$$

with $tr\{\cdot\}$ being the trace of $\cdot$.

4. *The second order moments matrix of $x[k]$, denoted by $M_x[k]$ is defined by*

$$M_x[k] \triangleq E_F\{x[k]x[k]^\intercal\} \tag{2.2.28}$$

■

Also, in this thesis we will encounter the following special random processes.

Definition 2.2 (**Some special random processes**)**.**

1. *x is said to be white, if and only if it is a sequence of uncorrelated random variables with the same mean and variance matrix.*
2. *x is an i.i.d. sequence if and only if it is a sequence of identically distributed and independent random variables.*
3. *A stochastic process is said to be ergodic if its statistical properties can be deduced from a single, sufficiently long, random sample of the process, representing the average statistical characteristics of the entire process.*
4. *The gaussian white noise is a stationary and ergodic random process with zero mean from an i.i.d. sequence, that is defined by its fundamental property: any two values from the sequence are statistically independent without considering how close they are in time.*
5. *x is an uncorrelated process, without specifying with respect to what, if and only if x is uncorrelated with any other random variable or process.*
6. *x is a second order process if and only if its mean and second order moments matrix exist and are finite for every $k \in \mathbb{N}_0^+$ and, moreover, remain finite when $k \to \infty$.*

7. *x is an asymptotically wss (wide-sense stationary) process if and only if its stationary mean and stationary covariance function, denoted by μ_x and $R_x[\tau]$ respectively, are finite, exist and are defined by*

$$\mu_x \triangleq \lim_{k\to\infty} \mu_x[k], \tag{2.2.29a}$$

$$R_x[\tau] \triangleq \lim_{k\to\infty} R_x[k+\tau, k]. \tag{2.2.29b}$$

In these cases, we also define the stationary variance of x by

$$\sigma_x^2 \triangleq \lim_{k\to\infty} \sigma_x^2[k]. \tag{2.2.30}$$

■

An important concept in linear systems is stability. In the typical transfer function description of linear systems, stability is usually understood as bounded-input bounded-output, or BIBO stability. That is, a linear system mapping $u[k]$ to $y[k]$ is considered stable if any bounded input $u[k]$ results in a bounded output $y[k]$. However, for state space models a slightly stronger definition is needed for the internal state of the system, $x[k]$.

Internal stability study boundedness and asymptotic behavior of solutions of the following expression, when no input signal is supplied:

$$x[k+1] = Ax[k], \qquad x[k_0] = x_0 \tag{2.2.31}$$

so bounds of interest are independent of the choice of initial state x_0.

Definition 2.3 (**Exponential stability**, see [32].)**.**
A system represented through a linear state equation given in the form of (2.2.31) *is said to be exponentially stable on a given equilibrium point (point in the state space where dynamics of the system is zero), i.e., for the equilibrium point $x[k] = 0$ since* (2.2.31) *does not consider an external input, if there exists* ***positive*** *constants c, γ and λ such that*

$$||x[k]|| \leq \gamma ||x[k_0]|| e^{-\lambda(k-k_0)}, \quad \forall ||x[k_0]|| < c, \tag{2.2.32}$$

with $||x[k]||$ representing the norm of vector $x[k]$ at time k, and k_0 denotes the initial time step. Moreover, the system (2.2.31) *is said to be globally exponentially stable if it is exponentially stable for any initial state $x[k_0]$.*

As a consequence, (2.2.31) *is exponentially stable if and only if all eigenvalues of matrix A (as defined on* (2.2.22)*) lie strictly inside the unit circle.* ■

For the sake of clarity of Definition 2.3, suppose that λ is a real eigenvalue of A in (2.2.31) with magnitude $|\lambda| \geq 1$. Then, let p be an eigenvector associated with λ, so the following expressions allow us to observe that the corresponding solution of (2.2.31) does not go to zero as $k \to \infty$ when $x_0 = p$, for $p \in \mathbb{R}^{n_x}$.

$$A^k p = \lambda^k p; \qquad k \geq 0, \tag{2.2.33}$$

Remark 2.2. *The same previous analysis can be done in the general case when λ is a complex eigenvalue of matrix A with $|\lambda| \geq 1$, such as $x_0 = Re\{p\}$ or $x_0 = Im\{p\}$, that is, the real or complex component of the corresponding eigenvector respectively.* □

If the system in (2.2.22) is unstable, then a controller could stabilize the plant if there exists a matrix L such that $(A - BL)$ is stable. To illustrate this concept, consider a simple state-feedback architecture shown in Figure 2.1, where $L \in \mathbb{R}^{n_u \times n_x}$ corresponds to a (matrix) gain that weights the plant state $x[k]$ in order to generate a control signal $u[k]$ for tracking reference $\bar{r}[k]$ at time k. Usually, in a state-feedback control scheme as shown in Figure 2.1, $r_L[k]$ is a pre-filtered reference translated from the external reference $\bar{r}[k] \in \mathbb{R}^{ny}$ which is the desired value that should take $y[k]$ at time k, into a suitable vector with appropiate dimensions.

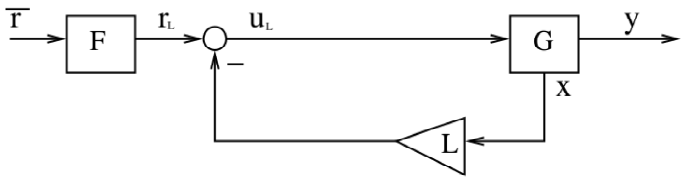


Figure 2.1: Classical state-feedback control scheme

It is well known from the literature (e.g., [15, 34]), that a stabilizing controller (matrix $L \in \mathbb{R}^{nu \times nx}$) for the plant G, has to be designed such that the eigenvalues of $eig(A - BL)$ lie inside the unit circle. Moreover, if we want to focus also on tracking, that is getting $y[k]$ as close as possible to $\bar{r}[k]$, the prefilter F would be given by

$$F = -\left(C\left(A - BL - I\right)^{-1} B\right)^{-1}. \tag{2.2.34}$$

Then, the state update in (2.2.22) would reduce to

$$x[k+1] = (A - BL)x[k]. \tag{2.2.35}$$

Therefore, if $(A - BL)$ is stable, the state $x[k] \to 0$. In this sense, the $u[k] = -Lx[k]$ can be understood as a stabilizing feedback control law, since it drives the system state to zero from any initial condition.

Since in this thesis we are studying dynamic systems subject to random inputs, deterministic notions of stability need to be extended accordingly. In particular, we will focus on the concept of MSS (mean square stability).

Assumption 2.1. *Consider the system in* (2.2.22)*. Then, we will assume that the following holds:*

- *The initial state $x[0] = x_0$ is a second order random variable having mean μ_{x_0} and variance matrix $P_{x_0} \geq 0$.*
- *The input u is characterized in terms of the state x.*

- *The noise v is a second order zero-mean white sequence uncorrelated with x_0, and variance matrix $P_v \geq 0$.*

□

Then, MSS is defined next.

Definition 2.4 (**Mean square stability**).
The system in (2.2.22) *is mean square stable if and only if, for every x_0 and every u and v that satisfy Assumption 2.1, there exist $\mu_x \in \mathbb{R}^{n_x}$ and $M_x \in \mathbb{R}^{n_x \times n_x}$, $M_x \geq 0$, such that*

$$\lim_{k\to\infty} E\{x[k]\} = \mu_x, \tag{2.2.36a}$$

$$\lim_{k\to\infty} E\{x[k]x[k]^\intercal\} = M_x. \tag{2.2.36b}$$

■

Definition 2.4 states that the system (2.2.22) is mean square stable if and only if its state x has well defined and finite stationary mean and stationary second order moments matrix.

It is often the case, as one of the particular interest in this thesis, that the internal system x is not directly available and has to be deduced from samples of the (usually noisy) measured output y.

As intuition may suggest, state-estimators only make sense when it is possible to do so. Thus, observability is next defined.

Definition 2.5 (**Observability**).
Consider the system in (2.2.22). *Then the state x is said to be completely observable if the following matrix is full-rank:*

$$rank\left(\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n_x-1} \end{bmatrix}\right) = n_x, \tag{2.2.37}$$

where n_x is the dimension of the state vector x. ■

Although the notion of observability has been extensively treated in the literature, for the sake of clarity, recall the model in (2.2.22):

$$x[k+1] = Ax[k] + Bu[k] + v[k], \tag{2.2.38a}$$

$$y[k] = Cx[k] + w[k]. \tag{2.2.38b}$$

Then, it is straightforward that

$$\begin{aligned} y[0] &= Cx[0] + w[0] \\ y[1] &= Cx[1] + w[1] \\ &= CAx[0] + CBu[0] + Cv[0] + w[1] \\ y[2] &= CA^2x[0] + CABu[0] + CBu[1] + CAv[0] + Cv[1] + w[2] \end{aligned}$$

$$\vdots$$

$$y[n-1] = CA^{n-1}x[0] + CBu[n-2] + ... + CA^{n-2}Bu[0] + Cv[n-2] + ... + CA^{n-2}v[0] + ... + w[n-1]$$

which expressed on a matricial form, gives

$$\begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ y[n-1] \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} x[0] + \begin{bmatrix} 0 & 0 & ... & 0 & 0 \\ CB & 0 & ... & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ CA^{n-2} & CA^{n-3} & ... & CB & 0 \end{bmatrix} \begin{bmatrix} u[0] \\ u[1] \\ \vdots \\ u[n-2] \\ u[n-1] \end{bmatrix}$$
$$+ \begin{bmatrix} 0 & 0 & ... & 0 & 0 \\ C & 0 & ... & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ CA^{n-2} & CA^{n-3} & ... & C & 0 \end{bmatrix} \begin{bmatrix} v[0] \\ \vdots \\ v[n-2] \\ v[n-1] \end{bmatrix} + \begin{bmatrix} w[0] \\ w[1] \\ \vdots \\ w[n-1] \end{bmatrix} \quad (2.2.39)$$

Then, from (2.2.39) it can be noted that for reconstructing the initial state at time $n-1$ from samples of y up to time $n-1$, the referred matrix has to be full-rank. Moreover, for the system to be completely observable at any time n, it would suffice to choose an arbitrary n which in the literature it is often chosen as $n = n_x$.

Remark 2.3. *This notion of observability on Definition 2.5 remains for the case of non-strictly causal systems, i.e., $D \neq 0$ in* (2.2.7) *(equivalently adding a known input $u[k]$ for direct influence of $y[k]$) on* (2.2.22)*, and its proof is straightforward.* □

Assuming the system in (2.2.22) is completely observable, the best linear estimator (in a minimum mean square sense) is given by the Kalman Filter, originally shown in [20, 21], and covered in any standard text on linear systems such as [35].

Definition 2.6 (**Best Linear and Unbiased Estimator**)**.**
Assume a linear state space model given by (2.2.22) *with disturbances $v[k]$ and $w[k]$ as uncorrelated (gaussian) white noises with zero-mean and variances P_v and P_w respectively. Then, the state estimates $\hat{x}$, constructed from observations of the measured output $\{y[k], y[k-1], ..., y[0]\}$ will be given by*

$$\hat{x}[k|j] \triangleq E\{x[k]|y[0:j], u[0:j]\}, \quad (2.2.40)$$

where $y[0:j]$ represents the subsequence $(y[0], y[1], ..., y[j])$. Thus, $\hat{x}[k|j]$ is the MMSE (minimum mean square error) estimate of the state $x[k]$ given the observed output up to sample j. Corresponding to the state estimate $\hat{x}[k|j]$, let $P[k|j]$ denote the estimation error variance,

$$P[k|j] \triangleq E\{(x[k] - \hat{x}[k|j])\,(x[k] - \hat{x}[k|j])^{\intercal}\,|y[0:j], u[0:j]\}. \quad (2.2.41)$$

Note that the above estimates are difficult to compute for an arbitrary distribution in $v[k]$ and $w[k]$, so Gaussianity assumption should still hold. If this is the case, estimates can be described by a simple set of recursive equations, whose estimator is the so-called Kalman Filter.

$$\hat{x}[k|k] = \hat{x}[k|k-1] + P[k|k-1]C^{\intercal}K[k]e[k] + Bu[k] \tag{2.2.42a}$$
$$K[k] = \left(CP[k|k-1]C^{\intercal} + P_w\right)^{-1} \tag{2.2.42b}$$
$$e[k] = y[k] - C\hat{x}[k|k-1] \tag{2.2.42c}$$
$$\hat{x}[k|k-1] = A\hat{x}[k-1|k-1] \tag{2.2.42d}$$
$$P[k|k-1] = AP[k-1|k-1]A^{\intercal} + P_v \tag{2.2.42e}$$
$$P[k|k] = P[k|k-1] - P[k|k-1]C^{\intercal}K[k]CP[k|k-1] \tag{2.2.42f}$$

where as could be deduced, $P_u[k]$ refers to variance of the known input $u[k]$ at time k. ■

2.3 Reinforcement Learning with Gaussian Processes

Since we will be dealing with gaussian processes, meaning that this process is completely determined by its second order moments, i.e. by its mean and covariance functions, an appropiate derivation of algorithms is required, so subsequent sections in this chapter will review reinforcement learning algorithms on systems with uncertainty modeled by a gaussian process.

2.3.1 Uncertainty and Learning

It is broadly accepted that probability theory is an appropiate manner of tackling the uncertainty representation problem, and accordingly, probabilistic models are within the most used framework for systems facing uncertainty and disturbances. In a more particular setup, parametric distributions enable representation and generalization of complex data within a bounded quantity of parameters, which correspond to statistics that summarize the knowledge of the initially unknown and empirically constructed distribution.

If a family of distributions is assumed to take place on a random variable over the analyzed system, parameters should be fully informative in order to be useful. For example, mean and variance are sufficient statistics, or parameters for the family of Gaussian distributions, because its complete description can be specified through these parameters.

Stochastic models can be used in several ways, such as computing statistics like mean and variance using expectations, or predicting probability of correlated quantities. Consider two correlated random variables (according to Definition 2.1) with known probability distributions $Pr(X|Y = y)$ and $Pr(Y)$. Then, the probability of $Y = y$ given that $X = x$ has been measured, can be computed by using the Bayes rule:

$$Pr(Y = y|X = x) = \frac{Pr(X = x|Y = y) \cdot Pr(Y = y)}{Pr(X = x)}, \tag{2.3.1}$$

where the probability of $X = x$ can be computed using the law of total probability:

$$Pr(X = x) = \int_{y \in Y} Pr(X = x|Y = y) \cdot Pr(Y = y)dy. \tag{2.3.2}$$

In these models, it is assumed that the parameters of distributions are given. On the other hand, as usually found in real-world applications, parameters have to be inferred

with a finite set of samples or observations from the phenomenon, if available. A classical statistical learning procedure refers to the maximum-likelihood estimation.

Let X be a random variable, and x the vector of available samples of X. Assuming samples of this random variable are independent and identically distributed, a maximum-likelihood estimation of parameters of X is:

$$\theta^* = argmax_\theta \mathcal{L}(x|\theta), \tag{2.3.3a}$$

$$\mathcal{L}(x|\theta) \propto \Pi_i Pr(X = x_i|\theta), \tag{2.3.3b}$$

with $\mathcal{L}$ being the likelihood function. This approach is also often known as frequentist, because it relies on how frequent are values from observations to infer the parameters of the model that describes the system.

2.3.2 Reinforcement Learning

When we talk about Reinforcement learning, we refer to the area inside Machine learning, concerned with solving sequential decision problems modelled by Markovian Decision Processes (MDPs). Nevertheless, the applications spectrum has been extended to areas such as Robotics [23] or Control Theory [17, 25, 40].

In simple terms, the main objective of this type of learning is to maximize the expected long-term reward, in an (initially) unknown environment through finding an optimal sequence of actions to take for a given problem. For instance, a finite-horizon problem implying the agent (learner) has to choose actions such that the expected return J is maximized over the next H time steps:

$$J = E_F[\sum_{i=0}^{H} r_i], \tag{2.3.4}$$

where r_i corresponds to the instantaneous reward obtained at time step i, over an unknown probability distribution F for the state transition function, which will be empirically obtained through interactions between the learning agent and its environment. Figure 1.2 illustrates the interaction process where the agent perceives both its external and internal features that allow it to infer its state, and then executes actions through the learned decision-maker which lead the agent to a certain reinforcement signal (reward or punishment).

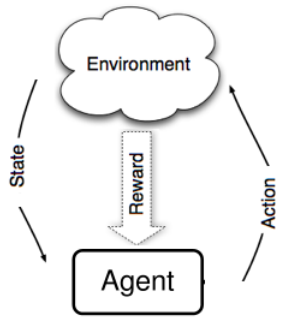


Figure 2.2: Reinforcement Learning scheme

While the agent (the learner) decides the actions to execute, it has to make a trade-off between exploiting what is already known to avoid executing actions that could lead to lower utilities, and exploring new solutions that eventually could allow the obtention of bigger future rewards. This trade-off is usually known as exploration-exploitation dilemma [19], which has been extensively treated in literature, developing undirected exploration methods (such as Boltzmann exploration [39]) which explores the whole state-action space by making a probabilistic choice of possible actions to take, and directed exploration methods, which use statistical information obtained through past experiences [42].

Then, reinforcement learning can be broadly summarized as a computational approach to the concept of learning from interactions with the environment, where the agent must sequentially interact with a (partially) unknown scenario in order to maximize the sum of (potentially unknown) scalar rewards. This scenario is not limited to the physical location and features of the real-world environment, but it also extends to internal stimuli that could affect the agent behavior.

In formal terms, a reinforcement learning problem, formulated as an appropiate MDP is composed by the tuple $< \mathcal{X}, \mathcal{U}, \mathrm{T}, \mathrm{R} >$ where

- $\mathcal{X}$: corresponds to the set of all possible states.
- $\mathcal{U}$: denotes the set of actions the agent could take.
- T: $\mathcal{X} \times \mathcal{U} \times \mathcal{X} \rightarrow [0, 1]$ is a state transition function, which assigns the probability of the agent being transferred from state x to state x' by executing action u.
- R: $\mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}$ corresponds to a (real-valued) reward scalar function.
- π: $\mathcal{X} \rightarrow \mathcal{U}$ is a mapping from states to actions, describing the policy (actions to take on given states).

When the decision problem involves states that are not being observed, it is necessary to introduce two new elements to the previous tuple, translating the problem to a POMDP (Partially Observable MDP)

- Z $= \{z_1, z_2, ..., \}$: corresponds to the set of observations
- O: $2^{\Omega} \rightarrow [0, 1]$ denotes the observation function, where $\Omega : \mathrm{Z} \times \mathcal{U} \times \mathcal{X}$, and assigns the probability of making an observation o when executing action u in state x.

The quality of a given policy π is quantified by the value function $V^{\pi}(x_t)$, defined as the expected (discounted) accumulated reward for starting in state x at time t:

$$V^{\pi}(x_t) = E_F[\sum_{i=0}^{\infty} \gamma^{i} r[t+i] | \pi], \tag{2.3.5}$$

where $\gamma \in [0, 1]$ corresponds to the discount factor, $r[t+i]$ stand for the immediate reward obtained at time $t+i$, and expectation is taken over an initially unknown distribution F for the state transition function, being built experimentally over the interactions with the environment.

Then, a policy π^* is said to be optimal, if the following expression holds,

$$V^{\pi^*}(x) \geq V^{\pi}(x), \quad \forall x \in \mathcal{X}, \pi \tag{2.3.6}$$

i.e., the optimal policy π^* will yield the biggest value function with respect to every possible policy π, independently of initial state x.

The optimal policy π^* will satisfy the Bellman optimality expression

$$V^*(x) = \max_{u \in \mathcal{U}} \sum_{x' \in \mathcal{X}} p(x'|x, u) \left(R(x, u) + \gamma V^*(x')\right), \tag{2.3.7}$$

where $V^*(x)$ is the short-hand for $V^{\pi^*}(x)$, $p(x'|x, u)$ denotes the probability of reaching state x' when u is executed on x and $R(x, u)$ is the obtained reward for executing action u on state x.

2.3.3 Algorithms

Recalling the basic definitions of MDPs presented in Section 2.3.2, whose main elements that will be used throughout this thesis, and introducing a slightly more familiar notation with the one existing in control theory, are

- $\mathcal{X}$: denotes the state space
- $\mathcal{U}$: denotes the action space
- R: $\mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}$ is the immediate reward
- $p : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \rightarrow [0, 1]$ is the transition probability distribution, which will be assumed to be stationary

A useful quantity is the discounted return

$$D(x[k]) = \sum_{i=k}^{\infty} \gamma^{i-k} R(x[i], u[i]), \tag{2.3.8}$$

which along with the stationarity of the MDP yield

$$D(x[k]) = R(x[k], u[k]) + \gamma D(x[k+1]). \tag{2.3.9}$$

Let us define the expectation operator E_π as the expectation over all possible trajectories and all possible rewards collected therein. This allows us to define the value function $V(x[k])$ as the result of applying this expectation operator to the discounted return $D(x[k])$, i.e.,

$$V(x[k]) = E_\pi\{D(x[k])\}. \tag{2.3.10}$$

Thus,

$$\begin{aligned} V(x[k]) &= E_\pi\{D(x[k])\} \\ &= E_\pi\{R(x[k], u[k]) + \gamma D(x[k+1])\} \\ &= E_\pi\{R(x[k], u[k])\} + \gamma E_{x[k+1]} E_\pi\{D(x[k+1])|x[k+1]\} \\ &= E_\pi\{R(x[k], u[k])\} + \gamma E_{x[k+1]}\{V(x[k+1])\} \end{aligned} \tag{2.3.11}$$

This last equality in (2.3.11) is known as the fixed-policy version of the Bellman equation [8].

The policy π that maximizes the expected discounted return from each state is called an optimal policy, and is denoted by π^*. Then, the value function corresponding to an optimal policy is called the optimal value, and is denoted by V^*. While there may exist multiple optimal policies, the optimal value is unique [9] and may be computed by solving the Bellman optimality equation

$$V^*(x[k]) = \max_{u \in \mathcal{U}} \sum_{x[k+1]} Pr\{(x[k], u[k], x[k+1])\} \left(R(x[k], u[k]) + \gamma V^*(x[k+1])\right), \tag{2.3.12}$$

where $Pr\{x[k], u[k], x[k+1]\}$ stands for the probability of being transferred to some state $x[k+1]$ at time $k+1$ when executing action $u[k]$ on state $x[k]$ at time k.

For the remainder of this section, omit the time dependence of the state, i.e. $x[k]$ will be referred to just as x, and the next state $x[k+1]$ will be x'. Then, one of the two most popular and convergent in probability dynamic programming algorithms, is the Value Iteration (VI) algorithm, whose pseudocode is shown Algorithm 1, and works by arbitrarily initializing a value estimate, for all $x \in \mathcal{X}$.

Algorithm 1 Value Iteration algorithm

$i = 0$
$\hat{V}_0(x) = 0 \qquad \forall x \in \mathcal{X}$
while $\Delta > \epsilon$ **do** ▷ (for a small positive number ϵ)
  for each $x \in \mathcal{X}$ **do**
    $\hat{V}_{i+1}(x) = \max_{u \in \mathcal{U}} \sum_{x' \in \mathcal{X}} Pr\{(x, u, x')\} \left(R(x, u) + \gamma V(x')\right)$
  end for
  $\Delta = \|\hat{V}_{i+1} - \hat{V}_i\|$
  $i = i + 1$
end while
return policy π ▷ such that maximizes (2.3.12)

This algorithm involves the computation of some norm of the difference between two subsequent value estimates. Several theoretical results exist, which provide bounds for the performance of π^* as a function of ϵ for different norms, including the maximum norm $\|\cdot\|_\infty$ [46] and weighted l_2 norms [27].

Algorithm 2 Policy Iteration algorithm

1: $\hat{\pi}_0(x) =$ some random action $\quad \forall x \in \mathcal{X}$
2: $\hat{V}_0(x) = 0 \quad \forall x \in \mathcal{X}$ ▷ arbitrarily
3: $i = 0$
4: **while** $\Delta > \epsilon$ **do** ▷ (for a small positive number ϵ)
5: **for** each $x \in \mathcal{X}$ **do**
6: $\hat{V}_{i+1}(x) = \max_{u \in \mathcal{U}} \sum_{x' \in \mathcal{X}} Pr\{(x, u, x')\} \left(R(x, u) + \gamma V(x')\right)$
7: **end for**
8: $\Delta = \|\hat{V}_{i+1} - \hat{V}_i\|$
9: $i = i + 1$
10: **end while**
11: $policy\_stable = true$
12: $i = 0$
13: **for** each $x \in \mathcal{X}$ **do**
14: $\hat{\pi}_{i+1}(x) = argmax_{u \in \mathcal{U}} \sum_{x' \in \mathcal{X}} Pr\{(x, u, x')\} \left(R(x, u) + \gamma V(x')\right)$
15: **if** $\hat{\pi}_i(x) \neq \hat{\pi}_{i+1}(x)$ **then**
16: $policy\_stable = false$
17: **end if**
18: **end for**
19: **if** *policy_stable* **then**
20: **return** policy $\hat{\pi}_i$
21: **else**
22: go to line 3
23: **end if**

On the other hand, Policy Iteration (PI) works by the process of policy improvement. Specifically, it starts with some initial random policy. Then, at each successive iteration, it evaluates the value function for the current policy, and performs a policy improvement step in which a new policy is generated by selecting a greedy action at each state, with respect to the values of the current policy. If the improved policy is the same as the policy improved upon, then we are assured that the optimal policy has been found.

The pseudocode for Policy Iteration is shown in Algorithm 2.

Remark 2.4. *Note that the subscript for value and policy estimations in Algorithm 1 and 2 respectively, refers to the current iteration of the corresponding Algorithm based on a fixed set of observations $x[k]$ and $u[k]$, and it has no relation with time steps k.* □

Most successful Reinforcement Learning algorithms are descended from one of the two Dynamic Programming algorithms described above, VI and PI. However, there are two major features distinguishing the RL setting from the traditional decision theoretic setting.

First, while in decision theory it is assumed that the environment model is fully known, in RL no such assumption is made. Second, in RL, the learning process is usually assumed to take place online, namely, concurrently with the accumulation of actual (or simulated) data acquired by the learning agent as it explores its environment. These two features make RL a significantly more difficult challenge, and place serious constraints on any potential RL algorithm. Probably the two best known RL algorithms, TD(λ) [38] and Q-learning [43], serve well to demonstrate how RL methods handle these constraints. For simplicity we assume that the state and action spaces are finite, and that the state values, or state-action values are stored explicitly in a lookup table.

TD(λ) is aimed at evaluating the value function for a given policy π. The input to the algorithm is a sequence of state-rewards couples, generated by the MDP controlled by the policy π. The idea in TD(λ) is to gradually improve value estimates by moving them towards the weighted average of multi-step lookahead estimates, which take into account the observed rewards. In the simplest case, of $\lambda = 0$, this amounts to moving the value estimate of the current state, $\hat{V}(x)$, toward the one-step lookahead estimate $R(x,u) + \gamma\hat{V}(x')$. Temporal difference (TD) methods avoid making direct use of the transition model by sampling from it.

The pseudocode for TD(0) is given in Algorithm 3. The update term $R(x[k],u[k]) + \gamma\hat{V}(x[k+1]) - \hat{V}(x[k])$ is referred to as the temporal difference at time $k+1$.

Algorithm 3 Tabular TD(0) algorithm

$\hat{V}(x[0]) = 0 \qquad \forall x \in \mathcal{X}$
for $k = 1, 2, ..., n$ **do**
  Observe samples $(x[k], R(x[k], u[k]), x[k+1])$
  $\hat{V}(x[k]) = \hat{V}(x[k]) + \alpha\left(R(x[k],u[k]) + \gamma\hat{V}(x[k+1]) - \hat{V}(x[k])\right)$
end for
return $\hat{V}$

In many cases, RL tasks are naturally divided into learning episodes. In such episodic learning tasks, the agent is placed at some (typically randomly chosen) initial state, and is then allowed to follow its policy until it reaches a terminal state, where the episode terminates and a new one may begin. A terminal state is modeled as a state with zero reward and with only self transitions, for any action. Then, if $x[k+1]$ is terminal, in the TD(0) update, as well as in algorithms presented in the sequel, we define $\hat{V}(x[k+1]) = 0$.

As any value estimation algorithm requires a policy improvement step, and it involves knowledge of the transition model which is typically unavailable. This is often solved by introducing state-action values (also known as Q-values), rather than just state values.

For a given policy π, the Q-value for the state-action pair $(x[k], u[k])$ is the expected discounted return over all trajectories starting from $x[k]$, for which the first action is $u[k]$, and with all subsequent actions chosen according to π. The optimal state value V^* is related to the optimal state-action value Q^* as

$$V^*(x[k]) = \max_{u \in \mathcal{U}} Q^*(x[k], u[k]). \tag{2.3.13}$$

As on TD algorithms, where expectations were replaced with actual samples, we can by analogy derive the Q-learning algorithm, which can be viewed as an asynchronous, stochastic version of VI, and whose pseudocode is shown in Figure 4.

Algorithm 4 Q-learning algorithm

1: $\hat{Q}(x[0], u[0]) = 0 \qquad \forall x \in \mathcal{X}, \quad u \in \mathcal{U}$
2: **for** $k = 1, 2, ..., n$ **do**
3: **Observe** $x[k], u[k], R(x[k], u[k]), x[k+1]$
4: $\hat{Q}(x[k], u[k]) = \hat{Q}(x[k], u[k]) + \alpha \left(R(x[k], u[k]) + \gamma \max_{u[k+1]} \hat{Q}(x[k+1], u[k+1]) - \hat{Q}(x[k], u[k]) \right)$
5: **end for**
6: **return** $\hat{Q}$

Then, by assuming $\hat{Q} = Q^*$ (i.e., Q-learning has converged to the optimum), then an optimal action for each state can be easily computed by a single maximization operation

$$\pi^*(x[k]) = arg \max_{u[k]} Q^*(x[k], u[k]), \tag{2.3.14}$$

with ties broken arbitrarily.

TD and Q-learning algorithms, as well as many other algorithms relying on a lookup-table representation, are useful in providing a proof-of-concept. However, real world problems can rarely be solved using such representations, due to the large, and sometimes infinite state and action spaces which characterize such problems. Since a tabular representation is unfeasible, it is necessary, in such problems, to use some form of function approximation to represent the value function and possibly also the policy, which for synthesis purposes of this current chapter, will be introduced where appropiate.

2.4 Summary

This chapter has provided basic definitions and results that will play a relevant role in the remainder of this thesis. Other known results of less widespread use can be found in the appendix. Nevertheless, most definitions and the notation introduced in this chapter and appendix are sometimes used without further comment in the forthcoming chapters. Therefore, if the reader finds an unfamiliar symbol, they should be able to find its definition in this chapter, in the appendix or in the list of symbols at the beginning of this thesis.

Chapter 3

STATE-FEEDBACK CONTROL

3.1 Introduction

In this chapter we will focus on state-feedback control for a given plant, whose dynamics will be assumed to be described by a linear model. Although later in the following chapters this assumption could be relaxed, we will still assume that the stabilizing controller to be designed is also linear in terms of the state of the plant and the state of its own dynamics. Complexity of the controller is usually desired to be kept as low as possible, since this has a direct impact on the required computation for generating control signals.

Then, this chapter will introduce the general case of state-feedback control, along with some particular cases that are often found in the literature. Later on, a stability analysis will be made for such cases, stating conditions that ensure that the control loop remains stable in the mean-square sense for a given plant.

Partial observability on the state of the plant will be tackled by substituting the true state for suitable Kalman filter estimations. Then, we will re-arrange appropiately the Kalman filter equations before we present the corresponding stability analysis when feedback is given on estimations instead of on the true state.

3.2 Modelling the controller

Recall that the plant G, at the moment, is assumed to be linear and strictly causal, i.e., its dynamics are described by

$$x[k+1] = Ax[k] + Bu[k] + v[k], \tag{3.2.1a}$$

$$y[k] = Cx[k] + w[k], \tag{3.2.1b}$$

with uncorrelated (Gaussian) zero-mean noises $v[k]$ and $w[k]$ with constant variances P_v and P_w respectively, and (A, B, C) are the state space variables with dimensions $A \in \mathbb{R}^{nx \times nx}$, $B \in \mathbb{R}^{nx \times nu}$ and $C \in \mathbb{R}^{ny \times nx}$, where nx, nu and ny stand for the number of states, number of control signals and number of outputs respectively.

Recall the classical state-feedback architecture shown in Figure 2.1 on Section 2.2, where $L \in \mathbb{R}^{n_u \times n_x}$ corresponds to a (matrix) gain that weights the plant state $x[k]$ in order to generate a control signal $u[k]$ for tracking reference $\bar{r}[k]$ at time k. Usually, on a state-feedback control scheme as shown in Figure 3.1, $r_L[k]$ (or $r[k]$ in Figure 3.2) is a pre-filtered

reference translated from the external reference $\bar{r}[k] \in \mathbb{R}^{ny}$ which is the desired value that should take $y[k]$ at time k, into a suitable vector with appropiate dimensions.

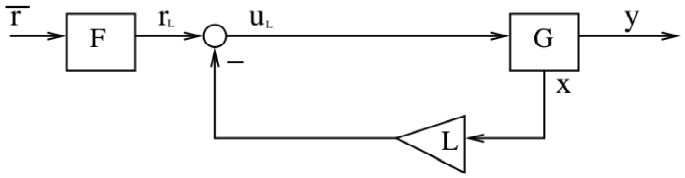


Figure 3.1: Classical state-feedback control scheme

It is well known from the literature, that a stabilizing controller (matrix $L \in \mathbb{R}^{nu \times nx}$) for the plant G, has to be designed such that the eigenvalues of $eig(A - BL)$ lie inside the unit circle. Moreover, if we want to focus also on tracking, that is getting $y[k]$ as close as possible to $\bar{r}[k]$, the prefilter F would be given by

$$F = -\left(C\left(A - BL - I\right)^{-1} B\right)^{-1}. \tag{3.2.2}$$

If we want to design a controller such that it has its own dynamics, we could consider a state-feedback control scheme as shown in Figure 3.2,

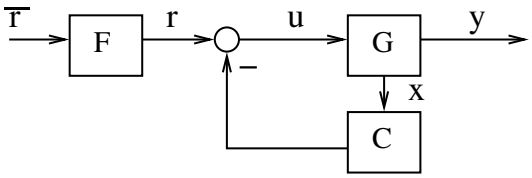


Figure 3.2: State-feedback control scheme

where we will consider again that the reference vector $r[k]$ is a pre-filtered version of $\bar{r}[k]$, resulting in a nu-elements vector representing the desired output $y[k]$ which is contained in $\bar{r}[k]$. Then, the controller is synthesized in C, and assuming linearity, it can be described in a state model by

$$x_c[k+1] = A_c x_c[k] + B_c x[k], \tag{3.2.3a}$$

$$u[k] = r[k] - \left(C_c x_c[k] + D_c x[k]\right), \tag{3.2.3b}$$

where the c subscript is set to stress the difference between matrices (A, B, C) from the plant model and (A_c, B_c, C_c, D_c) from the controller, as well as the state of the plant, $x[k]$, and the internal state of the controller itself, $x_c[k]$.

Note that once the controller has been designed, the pre-filter F for transforming reference $\bar{r}[k]$ into $r[k]$ should be chosen such that the transfer function from $\bar{r}[k]$ to $y[k]$ is unitary, in order to ensure stationary tracking. Indeed, prefilter F is given by

$$F = -\left(C\left(A - I - BD_c + BC_c\left(A_c - I\right)^{-1} B_c\right)^{-1} B\right)^{-1} \tag{3.2.4}$$

Remark 3.1. *It can be seen from* (3.2.3)*, that when $C_c = 0$ we have exactly the same state-feedback law as in the simpler case (static controller) shown in Figure 3.1, with D_c and L being equivalent. When this is the case, we could still have a dynamic controller, but the dynamics (and hence stability) of the controller itself do not play any role in the stability of the control loop, allowing the controller to be unstable, as long as it stabilizes the plant.*

In theory there is no problem with having an unstable controller that stabilizes the control loop, but in practice this represent a non-desirable choice, since in practical applications (like on Robotics), an unstable controller could lead the physical system (the robot respectively) to crash or become damaged.

Regarding the dimension of parameters of the state space model of the controller, note that $A_c \in \mathbb{R}^{nx_c \times nx_c}$, $B_c \in \mathbb{R}^{nx_c \times nx}$, $C_c \in \mathbb{R}^{nu \times nx_c}$ and $D_c \in \mathbb{R}^{nu \times nx}$, where nx_c is the number of elements of the internal state of the controller $x_c[k]$, stressing that is not necessary to have $nx_c = nx$.

Remark 3.2. *If we set the reference $r[k] = 0 \quad \forall k$, and $C_c = 0$, the problem is reduced to regulation. The main objective of the regulation problem is to make the state $x[k]$ decrease to zero as $k \to \infty$, which is the reason for having $r[k] = 0$.*

□

When tackling the partially observable case, due to missing elements from the state vector, or given that there could be high costs associated with measuring each element from the resulting state, suitable estimations have to be made. Such a sub-system providing estimations, will be called an *observer*. Then, the classical state-feedback scheme in Figure 3.1 which makes use of the estimated state instead of the true state is depicted in Figure 3.3, while our proposed approach is shown in Figure 3.4.

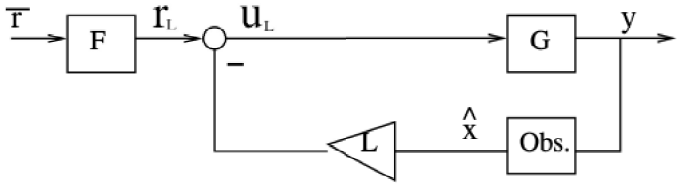


Figure 3.3: Classical (observed) state-feedback control scheme

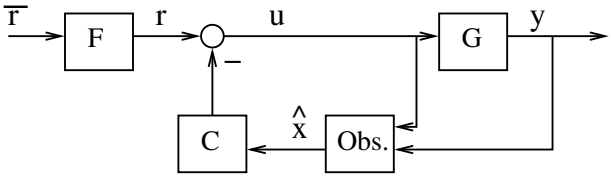


Figure 3.4: Proposed (observed) state-feedback control scheme

The observer will be modeled by a Kalman Filter, shown in Section 2.2, whose equations

could be re-arranged as

$$\hat{x}[k+1] = A\hat{x}[k] + Bu[k] + K[k]\left(y[k] - C\hat{x}[k]\right) \tag{3.2.5a}$$

$$K[k] = \left(C\left(AP[k-1]A^{\intercal} + P_v\right)C^{\intercal} + P_w\right)^{-1} \tag{3.2.5b}$$

noting that notation on time dependency for estimations has been reduced to $\hat{x}[k] = \hat{x}[k|k]$, and consequently the estimation error variance $P[k] = P[k|k]$.

Then, for the particular case of interest of observed state feedback, Kalman filter yields

$$\hat{x}[k+1] = \left(A - BD_c - K[k]C\right)\hat{x}[k] + Br[k] - BC_c x_c[k] + K[k]y[k] \tag{3.2.6a}$$

$$P[k+1] = \left(A - K[k]C\right)P[k]\left(A - K[k]C\right)^{\intercal} + P_v + K[k]P_w K^{\intercal}[k] \tag{3.2.6b}$$

$$K[k] = \left(C\left(AP[k-1]A^{\intercal} + P_v\right)C^{\intercal} + P_w\right)^{-1} \tag{3.2.6c}$$

It can be observed that when the plant achieves stationarity, kalman filter gain, $K[k]$ will converge to a constant value, in particular a matrix $K \in \mathbb{R}^{nx \times ny}$ whose elements are constant values.

3.3 Stability analysis

Before going deeper into the more general case, when the feedback loop is based on state estimations, we will analyze the state-feedback control scheme over the true state, which is expected to give some directions into what to do in the partially observable case.

Theorem 3.1. *Consider a strictly causal plant described by* (3.2.1)*, and a controller given by* (3.2.3)*. Then, a MSS controller will stabilize (in mean-square sense) the plant if and only if the eigenvalues of $\bar{A}$ lie inside the unit circle, where $\bar{A}$ is a block matrix given by*

$$\bar{A} = \begin{bmatrix} A - BD_c & -BC_c \\ B_c & A_c \end{bmatrix}. \tag{3.3.1}$$

Given that $\bar{A}$ is not symmetric nor triangular, sufficient and neccessary conditions on eigenvalues of matrices that compose $\bar{A}$ cannot be obtained. Nevertheless, the following expression must hold for representing a sufficient condition in order to obtain a stable controller that stabilizes the plant, with finite $x[0]$, $x_c[0]$ and $(A - BD_c)$ non-singular,

$$\max\left(|eig(A - BD_c)|, |eig(A_c + B_c(A - BD_c)^{-1}BC_c)|\right) < 1, \tag{3.3.2}$$

where making a slight abuse of notation, $eig(\cdot)$ denotes the greatest eigenvalue of $\cdot$.

■

Proof: Consider an augmented state vector $\bar{x}[k] = \begin{bmatrix} x^{\intercal}[k] & x_c^{\intercal}[k] \end{bmatrix}^{\intercal}$. Then, the augmented system will be given by

$$\bar{x}[k+1] = \underbrace{\begin{bmatrix} A - BD_c & -BC_c \\ B_c & A_c \end{bmatrix}}_{\bar{A}}\bar{x}[k] + \underbrace{\begin{bmatrix} B \\ 0_{nx \times nu} \end{bmatrix}}_{\bar{B}} r[k] + \underbrace{\begin{bmatrix} v[k] \\ 0_{nx_c \times 1} \end{bmatrix}}_{\bar{v}[k]}, \tag{3.3.3}$$

where $0_{nx_c \times 1}$ is a zero-entries column vector with nx_c elements.

Then, unrolling the system in (3.3.3) in terms of the initial (augmented) state, yields

$$\bar{x}[k+1] = \bar{A}^{k+1}\bar{x}[0] + \sum_{i=0}^{k} \bar{A}^i \left(\bar{B}r[k-i] + \bar{v}[k-i]\right), \tag{3.3.4}$$

where given that $v[k]$ is zero-mean white noise, its expectation is given by

$$\xi\{\bar{x}[k+1]\} = \xi\{\bar{A}^{k+1}\}\bar{x}[0] + \sum_{i=0}^{k} \xi\{\bar{A}^i \bar{B}r[k-i]\}, \tag{3.3.5}$$

which corresponds to a matrix power series, so for having finite $\xi\{\bar{x}[k+1]\}$ in the limit when $k \to \infty$, $\bar{A}$ has to be such that

$$\|\bar{A}\|_2 < 1, \tag{3.3.6}$$

which in terms of its greatest eigenvalue, the condition is turned into

$$|eig(\bar{A})| < 1. \tag{3.3.7}$$

It can be seen that making an analogous analysis for the second order moment of $\bar{x}[k]$ yields the same condition for $\bar{A}$, given that the only term affecting variance of $\bar{v}[k]$ is $v[k]$ and its variance is already assumed to be finite.

Then, in order to obtain a sufficient condition in terms of the matrices in $\bar{A}$, it can be decomposed by using LU factorization,

$$\bar{A} = \underbrace{\begin{bmatrix} I_{nx \times nx} & 0_{nx \times nx_c} \\ B_c(A - BD_c)^{-1} & I_{nx_c \times nx_c} \end{bmatrix}}_{L} \cdot \underbrace{\begin{bmatrix} A - BD_c & -BC_c \\ 0_{nx_c \times nx} & B_c(A - BD_c)^{-1}BC_c + A_c \end{bmatrix}}_{U} \tag{3.3.8}$$

An upper bound on the eigenvalues of a matrix product $L \cdot U$ can be given by (see Lemma A.2 on Appendix),

$$eig(\bar{A}) \leq eig(U), \tag{3.3.9}$$

by noting that all eigenvalues of L are 1.

Finally, given that the eigenvalues of a diagonal block matrix are given by the union of the eigenvalues of such matrices,

$$eig(U) = eig(A - BD_c) \cup eig(A_c + B_c(A - BD_c)^{-1}BC_c), \tag{3.3.10}$$

so in order to ensure $|eig(\bar{A})| < 1$, we restrict our attention to its upper bound,

$$\max\left(|eig(A - BD_c)|, |eig(A_c + B_c(A - BD_c)^{-1}BC_c)|\right) < 1 \tag{3.3.11}$$

□□

Recall that eigenvalues have a direct relationship with the speed that the time-response achieves stationarity. Indeed, the larger the eigenvalues are (remaining inside the unit circle), means the transient of the time-response will disappear faster. If on the other hand, the eigenvalues are located outside the unit circle, the time-response will oscillate. Then,

assume for a moment that $B_c = 0$ or $C_c = 0$, so it is clear to see that the controller can not be internally unstable and D_c has to be designed such that $(A - BD_c)$ have eigenvalues inside the unit circle, otherwise stability of the loop and of the controller itself can not be guaranteed.

Theorem 3.2. *Consider a strictly causal plant described by* (3.2.1)*, and a controller given by* (3.2.3)*. Then, a MSS controller will stabilize (in mean-square sense) the plant if and only if the eigenvalues of* $\bar{\bar{A}}$ *lie inside the unit circle, where* $\bar{\bar{A}}$ *is a block matrix given by*

$$\bar{\bar{A}} = \begin{bmatrix} A - KC & 0_{nx \times nx_c} & 0_{nx \times nx} \\ -B_c & A_c & B_c \\ BD_c & -BC_c & A - BD_c \end{bmatrix}. \tag{3.3.12}$$

As conditions established in Theorem 3.1, the following expression must hold for representing a sufficient condition in order to obtain a stable controller that stabilizes the plant, with finite $x[0]$*,* $x_c[0]$*, arbitrary initial (finite) guess* $\hat{x}[0]$ *and* $P[0]$*, and* $(A - BD_c)$ *non-singular,*

$$\max\left(|eig\left(A - KC\right)|, |eig(A_c)|, |eig\left(A - BD_c\right)|\right) < 1, \tag{3.3.13}$$

where time dependency of K *has been omitted for simplicity.* ■

Proof: The convergence of the expected value of the state will be analyzed, since analysis of second order moments yields the same conditions for this case again, as in Theorem 3.1. In a similar form, we can build an augmented system,

$$\bar{\bar{x}}[k+1] = \underbrace{\begin{bmatrix} A - KC & 0_{nx \times nx_c} & 0_{nx \times nx} \\ -B_c & A_c & B_c \\ BD_c & -BC_c & A - BD_c \end{bmatrix}}_{\bar{\bar{A}}} \bar{\bar{x}}[k] + \underbrace{\begin{bmatrix} 0_{nx \times nu} \\ 0_{nx_c \times nu} \\ B \end{bmatrix}}_{\bar{\bar{B}}} r[k] + \underbrace{\begin{bmatrix} -Kw[k] \\ 0_{nx_c \times 1} \\ v[k] \end{bmatrix}}_{\bar{\bar{v}}[k]}, \tag{3.3.14}$$

where $\bar{\bar{x}}[k] = \begin{bmatrix} \tilde{x}^\intercal[k] & x_c^\intercal[k] & x^\intercal[k] \end{bmatrix}^\intercal$, and $\tilde{x}[k]$ corresponds to the estimation error at time k,

$$\tilde{x}[k] = x[k] - \hat{x}[k]. \tag{3.3.15}$$

Then, unrolling the system in terms of its initial state, yields

$$\bar{\bar{x}}[k+1] = \bar{\bar{A}}^{k+1}\bar{\bar{x}}[0] + \sum_{i=0}^{k} \bar{\bar{A}}^i \left(\bar{\bar{B}} r[k-i] + \bar{\bar{v}}[k-i]\right), \tag{3.3.16}$$

where given that both $v[k]$ and $w[k]$ are zero-mean white noises, its expectation is given by

$$\xi\{\bar{\bar{x}}[k+1]\} = \xi\{\bar{\bar{A}}^{k+1}\}\bar{\bar{x}}[0] + \sum_{i=0}^{k} \xi\{\bar{\bar{A}}^i \bar{\bar{B}} r[k-i]\}, \tag{3.3.17}$$

so for having finite $\xi\{\bar{\bar{x}}[k+1]\}$ in the limit when $k \to \infty$,

$$|\max eig(\bar{\bar{A}})| < 1. \tag{3.3.18}$$

Then, in order to obtain a sufficient condition in terms of the matrices on $\bar{\bar{A}}$, it can be decomposed by using LU factorization,

$$\bar{\bar{A}} = L \cdot U, \tag{3.3.19}$$

where

$$L = \begin{bmatrix} I_{nx\times nx} & 0_{nx\times nx_c} & 0_{nx\times nx} \\ 0_{nx_c\times nx} & I_{nx_c\times nx_c} & B_c\left(A - BD_c\right)^{-1} \\ 0_{nx\times nx} & 0_{nx\times nx_c} & I_{nx\times nx} \end{bmatrix}, \tag{3.3.20a}$$

$$U = \begin{bmatrix} A - KC & 0_{nx\times nx_c} & 0_{nx\times nx} \\ -B_c & A_c & 0_{nx_c\times nx} \\ BD_c & -BC_c & A - BD_c \end{bmatrix}. \tag{3.3.20b}$$

Then, given that the eigenvalues of L are all equal to 1, the upper bound for the eigenvalues of $\bar{\bar{A}}$ is given by

$$eig(\bar{A}) \leq eig(U), \tag{3.3.21}$$

Finally,

$$eig(U) = eig(A - KC) \cup eig(A_c) \cup eig(A - BD_c), \tag{3.3.22}$$

so in order to ensure $|eig(\bar{\bar{A}})| < 1$, we restrict our attention to its upper bound,

$$\max\left(|eig(A - KC)|, |eig(A_c)|, |eig(A - BD_c)|\right) < 1 \tag{3.3.23}$$

□□

Theorem 3.2 estates that if the controller is internally stable, then the observer has to be designed such that the estimation error remains bounded (zero for an optimal filter in a steady state). Since the observer is given by a Kalman filter, which gives stable estimations, we focus on the choice of A_c and D_c for satisfying conditions in Theorem 3.2 .

3.4 Simulation results

The analysis made in this Chapter make use of a strong assumption of full knowledge of the model of the plant, in particular, knowledge about variance of both process and measurement noise, and model parameters A, B and C. This assumption can not be fulfilled in practical applications, since modelling error or approximations can be very harmful for the controller design, so in the following chapters we will focus on learning those (or part of those) parameters. Before validating results shown in this Chapter, we will first implement a simulation of an arbitrary plant, whose model would be such that it allow us to show some interesting properties in the results developed.

First, consider a SISO plant (one-input one-output, $nu = 1, ny = 1$) be given by

$$x[k+1] = Ax[k] + Bu[k] + v[k], \tag{3.4.1a}$$

$$y[k] = Cx[k] + w[k], \tag{3.4.1b}$$

where as before, $v[k]$ and $w[k]$ are the process and measurement noise respectively, zero-mean and unitary variance white noises.

Let A, B and C be given by

$$A = \begin{bmatrix} 0.5 & 0 \\ 0.7 & 1.2 \end{bmatrix}, \tag{3.4.2a}$$

$$B = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}, \tag{3.4.2b}$$

$$C = \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{3.4.2c}$$

It can be observed that the plant is internally unstable, since one of its eigenvalues lie outside the unit circle. Then, parameters A_c, B_c, C_c and D_c of the controller have to be found for generating a (scalar) control signal $u[k]$,

$$x_c[k+1] = A_c x_c[k] + B_c x[k], \tag{3.4.3a}$$

$$u[k] = r[k] - \left(C_c x_c[k] + D_c x[k]\right), \tag{3.4.3b}$$

whose matrices were set to

$$A_c = 0.4 \qquad B_c = \begin{bmatrix} 1 & -1.52 \end{bmatrix},$$
$$C_c = -0.5 \qquad D_c = \begin{bmatrix} 0.3 & 2.1 \end{bmatrix}.$$

These values were chosen such that the augmented system matrix $\bar{A}$ from Theorem 3.1 has prescribed eigenvalues. Indeed, $eig(\bar{A}) \in \{0.5, 0.59, 0.8\}$.

Remark 3.3. *As the reader may infer, hand-tuning of a dynamic state-feedback controller can be time consuming, so actually the previous parameters were obtained from the proposed reinforcement learning approach for tuning the controller parameters under this same plant in the following Chapter.* □

Note from (3.4.3), that we have assumed for this first example that the feedback is made on the true state. The reference has been set to

$$r[k] = \begin{cases} 0 & k < 60 \\ 10 & k \geq 60 \end{cases} \tag{3.4.4}$$

We compare this control scheme with the classical state-feedback architecture shown in Figure 3.1, with output $y_L[k]$, such that the control signal is given by

$$u_L[k] = r_L[k] - Lx[k], \tag{3.4.5}$$

and L is set to

$$L = \begin{bmatrix} 0.3 & 4 \end{bmatrix},$$

such that $eig(A - BL) \in \{0.5, 0.8\}$.

As expected, from analyzing eigenvalues of $\bar{A}$ and $A - BL$, we see that the closed loop system achieves stationarity with the same velocity (given by its greatest eigenvalue, which represents the velocity of the slowest disappearing natural mode), but with different values for D_c and L. This is possible because in Figure 3.2, we added degrees of freedom in the

controller with respect to the scheme shown in Figure 3.1, at the cost of introducing an additional natural mode in the closed loop response.

As shown in simulation results depicted in Figure 3.5, the introduced degrees of freedom in the controller allow us to set matrices values such that the closed loop response achieves stationarity as fast as desired, and confines the control signal (output of controller) to be smaller than in the classical architecture.

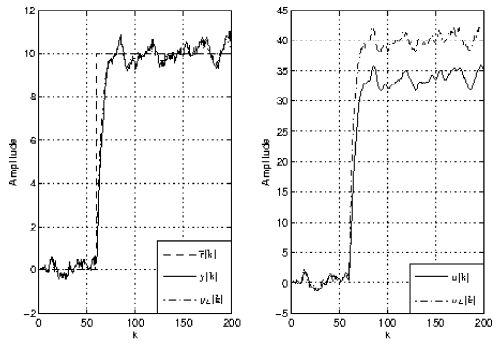


Figure 3.5: Full state-feedback, step reference tracking

Figure 3.6 shows the same comparison for a sinusoidal reference signal, where the same results remain, namely, speed of convergence is the same although the dynamic controller achieves lower peaks in the control signal, despite the difference in the values of D_c and L, due to the additional degrees of freedom. Nevertheless, the reader should note that reference tracking in this case will be achievable only if the closed loop dynamics are fast enough to keep track of changes in the reference.

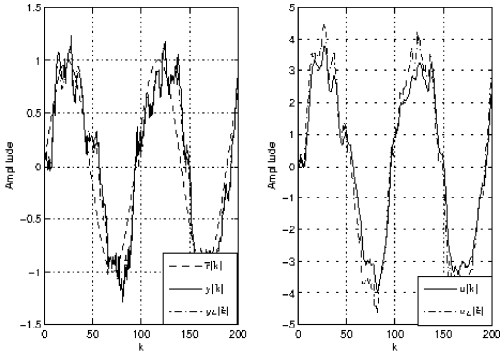


Figure 3.6: Full state-feedback, sinusoidal reference tracking

Figure 3.7 shows the external reference and its pre-filtered version injected into the control system in both previous examples. Given that the pre-filter F depends on the architecture of the state-feedback control loop, the pre-filtered version is different for each

case, so according to control signals given by (3.4.3b) and (3.4.5), the input of the plant in each case will have a different amplitude.

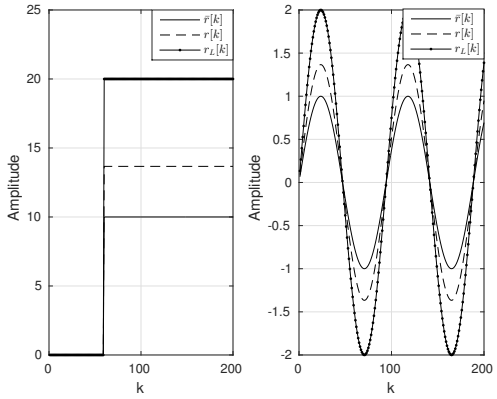


Figure 3.7: Pre-filtered reference

For the partially observable case, we set exactly the same values for the controller parameters, which now are given by

$$x_c[k] = A_c x_c[k] + B_c \hat{x}[k], \tag{3.4.6a}$$

$$u[k] = r[k] - \left(C_c x_c[k] + D_c \hat{x}[k]\right), \tag{3.4.6b}$$

and their performance was compared with the (observed) state-feedback control law

$$u[k] = r_L[k] - L\hat{x}[k], \tag{3.4.7}$$

Results are shown in Figure 3.8 and 3.9 for a step reference and sinusoidal reference respectively.

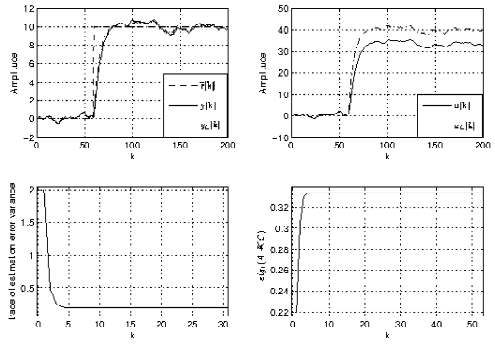


Figure 3.8: Observed state-feedback, step reference tracking

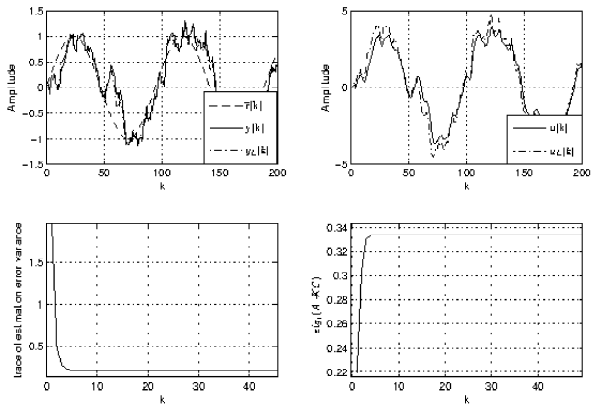


Figure 3.9: Observed state-feedback, sinusoidal reference tracking

Figure 3.8 and 3.9, also shows the evolution of the (trace) estimation error variance, and the evolution of one of the eigenvalues of $(A - KC)$ (the other one is fixed due to the nature of A chosen in these examples), noting that it remains inside the unit circle, and as expected, it achieves its stationary value as soon as the estimation error variance settles.

3.5 Summary

This chapter has introduced a rarely used and more general scheme for state-feedback control, including a controller allowed to have its own dynamics instead of having just a feedback gain. Also, conditions that ensure mean-square stability for a certain plant were given.

Simulation results illustrates validation of expected theoretical results, showing that the proposed controller scheme performs the same as the classical one in terms of the tracking error, but this is achieved at a lower effort in terms of the control signal, at the cost of tuning more parameters, a problem that will be tackled in the following chapter.

Simulation results were implemented on an arbitrary plant for both cases: state-feedback over the true state, and a feedback loop based on state estimations from a Kalman filter.

Chapter 4

ADAPTIVE CONTROL USING REINFORCEMENT LEARNING

State feedback controllers are appealing due to their structural simplicity. Nevertheless, when stabilizing a given plant, their dynamics could lead the gain of a static feedback controller to take higher values than desired. On the other hand, a dynamic state feedback controller is capable of achieving the same or even better performance by introducing additional parameters into the model to be designed. In this thesis, the Linear Quadratic Tracking problem will be tackled using a (linear) dynamic state feedback controller, whose parameters will be chosen by means of applying reinforcement learning techniques, which have been proved to be specially useful when the model of the plant to be controlled is unknown or inaccurate.

4.1 Introduction

Reinforcement learning (RL) is tipically concerned with solving sequential decision problems modelled by Markov Decision Processes (MDPs). Applications of RL, as a research field within Machine Learning, has been extended to areas such as Robotics [23] or Control Theory [17, 25, 40], by means of choosing a suitable representation of the problem to be solved.

The LQR, i.e., the regulator problem when the system is assumed to be linear, and the performance index is given in terms of a quadratic function [2], is particularly appealing given that its solution is obtained by solving an algebraic Riccati equation (ARE). Then, PI algorithms basically start with an admissible control policy and then iterate between policy evaluation and policy improvement steps until variations on the policy or the specified value function are negligible, as seen in [11, 26, 41].

On the other hand, the linear quadratic tracking (LQT) problem also assumes a linear model for the process dynamics and a quadratic function for the performance index, but the main objective is to design a controller such that the measured output of the process to be controlled, follows an exogenous reference signal, so the LQR could be considered as a particular case of the LQT problem. Although, as mentioned before, RL algorithms have been extensively applied for solving the LQR problem, the LQT has not received much attention in the literature mainly because for most reference signals the infinite horizon cost

becomes unbounded [7]. Work in [30] tackles the problem in the continous time domain by solving an augmented ARE obtained from the original system dynamics and the reference trajectory dynamics, while [22] take a similar approach for the discrete-time case, where a Q-learning algorithm is obtained for solving the LQT problem without any model knowledge.

Then, when considering noisy systems, the performance index and notions of stability have to be modified accordingly. This problem has been extensively treated in literature from the classical control, or model-based approach [13, 18, 47], unlike on the learning paradigm. [24] used neural networks for reducing calculus efforts in providing optimal control for the stochastic LQR, while other works focus on relaxing assumptions on the ARE under different scenarios, but still requiring knowledge of the system dynamics [12, 49].

The remainder of this chapter is organized as follows: Section 4.2 shows the synthesis procedure for obtaining a stabilizing dynamic controller for minimizing the LQT performance criteria, and the main results. Section 4.3 gives an illustration of experimental results in a simulation environment.

4.2 Controller Synthesis using Reinforcement Learning

When tackling the problem of making a system follow a given trajectory, the main objective is to find the appropiate control signals (or actions that the controller should generate) for making a variable of the system to be controlled to keep track of the desired reference value.

Then, as seen in the previous chapter, a control signal $u[k]$ must be generated and used as input of the controlled system, which has an output $y[k]$ that should keep track of the reference $\bar{r}[k]$ at time k. Therefore, a model is needed.

One of the typical options is to assume a given structure for the model, and then tune its parameters until the model and real system dynamics match. Another option, could be to use physical laws for building a model and set relations between all the variables of the system.

Recall that a (stochastic) linear, discrete-time and strictly causal system has a state space representation given by (2.2.22), i.e.:

$$x[k+1] = Ax[k] + Bu[k] + v[k], \tag{4.2.1a}$$

$$y[k] = Cx[k] + w[k], \tag{4.2.1b}$$

where $x[k] \in \mathbb{R}^{n_x}$, $y[k] \in \mathbb{R}^{n_y}$ and $u[k] \in \mathbb{R}^{n_u}$ corresponds to the internal state, measured output and control signal respectively at time k, and A,B,C and D are (usually) known matrices of appropiate dimensions, while $v[k]$ and $w[k]$ are uncorrelated (gaussian) zero-mean white noises, namely process and measurement noise with constant variance P_v and P_w respectively.

When considering a (static) linear state feedback controller, the control law would be given by

$$u[k] = F\bar{r}[k] - Lx[k], \tag{4.2.2}$$

where $L \in \mathbb{R}^{n_u \times n_x}$ stands for the feedback gain, and F is a pre-filter such that $y[k]$ gets as close as possible to $\bar{r}[k]$.

Then, the performance index for the stochastic infinite-horizon LQT problem at time k will be given by

$$J[k] = \xi\{\sum_{i=k}^{\infty} (\bar{r}[i] - y[i])^{\intercal} Q (\bar{r}[i] - y[i]) + u^{\intercal}[i] R u[i]\}, \tag{4.2.3}$$

with $Q > 0$ and $R \geq 0$ weighting matrices of appropiate dimensions and last (quadratic) term is considered for avoiding a high-amplitude control signal.

Since we are dealing with stochastic systems, an appropiate notion of stability is given by mean-square stability (see Definition 2.4).

Lemma 4.2.1.

The process given by (2.2.22)*, whose state is given by* $x[k]$ *at time* k*, will be mean-square stable (MSS) if and only if*

$$\lim_{k \to \infty} |\xi\{x[k] x^{\intercal}[k]\}| < \infty, \tag{4.2.4}$$

independently of the initial state $x[0] = x_0$*.*

Moreover, the controller (4.2.2) *stabilizes the system in* (2.2.22)*, if the reference* $\bar{r}[k]$ *decays asymptotically to zero, and* L *is such that the closed-loop eigenvalues are inside the unit circle, i.e.,*

$$\lim_{k \to \infty} \bar{r}[k] = 0, \tag{4.2.5a}$$

$$|\bar{\lambda}(A - BL)| < 1. \tag{4.2.5b}$$

■

Proof:

For the definition of mean square stability, the reader is encouraged to see [35]. Although the conditions set on Lemma 4.2.1 can be found in standard stochastic control theory literature, we show how these conditions are obtained for sake of clarity.

By replacing (4.2.2) in (2.2.22), the second order moments matrix of $M_x[k] = \xi\{x[k] x^{\intercal}[k]\}$ are given by

$$M_x[k] = (A - BL) M_x[k-1] (A - BL)^{\intercal} + B M_{\bar{r}}[k-1] B^{\intercal} + P_v, \tag{4.2.6}$$

with P_v according to (2.2.22), and $M_{\bar{r}}[k] = \xi\{\bar{r}[k] \bar{r}^{\intercal}[k]\}$.

Then, in terms of the initial state $x[0] = x_0$,

$$M_x[k] = (A - BL)^k M_x[0] (A - BL)^{k^{\intercal}} + \sum_{i=1}^{k} (A - BL)^{i-1} (B M_{\bar{r}}[k-i] B^{\intercal} + P_v) (A - BL)^{i-1^{\intercal}},$$

where it can be seen that for the system being MSS, it is necessary to get $M_{\bar{r}}[k]$ bounded as k grows to infinity, so $r[k]$ (or $r_L[k]$ depending on the state-feedback architecture), and therefore $\bar{r}[k]$ has to decay asymptotically. Since the factor $(A - BL)$ is part of a matrix power series, its spectral radium has to be less than unit, which directly involves

$$|\bar{\lambda}(A - BL)| < 1. \tag{4.2.7}$$

□□□

The asymptotically decaying assumption in reference $\bar{r}$ limits the class of trajectories to be used, and more importantly, sense of minimality in (4.2.3) is lost. Therefore, as in [22], the following section will introduce a discounted performance index for the LQT setup, and assuming the reference is being generated by a system T_r.

4.2.1 Problem Formulation

Recall the dynamic state feedback controller in the previous chapter, allowing it to have its own dynamics component in the resulting control signal, but still in the realm of linear controllers:

$$x_c[k+1] = A_c x_c[k] + B_c x[k], \tag{4.2.8a}$$
$$u[k] = F\bar{r}[k] - \left(C_c x_c[k] + D_c x[k]\right), \tag{4.2.8b}$$

where the c subscript is set to stress the difference between matrices (A, B, C) from the plant model and (A_c, B_c, C_c, D_c) from the controller, as well as the state of the plant, $x[k]$, and the internal state of the controller itself, $x_c[k]$. Also, there is a prefilter F for transforming reference $\bar{r}[k]$ into $r[k]$ which should be chosen such that the transfer function from $\bar{r}[k]$ to $y[k]$ is unitary, in order to ensure stationary tracking.

Then, the performance index for the stochastic infinite-horizon LQT problem at time k will be given by

$$J[k] = \xi\{\sum_{i=k}^{\infty} \left(\bar{r}[i] - y[i]\right)^{\intercal} Q \left(\bar{r}[i] - y[i]\right) + u^{\intercal}[i] R u[i]\}, \tag{4.2.9}$$

with $Q > 0$ and $R \geq 0$ weighting matrices of appropiate dimensions.

The asymptotically decaying reference $\bar{r}[k]$ requirement is also necessary for convergence of the sum in the performance index for the stochastic LQT problem, as can be seen in (4.2.9). This requirement can be relaxed when introducing a discount factor, $\gamma \in (0, 1)$, such that

$$J[k] = \xi\{\sum_{i=k}^{\infty} \gamma^{i-k} \left(z^{\intercal}[i] Q_a z[i] + u^{\intercal}[i] R u[i]\right)\}, \tag{4.2.10}$$

with

$$z[k] = \begin{bmatrix} \left(\bar{r}[k] - y[k]\right) \\ x_c[k] \end{bmatrix}, \quad Q_a = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}, \tag{4.2.11}$$

where both Q_1 and Q_2 are positive definite matrices, penalizing the control error and avoiding to get the dynamics of the controller itself boundless respectively.

4.2.2 PI for solving the stochastic LQT with known dynamics

In order to make the LQT problem look more like a RL problem, let the value function $V(X[k])$ be

$$V(X[k]) = J[k], \tag{4.2.12}$$

where $X[k]$ stands for an augmented state vector containing the internal state of the process to be controlled, the state of the controller itself and the exogenous reference, i.e.,

$$X[k] = \begin{bmatrix} x^\intercal[k] & x_c^\intercal[k] & \bar{r}^\intercal[k] \end{bmatrix}^\intercal. \tag{4.2.13}$$

Then, the value function can be written as

$$V(X[k]) = \xi\{\sum_{i=k}^{\infty} \gamma^{i-k} \left(tr\left(P_w Q_1\right) + X^\intercal[i]\bar{Q}X[i] + u^\intercal[i]Ru[i]\right)\}, \tag{4.2.14}$$

with Q_1 as on (4.2.11) and $\bar{Q}$ given by

$$\bar{Q} = \begin{bmatrix} C^\intercal Q_1 C & 0 & -C^\intercal Q_1 \\ 0 & Q_2 & 0 \\ -Q_1 C & 0 & Q_1 \end{bmatrix}. \tag{4.2.15}$$

Theorem 4.2.1.
Consider the control law

$$u[k] = F\bar{r}[k] - \left(C_c x_c[k] + D_c x[k]\right), \tag{4.2.16}$$

with $x_c[k]$ as described in (4.2.8), and $\bar{r}[k]$ produced by the model

$$\bar{r}[k+1] = T_r \bar{r}[k], \tag{4.2.17}$$

Then, assuming the optimal value function is quadratic in the augmented state vector, i.e.,

$$V^*(X[k]) = X^\intercal[k]PX[k] + g[k], \tag{4.2.18}$$

for some stationary and symmetric matrix $P > 0$, and $g[k]$ such that

$$g[k+1] = \left(\frac{1}{\gamma}\right) g[k] + tr\left(\frac{1}{\gamma} P_w Q_1 + P_v P_{11}\right), \tag{4.2.19}$$

parameters (F, C_c, D_c) will be given by

$$F = \gamma Z^{-1} B^\intercal P_{13} T_r, \tag{4.2.20a}$$
$$C_c = -\gamma Z^{-1} B^\intercal P_{12} A_c, \tag{4.2.20b}$$
$$D_c = -\gamma Z^{-1} M, \tag{4.2.20c}$$

with

$$Z = \left(R + \gamma B^\intercal P_{11} B\right), \tag{4.2.21a}$$
$$M = \left(A^\intercal P_{11} B + B_c^\intercal P_{21} B\right), \tag{4.2.21b}$$

and each matrix P_{ij} from P on (4.2.18) *is such that*

$$\begin{aligned}
P_{11} &= C^\intercal Q_1 C + \gamma\left(A^\intercal P_{11} A + B_c^\intercal P_{21} A\right. && \text{(4.2.22a)}\\
&\qquad \left. + A^\intercal P_{12} B_c + B_c^\intercal P_{12} B_c\right) - \gamma^2 M Z^{-1} M^\intercal,\\
P_{12} &= \gamma\left(A^\intercal P_{12} A_c + B_c^\intercal P_{22} A_c\right) - \gamma^2 M Z^{-1} B^\intercal P_{12} A_c, && \text{(4.2.22b)}\\
P_{22} &= Q_2 + \gamma A_c^\intercal P_{22} A_c - \gamma^2 A_c^\intercal P_{21} B Z^{-1} B^\intercal P_{12} A_c, && \text{(4.2.22c)}\\
P_{13} &= -C^\intercal Q_1 + \gamma\left(A^\intercal P_{13} T_r + B_c^\intercal P_{23} T_r\right) && \text{(4.2.22d)}\\
&\qquad - \gamma^2 M Z^{-1} B^\intercal P_{12} F,\\
P_{23} &= \gamma A_c^\intercal P_{23} T_r - \gamma^2 A_c^\intercal P_{21} B Z^{-1} B^\intercal P_{13} T_r, && \text{(4.2.22e)}
\end{aligned}$$

where $P_{ij} = P_{ji}^\intercal$, as a consequence of the symmetry of P.

■

Proof:

Value function in (4.2.14) can be rewritten as

$$V\left(X[k]\right) = tr\left(P_w Q_1\right) + \xi\{X^\intercal[k]\bar{Q}X[k] + u^\intercal[k]Ru[k] + \gamma V\left(X[k+1]\right)\}. \tag{4.2.23}$$

On the other hand, since the value function is assumed to be quadratic in terms of the augmented state vector,

$$V\left(X[k+1]\right) = X^\intercal[k+1]PX[k+1] + g[k+1], \tag{4.2.24}$$

but from (4.2.13) we have

$$X[k+1] = \begin{bmatrix} x^\intercal[k+1] & x_c^\intercal[k+1] & \bar{r}^\intercal[k+1] \end{bmatrix}^\intercal. \tag{4.2.25}$$

Then, by replacing (2.2.22), (4.2.8) and (4.2.17) into (4.2.25), we have

$$\begin{aligned}
V\left(X[k]\right) = tr\left(P_w Q_1 + \gamma P_v P_{11}\right) + \xi\{&\gamma g[k+1]\\
&+ \left(H^\intercal + u^\intercal\right) Z \left(H + u\right) + X^\intercal[k]\bar{P}X[k]\}, \qquad \text{(4.2.26)}
\end{aligned}$$

where Z is defined as in (4.2.21a), H is given by

$$H = -\gamma Z^{-1}\left(M^\intercal x[k] + B^\intercal P_{12} A_c x_c[k] + B^\intercal P_{13} T_r \bar{r}[k]\right), \tag{4.2.27}$$

with M as in (4.2.21b), and $\bar{P}$ is given by

$$\bar{P} = \begin{bmatrix} \bar{P}_{11} & \bar{P}_{21}^\intercal & \bar{P}_{31}^\intercal \\ \bar{P}_{21} & \bar{P}_{22} & \bar{P}_{32}^\intercal \\ \bar{P}_{31} & \bar{P}_{32} & \bar{P}_{33} \end{bmatrix}, \tag{4.2.28}$$

where each matrix $\bar{P}_{ij}$, an updated value of P_{ij} is such that

$$\begin{aligned}
\bar{P}_{11} = C^\intercal Q_1 C + \gamma\left(A^\intercal P_{11} A + B_c^\intercal P_{21} A + A^\intercal P_{12} B_c\right.\\
\left. + B_c^\intercal P_{22} B_c\right) - \gamma^2 M Z^{-1} M^\intercal, \qquad \text{(4.2.29a)}
\end{aligned}$$

$$\bar{P}_{12} = (A_c^\intercal P_{21} A + A_c^\intercal P_{22} B_c) - \gamma^2 A_c^\intercal P_{21} B Z^{-1} M^\intercal, \tag{4.2.29b}$$

$$\bar{P}_{22} = Q_2 + \gamma A_c^\intercal P_{22} A_c - \gamma^2 A_c^\intercal P_{21} B Z^{-1} B^\intercal P_{12} A_c, \tag{4.2.29c}$$

$$\begin{aligned}\bar{P}_{31} = &-Q_1 C + \gamma \left(T_r^\intercal P_{31} A + T_r^\intercal P_{32} B_c\right) \\ &- \gamma^2 T_r^\intercal P_{31} B Z^{-1} \left(B^\intercal P_{11} A + B^\intercal P_{12} B_c\right),\end{aligned} \tag{4.2.29d}$$

$$\bar{P}_{32} = \gamma T_r^\intercal P_{32} A_c - \gamma^2 T_r^\intercal P_{31} B Z^{-1} B^\intercal P_{12} A_c, \tag{4.2.29e}$$

$$\bar{P}_{33} = Q_1 + \gamma T_r^\intercal P_{33} T_r - \gamma^2 T_r^\intercal P_{31} B Z^{-1} B^\intercal P_{13} T_r. \tag{4.2.29f}$$

Finally, for minimizing the expression in (4.2.26), it can be seen that the optimal control law $u[k]$ has to be chosen such that

$$u[k] = \gamma Z^{-1} \left(M^\intercal x[k] + B^\intercal P_{12} A_c x_c[k] + B^\intercal P_{13} T_r \bar{r}[k]\right) \tag{4.2.30}$$

so comparing terms in (4.2.16) and (4.2.30) yields (4.2.20).

In a similar form, by replacing (4.2.18) into (4.2.26), and comparing both sides of the expression, it can be found

$$g[k] = \gamma g[k+1] + tr\left(P_w Q_1 + \gamma P_v P_{11}\right), \tag{4.2.31}$$

so a little algebra yields (4.2.19).

□□□

Remark 4.2.1. *The assumption that $\bar{r}[k]$ is generated by the model described in* (4.2.17)*, is valid for a large class of useful trajectories, such as a unit step signal, sinusoidal waveforms and more.*

Then, Algorithm 5 shows the policy evaluation and improvement steps for the l-th iteration of PI.

It can be seen that Algorithm 5 can be implemented online, but all dynamics, including the generator model for the exogenous reference signal, have to be known. Despite this fact, expressions can be easily computed, and we can achieve a state feedback controller with its own (stable) dynamics by just designing two parameters (A_c and B_c, since the rest depend on these) in place of the classical standard case with just one parameter, which could lead to a high gain for some processes.

Algorithm 5 Dynamic State Feedback Controller Design

1: **(Policy Evaluation)** Solve for each left-hand variable in (4.2.22), given the data from previous iteration, e.g.,

$$\begin{aligned}
P_{11}^{(l)} &= C^{\intercal} Q_1 C - \left(R + \gamma B^{\intercal} P_{11}^{(l)} B\right) D_c^{(l-1)} D_c^{\intercal,(l-1)} \\
&+ \gamma \left(A^{\intercal} P_{11}^{(l)} A + A^{\intercal} P_{12}^{(l-1)} B_c^{(l-1)} \right. \\
&\qquad \left. + B_c^{\intercal,(l-1)} P_{21}^{(l-1)} A + B_c^{\intercal,(l-1)} P_{12}^{(l-1)} B_c^{(l-1)}\right), \\
&\vdots \\
P_{23}^{(l)} &= \gamma A_c^{\intercal,(l-1)} \left(P_{23}^{(l)} T_r - P_{21}^{(l-1)} B F^{(l-1)}\right)
\end{aligned}$$

2: **(Policy Improvement)** Update parameters A_c and B_c with this new data, such that following augmented matrix has the smallest possible eigenvalues and:

$$\bar{\lambda}\left(\begin{bmatrix} \bar{A}_{11}^{(l)} & \bar{A}_{12}^{(l)} \\ B_c^{(l)} & A_c^{(l)} \end{bmatrix}\right) < 1,$$

where

$$\begin{aligned}
\bar{A}_{11}^{(l)} &= A - \gamma B Z^{-1,(l)} \left(A^{\intercal} P_{11}^{(l)} B + B_c^{(l)} P_{21}^{(l)} B\right), \\
\bar{A}_{12}^{(l)} &= -\gamma B Z^{-1,(l)} B^{\intercal} P_{12}^{(l)} A_c^{(l)},
\end{aligned}$$

with

$$Z^{(l)} = \left(R + \gamma B^{\intercal} P_{11}^{(l)} B\right). \tag{4.2.33}$$

Then, the rest of the parameters are updated as

$$\begin{aligned}
F^{(l)} &= \gamma Z^{-1,(l)} B^{\intercal} P_{13}^{(l)} T_r, \\
C_c^{(l)} &= -\gamma Z^{-1,(l)} B^{\intercal} P_{12}^{(l)} A_c^{(l)}, \\
D_c^{(l)} &= -\gamma Z^{-1,(l)} \left(A^{\intercal} P_{11}^{(l)} B + B_c^{\intercal,(l)} P_{21}^{(l)} B\right).
\end{aligned}$$

4.2.3 Stochastic LQT with unknown parameters

Consider the LQT Q-function given by

$$Q\left(X[k], u[k]\right) = X^{\intercal}\bar{Q}X[k] + u^{\intercal}[k]Ru[k] + tr\left(P_w Q_1\right) + \gamma\left(X^{\intercal}[k+1]PX[k+1] + g[k+1]\right), \tag{4.2.34}$$

with $\bar{Q}$ defined in (4.2.15). Then, this expression is equivalent to

$$Q\left(X[k], u[k]\right) = \begin{bmatrix} X[k] \\ u[k] \end{bmatrix}^{\intercal} H \begin{bmatrix} X[k] \\ u[k] \end{bmatrix} + tr\left(P_w Q_1 + \gamma P_v P_{11}\right) + \gamma g[k+1], \tag{4.2.35}$$

with symmetric H given by

$$H = \begin{bmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{bmatrix}, \tag{4.2.36}$$

where

$$H_{uu} = R + \gamma B^\intercal P_{11} B, \tag{4.2.37a}$$

$$H_{xu} = \gamma \begin{bmatrix} B^\intercal P_{11} A + B^\intercal P_{12} B_c \\ B^\intercal P_{12} A_c \\ B^\intercal P_{13} T_r \end{bmatrix}, \tag{4.2.37b}$$

and H_{xx} given by:

$$H_{xx} = \begin{bmatrix} C^\intercal Q_1 C + \gamma \left(A^\intercal P_{11} A + B_c^\intercal P_{21} A + A^\intercal P_{12} B_c + B_c^\intercal P_{22} B_c \right) & \gamma \left(A^\intercal P_{12} + B_c^\intercal P_{22} \right) A_c & -C^\intercal Q_1 + \gamma \left(A^\intercal P_{13} + B_c^\intercal P_{23} \right) T_r \\ \gamma \left(A_c^\intercal P_{21} A + A_c^\intercal P_{22} B_c \right) & Q_2 + \gamma A_c^\intercal P_{22} A_c & \gamma A_c^\intercal P_{23} F \\ -Q_1 C + \gamma \left(B^\intercal P_{11} A + B^\intercal P_{12} B_c \right) & \gamma B^\intercal P_{12} A_c & Q_1 + \gamma B^\intercal P_{13} T_r \end{bmatrix}. \tag{4.2.38}$$

Based on (4.2.35), by finding the roots of its first derivative, it can be seen that the control law in terms of H is given by

$$u[k] = -H_{uu}^{-1} H_{ux} X[k]. \tag{4.2.39}$$

Note that the Q-function in (4.2.34) satisfies the Bellman equation

$$Q\left(X[k], u[k]\right) = X^\intercal[k] \bar{Q} X[k] + u^\intercal[k] R u[k] + tr\left(P_w Q_1\right) + \gamma Q\left(X[k+1], u[k+1]\right). \tag{4.2.40}$$

Then, let $S[k]$ be

$$S[k] = \begin{bmatrix} 1 \\ X[k] \\ u[k] \end{bmatrix}, \tag{4.2.41}$$

so (4.2.35) is equivalent to

$$Q\left(X[k], u[k]\right) = S^\intercal[k] \bar{H} S[k], \tag{4.2.42}$$

with $\bar{H}$ given by

$$\bar{H} = \begin{bmatrix} tr\left(P_w Q_1 + \gamma P_v P_{11}\right) + \gamma g[k+1] & 0 & 0 \\ 0 & H_{xx} & H_{xu} \\ 0 & H_{ux} & H_{uu} \end{bmatrix}, \tag{4.2.43}$$

so now (4.2.40) becomes

$$S^\intercal[k] \bar{H} S[k] = X^\intercal[k] \bar{Q} X[k] + u^\intercal[k] R u[k] + \gamma S^\intercal[k+1] \bar{H} S[k+1]. \tag{4.2.44}$$

Finally, (4.2.44) leads to Algorithm 6,which would match the same terms in the Algorithm shown in [22], if we change our stochastic setup for a deterministic one, and not consider the internal state of the controller.

Algorithm 6 Controller synthesis with unknown dynamics

1: **(Policy Evaluation)** Based on observations of $S[k]$, $S[k+1]$ and $\left(X^{\intercal}[k]\bar{Q}X[k] + u^{\intercal,(l)}[k]Ru^{(l)}[k]\right)$ for the l-th iteration, use Least Squares to find $\bar{H}$,

$$S^{\intercal}[k]\bar{H}^{(l+1)}S[k] = X^{\intercal}[k]\bar{Q}X[k] + u^{\intercal,(l)}[k]Ru^{(l)}[k] + \gamma S^{\intercal}[k+1]\bar{H}^{(l+1)}S[k+1]. \quad (4.2.45)$$

2: **(Policy Improvement)** Update the control law

$$u^{(l+1)}[k] = -\left(H_{uu}^{-1}\right)^{(l+1)} H_{ux}^{(l+1)} X[k]. \quad (4.2.46)$$

4.3 Simulation results

Consider the linear system given by

$$x[k+1] = \begin{bmatrix} 0.5 & 0 \\ 0.7 & 1.2 \end{bmatrix} x[k] + \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} u[k] + v[k], \quad (4.3.1a)$$

$$y[k] = \begin{bmatrix} 1 & 1 \end{bmatrix} x[k] + w[k], \quad (4.3.1b)$$

where $v[k]$ and $w[k]$ are the process and measurement noise respectively, with zero-mean and unitary variance.

Let the generator model T_r for the reference signal vary with time, set arbitrarily to

$$T_r[k] = \begin{cases} 0 & k < 60 \\ 10 & k \geq 60 \end{cases} \quad (4.3.2)$$

and $r[0] = 1$. Also, penalizing weights for the performance index were set to $Q_1 = 5$, $Q_2 = 5$ and $R = 1$, and the discount factor $\gamma = 0.8$.

Figure 4.1 shows the evolution of the generated control signal during the learning process, with u_1 obtained from Algorithm 5 and u_2 obtained from Algorithm 6, where the prior leads to parameters

$$F = 1.3667 \quad A_c = 0.4 \quad B_c = \begin{bmatrix} 1 & -1.52 \end{bmatrix} \quad (4.3.3a)$$

$$C_c = -0.5 \qquad D_c = \begin{bmatrix} 0.3 & 2.1 \end{bmatrix}. \quad (4.3.3b)$$

Remark 4.1. *In Algorithm 6, note that 25 data samples were collected to perform least squares in each iteration, since* $(n_x + n_{x_c} + n_u + n_y) \cdot (n_x + n_{x_c} + n_u + n_y + 1)/2$*, data tuples or more are needed, i.e., at least 15 samples for this particular simulation experiment.* □

If we consider that $S[k]$ and $S[k+1]$ are available as observations (and as a consequence, x, x_c, u are also available as observations at time k and $k+1$), the least squares problem can be formulated as the classical approach for linear regression:

$$\Delta\theta = b, \quad (4.3.4)$$

where Δ is a matrix with coefficients, θ is the parameters vector desired to find, and b is a vector with solutions of the linear system, so $b = X^\intercal[k]\bar{Q}X[k] + u^\intercal[k]Ru[k]$, θ contains a bias element and parameters $H_{xx}, H_{xu}, H_{ux}, H_{uu}$, and Δ is the matrix resulting from decomposition of $S^\intercal[k]\bar{H}S[k] - \gamma S^\intercal[k+1]\bar{H}S[k+1]$.

Then, H_{uu} and H_{ux} learned, which construct the control law in (4.2.39) are,

$$H_{uu} = 341.9354, \tag{4.3.5a}$$

$$H_{ux} = \begin{bmatrix} 101.134 & 704.594 & -173.186 & -450.291 \end{bmatrix}, \tag{4.3.5b}$$

leading to

$$u[k] = \begin{bmatrix} -0.2957 & -2.06 \end{bmatrix} x[k] + 0.5064x_c[k] + 1.3169\bar{r}[k], \tag{4.3.6}$$

which is similar to the final parameters found by Algorithm 5.

These parameters lead to closed loop eigenvalues given by $\lambda \in \{0.5, 0.59, 0.8\}$, leading to a stabilizing controller as depicted in Remark 3.3.

It can be seen in Figure 4.1, that Algorithm 5 and 6 achieve the same performance when the learning process is finished, but the latter has a slower convergence rate as expected, since adding knowledge represents an advantage, but it comes with a very high cost if the process is too complex for obtaining an accurate model.

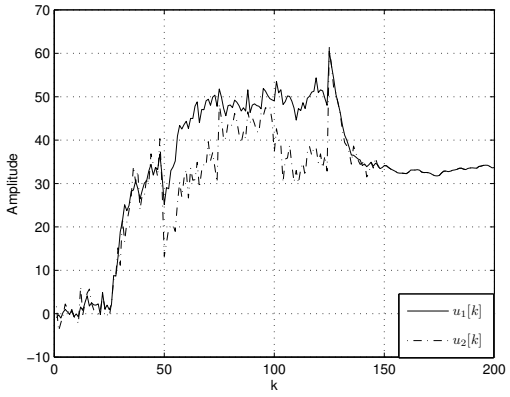


Figure 4.1: Control signal during learning process

4.4 Summary

Two PI algorithms were presented, one for the case when full knowledge of the model of the process to be controlled is assumed to be available, and the other one when there is no knowledge at all. The prior is specially useful for not having to tune by hand four parameters at a time, when the controller is allowed to have its own internal dynamics.

Chapter 5

EXPERIMENTAL RESULTS ON THE FURUTA PENDULUM

The rotary inverted pendulum is one of the most popular control experiments within the underactuated mechanical systems area, having a nonlinear model but allowing a linear approximation around the upper unstable position, broadening the extent of possible controller classes to the well-known and extensively studied linear realm. This chapter presents a switched strategy which involves a energy-based nonlinear controller and a linear quadratic regulator formulation for the swing-up and stabilization task respectively. Experimental results will be shown on a physical prototype designed and available for use at the institution (Electronics Dept. at Universidad Técnica Federico Santa María). Then, an adaptive approach which makes use of a simplified procedure from previous chapter will be shown and implemented over simulation, evaluating performance of resulting controller on the laboratory prototype.

5.1 Introduction

Underactuated plants are appealing due to their interesting spectrum applications such as aerospace systems or robotic plattforms [10, 28, 37], and the efforts of the controller may be diminished if it takes advantages of the structural dynamics often involved, instead of the classical feedback scheme where the controller aims to nullify the system to be controlled.

Control community have done a great variety of work related to underactuated systems and introduced some benchmark problems like the cart-pole [48], where the control problem requires not only to stabilize the pendulum in the upper unstable position, also displacement of the cart has to be considered, increasing design complexity. There are other similar underactuated mechanical systems with model of reduced order, allowing to capture the essence of a problem without introducing all the complexity frequently involved on real world applications. An example of a trivially underactuated system, with two degrees of freedom and one actuator, is the rotary inverted pendulum (RIP) also known as Furuta pendulum [4].

Different control techniques have been implemented on similar problems, such as [16] which applies different evolutionary algorithms in order to adjust parameters of a PID controller, including genetic algorithms, particle swarm optimization and ant colony opti-

mization which simulates behavior of ants and their ability to find the shortest path from their nest to a food source [14]. Although these methods may provide sufficient performance for the obtained controller, there are no guarantees of stability nor convergence speed in the intermediate process.

Other model-based control approaches on the Furuta pendulum have been also reported in literature, like [6] where the design and implementation of an adaptive sliding mode controller is discussed, along with other sliding mode variations in order to compare their performance within that class of controllers. Although nonlinear control does not deal with model approximations, linear control is a powerful approach which can provide a more in-depth stability analysis due to the extensively studied linear theory.

On the linear class of controllers, work on [31] a linear controller based on a linear observer is obtained under an active disturbance rejection control scheme. A similar approach to the switched control strategy to be used on the RIP can be found in [29], where an energy-based nonlinear controller is designed for the swing-up task of the flywheel inverted pendulum, and also a locally stabilizing controller is obtained by means of tuning the appropiate parameters of a PID controller in order to take the pendulum from its stable rest position to the upward unstable vertical line.

The remainder of this document is organized as follows. Section 5.2 presents the RIP nonlinear model along with its linear approximation around the unstable equilibrium point, and its validation using measured data from the physical prototype. Section 5.3 shows the energy-based nonlinear controller for the swing-up task, obtained from the Lyapunov global stability theory. Then, Section 5.4 formulates the LQR problem which leads to a state feedback control law in order to stabilize the pendulum in the upward position, while Section 5.5 presents the switched approach of both presented approaches. Finally, Section 5.6 presents a simplified procedure from the previous chapter for implementation on a simulated version of this task, transferring obtained parameters to physical prototype, and Section 5.7 presents simulation results with physical prototype parameters in order to implement the adaptive approach.

5.2 Rotary inverted pendulum

Consider a second order and controllable dynamical system given by:

$$\ddot{q} = f(q, \dot{q}, u, t), \tag{5.2.1}$$

where u corresponds to the control vector, q and $\dot{q}$ are the positions and velocities vector respectively, and t denotes the possible influence of time over the acceleration vector $\ddot{q}$. For the case that concerns this study, where dynamics are affine on the commanded torque, this expression can be rewritten as:

$$\ddot{q} = f_1(q, \dot{q}, t) + f_2(q, \dot{q}, t)u. \tag{5.2.2}$$

Then, in formal terms a control system described by (5.2.1) is called underactuated in the configuration $(q, \dot{q}, t)$ if it is not possible to drive an instantaneous acceleration on any arbitrary direction, i.e.:

$$rank(f_2(q, \dot{q}, t)) < dim(q). \tag{5.2.3}$$

The rotary inverted pendulum (RIP), also known as Furuta pendulum is an example of an underactuated system, and as illustrated in Figure 5.1, it consists of a controlled arm in the horizontal plane rotating around a central axis, and a pendulum linked to one of the ends of this arm, rotating over the vertical plane. This plant allows us to study nonlinear dynamics of simplified models which are also possible to construct physically, building mechatronic platforms with enhance reproducibility features.

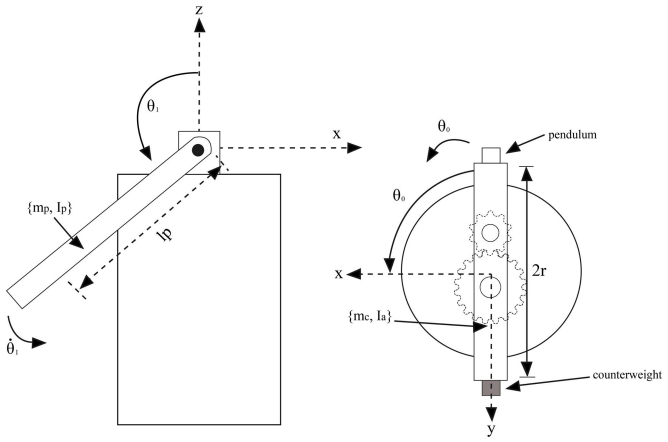


Figure 5.1: Side and top view of Furuta pendulum

According to Figure 5.1, there is a DC motor responsible for controlling the arm position measured by θ_0 through the armature voltage, and its main parameters are the armature resistance and inductance, R_a and L_a respectively.

5.2.1 Model formulation

Define $\{l_p, m_p, I_p, \theta_1\}$, where l_p stands for the length from the rotating axis of the pendulum to its center of mass, m_p denotes the mass of the pendulum with its moment of inertia I_p and its rotation angle θ_1. Then, r corresponds to the arm radius which has a moment of inertia I_a, introducing a counterweight of mass m_c which takes the center of the mass of the rotating arm at height h. Consider the rotation angle of the arm to be given by θ_0. A detailed description of parameters involved in the model along with its values for the physical prototype is shown in Table 5.1.

The Lagrangian is then given by (5.2.4), where E_k and E_p denote the kinetic and potential energy respectively, while q is the generalized coordinates vector, $q = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix}^\mathsf{T}$.

$$L(q, \dot{q}) = E_k(q, \dot{q}) - E_p(q, \dot{q}) \tag{5.2.4}$$

Kinetic energy for the pendulum is given by the sum of traslational and rotational components, while kinetic energy for the arm is given by rotation and tangential kinetic energy:

$$E_k = \frac{1}{2}\hat{J}_0\dot{\theta}_0^2 + \frac{1}{2}\hat{J}_1\dot{\theta}_1^2 + \frac{1}{2}m_p l_p^2\dot{\theta}_0^2 sin^2(\theta_1) - m_p r l_p \dot{\theta}_0\dot{\theta}_1 cos(\theta_1), \tag{5.2.5}$$

where $\hat{J}_0$ and $\hat{J}_1$ are given by

$$\hat{J}_0 = I_a + r^2(m_p + m_c), \tag{5.2.6a}$$

$$\hat{J}_1 = I_p + m_p l_p^2, \tag{5.2.6b}$$

while potential energy is given in terms of the pendulum and counterweight masses:

$$E_p = m_p g l_p cos(\theta_1) + m_c g h. \tag{5.2.7}$$

Symbol	Description	Value
m_p	Pendulum mass	$0.1\ kg$
m_c	Counterweight mass	$0.01\ kg$
I_p	Pendulum inertial moment	$5.1 \cdot 10^{-4}\ kg\ m^2$
I_a	Arm inertial moment	$3.1 \cdot 10^{-3}\ kg\ m^2$
r	Arm radius	$0.13\ m$
l_p	Pendulum mass center	$0.125\ m$
h	Arm center of mass height	$0.055\ m$
C_0	Arm friction coefficient	$10^{-4}\ N\ m\ s$
C_1	Pendulum friction coefficient	$10^{-4}\ N\ m\ s$
R_a	Armature resistor	$8\ \Omega$
L_a	Motor inductance	$10\ mH$
I_m	Motor inertia	$1.9 \cdot 10^{-6}\ kg\ m^2$
M_f	Motor mutual inductance	$0.0214\ N\ m/A$
K_g	Gear reduction coefficient	59927
K_{eg}	External gear reduction coefficient	16
g	Gravitational acceleration	$9.806\ m/s^2$

Table 5.1: RIP parameters

Then, according to the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \tau_i, \quad i = \theta_0, \theta_1 \tag{5.2.8}$$

with τ_i being the moments applied to each coordinate, leading to

$$M_1(\theta_1)\ddot{q} + M_2(\theta_1)\dot{q} + M_3(\theta_1) = T, \tag{5.2.9}$$

where M_1, M_2 and M_3 corresponds to inertia matrix, centripetal and Coriolis torque matrices respectively and T is the torque vector:

$$M_1(\theta_1) = \begin{bmatrix} \hat{J}_0 + m_p l_p^2 sin^2(\theta_1) & -m_p r l_p cos(\theta_1) \\ -m_p r l_p cos(\theta_1) & \hat{J}_1 \end{bmatrix}, \tag{5.2.10a}$$

$$M_2(\theta_1) = \begin{bmatrix} m_p l_p^2 \dot{\theta}_1 sin(2\theta_1) + C_0 & m_p r l_p \dot{\theta}_1 sin(\theta_1) \\ -\frac{1}{2} m_p l_p^2 \dot{\theta}_0 sin(2\theta_1) & C_1 \end{bmatrix}, \tag{5.2.10b}$$

$$M_3(\theta_1) = \begin{bmatrix} 0 \\ -m_p g l_p sin(\theta_1) \end{bmatrix}, \tag{5.2.10c}$$

$$T = \begin{bmatrix} \tau_l \\ 0 \end{bmatrix}, \tag{5.2.10d}$$

with C_0 and C_1 as the friction coefficients of the arm and pendulum respectively, with values given in Table 5.1. From (5.2.10), the reader can note that the motor actuates just on the arm and matrices model the movement transfer to the pendulum. This load torque, τ_l is related to the motor dynamics which is related with electrical torque, τ_e, and electrical angular velocity, ω_e, given by

$$V(t) = R_a i(t) + L_a \frac{d}{dt} i(t) + M_f \omega_e, \tag{5.2.11a}$$

$$\tau_e = M_f K_g i(t), \tag{5.2.11b}$$

$$\tau_e - \tau_l = I_m \frac{d}{dt} \omega_e, \tag{5.2.11c}$$

with parameters as described in Table 5.1, and $V(t)$,$i(t)$ the applied voltage and current to the motor at time t, respectively.

Finally, applied torque and angular velocity is related with its corresponding electrical variables as follows:

$$\omega = K_g^{-1} \omega_e, \tag{5.2.12a}$$

$$\tau = K_g \tau_e. \tag{5.2.12b}$$

Then, the RIP model including motor dynamics is given by:

$$\ddot{q} = \bar{M}_1(\theta_1)^{-1} \left(\bar{T} - \bar{M}_2(\theta_1) \dot{q} - M_3(\theta_1) \right), \tag{5.2.13}$$

where new matrices are described by:

$$\bar{M}_1(\theta_1) = \begin{bmatrix} \hat{J}_0 + m_p l_p^2 sin^2(\theta_1) + K_g^2 I_m & -m_p r l_p cos(\theta_1) \\ -m_p r l_p cos(\theta_1) & \hat{J}_1 \end{bmatrix}, \tag{5.2.14a}$$

$$\bar{M}_2(\theta_1) = \begin{bmatrix} m_p l_p \dot{\theta}_1 sin(2\theta_1) + C_0 & m_p r l_p \dot{\theta}_1 sin(\theta_1) \\ -\frac{1}{2} m_p l_p \dot{\theta}_0 sin(2\theta_1) & C_1 \end{bmatrix}, \tag{5.2.14b}$$

$$\bar{T} = \begin{bmatrix} M_f K_g i \\ 0 \end{bmatrix}. \tag{5.2.14c}$$

Define the state vector $x = \begin{bmatrix} \theta_0 & \dot{\theta}_0 & \theta_1 & \dot{\theta}_1 & i \end{bmatrix}^\intercal$. Then, in order to obtain a linearized model around the equilibrium point $x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}$, recall (5.2.11a) and solve for the derivative of current i:

$$\frac{d}{dt} i(t) = \frac{1}{L_a} V(t) - \frac{R_a}{L_a} i(t) - \frac{M_f}{L_a} \omega_e. \tag{5.2.15}$$

From (5.2.12a) and noting that this prototype has an external gear reduction system which relates the arm angular velocity with its corresponding variable obtained from the motor described by

$$\dot{\theta}_0 = K_{eg}^{-1}\omega, \tag{5.2.16}$$

expression (5.2.15) can be rewritten as

$$\frac{d}{dt}i(t) = \frac{1}{L_a}V(t) - \frac{R_a}{L_a}i(t) - \frac{M_f}{L_a K_g}\dot{\theta}_0. \tag{5.2.17}$$

Then, the following linear model is obtained

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{5.2.18}$$

where x is the previously defined state vector, u corresponds to the control signal (applied voltage) and matrices A and B given by

$$A = \alpha \cdot \begin{bmatrix} 0 & \frac{1}{\alpha} & 0 & 0 & 0 \\ 0 & -\hat{J}_1 C_0 & m_p^2 l_p^2 g r & m_p r l_p C_1 & M_f K_g \hat{J}_0 \hat{J}_1 \\ 0 & 0 & 0 & \frac{1}{\alpha} & 0 \\ 0 & m_p r l_p C_0 & \hat{J}_0 m_p g l_p & -\hat{J}_0 C_1 & K_g M_f m_p r l_p \\ 0 & -\frac{M_f}{L_a K_g \alpha} & 0 & 0 & -\frac{R_a}{L_a \alpha} \end{bmatrix}, \tag{5.2.19a}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{L_a} \end{bmatrix}^\intercal, \tag{5.2.19b}$$

with α given by the following expression:

$$\alpha = \frac{1}{\hat{J}_0 \hat{J}_1 - (m_p r l_p)^2}. \tag{5.2.20}$$

5.2.2 Experimental setup

- The signal acquisition was made with a National Instruments board specially designed for analog and discrete time I/O acquiring tasks. In particular the model used corresponds to CB-68LPR whose connections diagram is shown in the Appendix.
- The experimental results were obtained in an Intel Core Duo 2.94GHz with 4GB RAM running LabView 2014, first setting properly the input and output tasks on National Instruments - Measurement & Automation Explorer, as depicted also in the Appendix.
- As previously described, two encoders were used on this prototype (one for measuring pendulum position, and the other one used for measuring arm position). These encoders corresponds to the model AMT102-V.
- Sampling period was set to 2 miliseconds.

5.2.3 Model validation

Considering a region of 15 degrees around the down stable rest position, $\theta_1 = \pi$, the dynamic equation for the pendulum can be approximated as linear and can be written as:

$$\hat{J}_1\ddot{\theta}_1 + C_1\dot{\theta}_1 + m_p l_p g\theta_1 = 0, \tag{5.2.21}$$

or in its general form:

$$\ddot{\theta}_1 + 2\zeta\omega_n\dot{\theta}_1 + \omega_n^2\theta_1 = 0, \tag{5.2.22}$$

with ω_n as the natural oscillation frequency and ζ as the damping coefficient. From (5.2.21) and (5.2.22) we have:

$$\omega_n = 7.69, \tag{5.2.23a}$$

$$\zeta = 3.13 \cdot 10^{-3}, \tag{5.2.23b}$$

while experimentally we can estimate these parameters from a free response of the system with:

$$\hat{\zeta} = \frac{1}{\sqrt{1 + \left(\frac{\pi}{ln^2(M/N)}\right)}}, \tag{5.2.24a}$$

$$\hat{\omega}_n = \frac{2\pi}{T_s\sqrt{1-\hat{\zeta}^2}}, \tag{5.2.24b}$$

where M corresponds to the ratio from the first two consecutive peaks, N is the stationary value and T_s stands for the semiperiod of the maximum oscillation. Then, from Figure 5.2 we obtain :

$$\hat{\omega}_n = 9.2, \tag{5.2.25a}$$

$$\hat{\zeta} = 0.00585. \tag{5.2.25b}$$

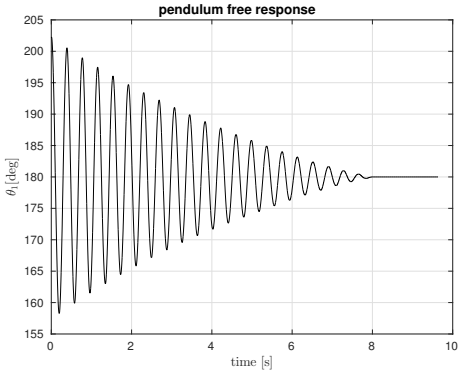


Figure 5.2: Pendulum damped oscillation

5.3 Swing-up controller

Model from (5.2.13) can be simplified if we do not consider centripetal and Coriolis torque, that is inserting zero values replacing current entries of matrices $\bar{M}_2(\theta_1)$ and $M_3(\theta_1)$, obtaining:

$$\hat{J}_1\ddot{\theta}_1 - m_p g l_p sin(\theta_1) = m_p r\ddot{\theta}_0 cos(\theta_1). \tag{5.3.1}$$

Then, by noting that angular acceleration θ_0 is directly controlled from the actuator, the total unforced energy for the pendulum would be given by:

$$E_u = \frac{1}{2}\hat{J}_1\dot{\theta}_1^2 + m_p g l_p cos(\theta_1) + m_c g h. \tag{5.3.2}$$

Given that the objective of the switched control strategy is to stabilize the pendulum at the upward position, that is when $\theta_1 = \dot{\theta}_1 = 0$, so desired energy at the target position is given by:

$$E_d = m_p g l_p + m_c g h, \tag{5.3.3}$$

implying that unforced normalized energy with respect to E_d is as follows:

$$\begin{aligned} E_{un} &= E_u - E_d, \\ &= \frac{1}{2}\hat{J}_1\dot{\theta}_1^2 + m_p g l_p \left(cos(\theta_1) - 1\right), \end{aligned} \tag{5.3.4}$$

leading to $E_{un} = 0$ at the equilibrium and $E_{un} = -2m_p g l_p$ at the rest (down) position. Figure 5.3 shows the evolution of unforced normalized energy over time when the pendulum is taken from the upward to the rest position with no control signal being applied on the system.

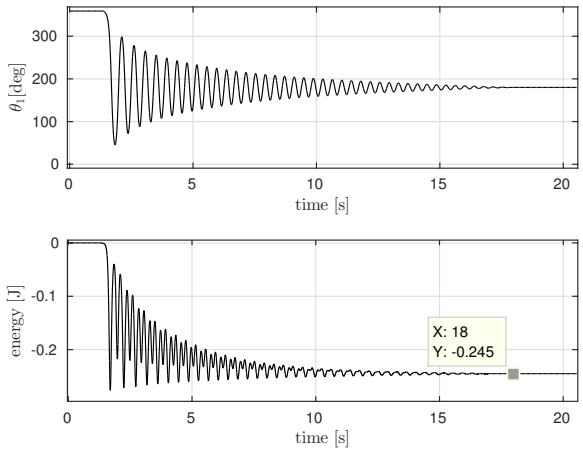


Figure 5.3: Unforced normalized energy evolution

In order to obtain a nonlinear controller that minimizes (5.3.4), consider the following positive definite Lyapunov function:

$$V = \frac{1}{2}E_{un}^2. \tag{5.3.5}$$

Then, a controller minimizing (5.3.4), which is taking $E_{un} \to 0$ asymptotically as $t \to \infty$ from any given initial condition, has to be chosen such that $\dot{V} < 0$. This derivative is given by:

$$\dot{V} = (E_u - E_d)\,\dot{E}_u, \tag{5.3.6}$$

where using (5.3.1) and (5.3.2) the derivative of unforced energy is obtained:

$$\dot{E}_u = m_p r \ddot{\theta}_0 cos(\theta_1)\dot{\theta}_1. \tag{5.3.7}$$

Therefore, proposed control law includes a positive swing-up gain,λ_{su} whose value is chosen such that (5.3.6) is never positive semi definite:

$$\ddot{\theta}_0 = -\lambda_{su}\,(E_u - E_d)\,cos(\theta_1)\dot{\theta}_1. \tag{5.3.8}$$

Given that controlling the angular velocity of the motor implies controlling its current, transfer function from current to voltage is simplified by means of neglecting armature inductance, so the relation between angular acceleration and applied voltage becomes:

$$V(t) = R_a i(t) + M_f \omega_e. \tag{5.3.9}$$

On the other hand, considering (5.2.11b) and $\tau_l \approx \hat{J}_0 \ddot{\theta}_0$ in (5.2.11c):

$$K_g M_f i(t) + \hat{J}_0 \lambda_{su}(E_u - E_d)cos(\theta_1)\dot{\theta}_1 = I_m \frac{d}{dt}\omega_e, \tag{5.3.10}$$

so by solving (5.3.9) in terms of the current and replacing this, and (5.2.12a) with (5.2.16) into (5.3.10), the non-linear control law expressed in terms of the applied voltage would be given by

$$V(t) = M_f K_g K_{eg} \dot{\theta}_0 - \bar{\lambda}_{su}(E_u - E_d)cos(\theta_1)\dot{\theta}_1, \tag{5.3.11}$$

with

$$\bar{\lambda}_{su} = \frac{R_a(\hat{J}_0 + I_m K_g K_{eg})}{K_g M_f}\lambda_{su}. \tag{5.3.12}$$

Figure 5.4 shows the pendulum position and the corresponding control signal being applied to the motor while performing the swing-up movement.

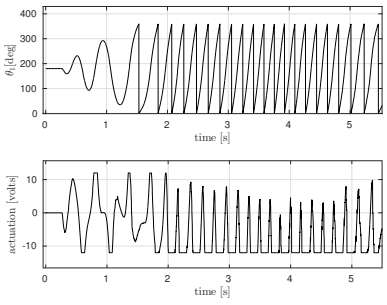


Figure 5.4: Pendulum position and actuation on the swing-up task

5.4 Linear quadratic regulator

Once the pendulum position has been taken from the down rest equilibrium to the upward unstable position, the next step is to keep the pendulum at zero degrees with the minimum possible deviation, so a stabilizing controller has to be obtained.

Considering a linear and time-invariant system described in state space variables just like (5.2.18), the cost function is given by:

$$J = \int_0^\infty \left(x^\intercal(t)Qx(t) + u^\intercal(t)Ru(t)\right) dt, \tag{5.4.1}$$

where Q and R are weighting matrices positive definite and positive semi-definite respectively. Then, state-feedback control law which minimizes (5.4.1) is given by:

$$u(t) = -Lx(t), \tag{5.4.2}$$

where L is such that

$$L = R^{-1}B^\intercal P, \tag{5.4.3}$$

and P stands for the solution of the algebraic Ricatti equation (ARE):

$$A^\intercal P + PA - PBR^{-1}B^\intercal P + Q = 0. \tag{5.4.4}$$

Matrices Q and R have to be designed by means of giving more weight to states that are more relevant than others for the particular experiment. In this case, priority is given to pendulum position, θ_1, while other important variables correspond to arm and pendulum velocities, so experiment proposal is given by:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.4.5a}$$

$$R = 10, \tag{5.4.5b}$$

where penalizing actuation factor, R, is chosen such that DC motor does not perform any sudden behavior changes desestabilizing pendulum position.

Then, solving (5.4.4) with parameters described in (5.2.19) and replacing into (5.4.3) yields:

$$L = \begin{bmatrix} -0.31 & -5.26 & 70.74 & 8.92 & 0.17 \end{bmatrix}. \tag{5.4.6}$$

Figure 5.5 shows the pendulum position control once it has reached the stabilizible region starting from a swing-up behavior.

Given that this full state-feedback implementation require velocity sensors for acquiring the appropiate measurement, a linear observer, O, as a soft-sensor is used, whose (continuous) transfer function is given by

$$O(s) = \frac{50s}{s + 50}. \tag{5.4.7}$$

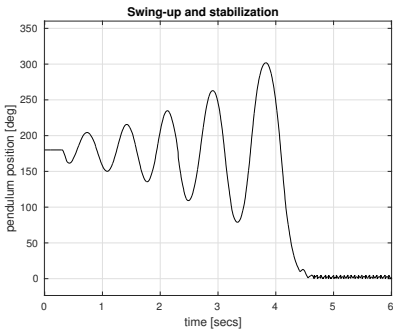


Figure 5.5: Pendulum position stabilization

5.5 Switched control strategy

The switched control strategy consists in combining the energy-based swing-up task described in this document, and the state-feedback controller obtained by means of solving the linear quadratic regulator problem discussed in the previous section.

Switching criteria is based on the stabilizable region of the pendulum, since a linear approximation of the model can be assumed when the pendulum position is less than 15 degrees, but also velocity (estimated by software, since there are no direct measurements of velocities) has to be considered, so LQR controller will perform if:

- pendulum position is less than 15 degrees,
- pendulum velocity is less than 30 rad/sec,

otherwise swing-up controller will be used.

In order to be able to control both degrees of freedom of this laboratory underactuated plant, more importance to the arm position (actuated link) has to be given, so Q is modified accordingly:

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.5.1}$$

while value of R stays the same, $R = 10$, leading to solution of the ARE:

$$P = \begin{bmatrix} .0601 & .0153 & -.1718 & -.0216 & -.0002 \\ \cdot & .0141 & -.175 & -.0219 & -.0003 \\ \cdot & \cdot & 2.2155 & .2746 & .004 \\ \cdot & \cdot & \cdot & .0344 & .0005 \\ \cdot & \cdot & \cdot & \cdot & 0 \end{bmatrix} \cdot 10^6, \tag{5.5.2}$$

where $\cdot$ is used to denote symmetry. This matrix allows us to obtain the state-feedback

controller described by

$$L = \begin{bmatrix} -3.1 & -6.07 & 80.02 & 10.08 & 0.19 \end{bmatrix}. \tag{5.5.3}$$

Then, Figure 5.6 shows the signals of interest in the switched control strategy, combining both the energy-based and lqr controller for swing-up and stabilization task respectively. The first plot shows the pendulum position, θ_1, while the following plot shows tracking of reference r for the actuated link, θ_0. Figure 5.6 also shows the control signal, u, which corresponds to the applied voltage over DC motor.

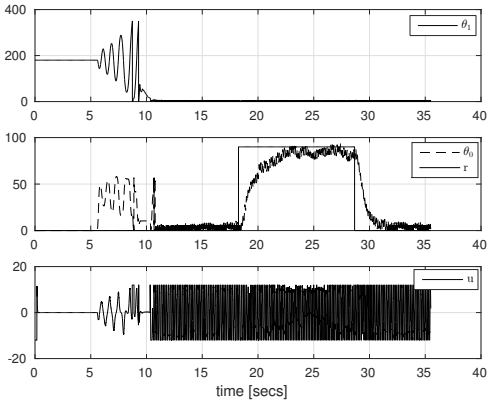


Figure 5.6: Pendulum and arm position control

Remark 5.1. *Oscillations in the actuaction signal are caused by practical issues such as the zero position, so if the vertical stable pendulum position is not exactly located at 0 degrees and actuation is higher than desired, the pendulum will always be struggling against gravity. Nevertheless, as will be shown in the following Section, when dynamic state-feedback controller is used, this problem is solved since its actuation magnitud is lower.*

5.6 Dynamic state-feedback controller

According to Chapter 3, Figure 5.7 shows the dynamic state-feedback controller performance over the Furuta pendulum. Recall that this controller is given by

$$x_c[k+1] = A_c x_c[k] + B_c x[k], \tag{5.6.1a}$$
$$u[k] = r[k] - \left(C_c x_c[k] + D_c x[k]\right), \tag{5.6.1b}$$

where $r[k]$ is a pre-filtered reference obtained from $\bar{r}[k]$, so matrices were set to:

$$\begin{aligned} A_c &= 0.4, \\ B_c &= \begin{bmatrix} 1 & 2 & 1 & 2 & 0 \end{bmatrix} \\ , C_c &= -0.5, \end{aligned}$$

$$D_c = \begin{bmatrix} -3 & -6 & 80 & 10 & 0 \end{bmatrix},$$

so pre-filter F is given by $F = 3.83$.

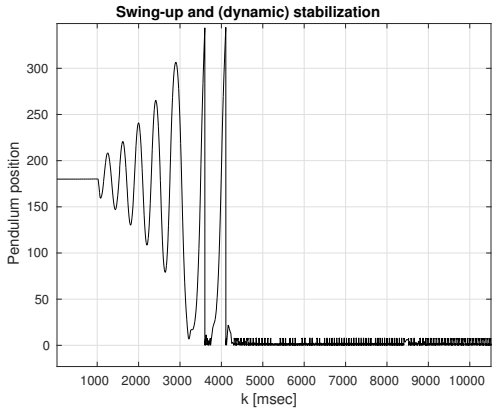


Figure 5.7: Pendulum stabilization under dynamic state-feedback

Then, applying the same switching criteria as in the previous Section, switched control strategy using this dynamic state-feedback controller is shown in Figure 5.8, with closed-loop eigenvalues in $\lambda \in \{0.31, 0.31, 0.98, 0.99, 0.99, 0.99\}$, noting that closed-loop eigenvalues of the static state-feedback architecture are $\lambda \in \{0.21, 0.93, 0.98, 0.98, 0.99\}$, so despite introducing additional dynamics the slowest mode is on the same magnitude in both architectures, but control signal has lower amplitude in the dynamic approach.

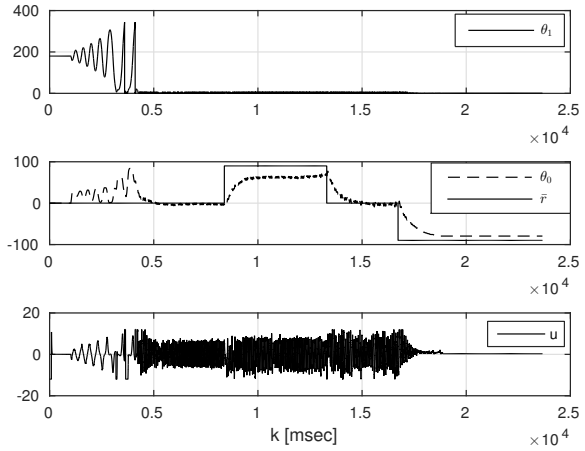


Figure 5.8: Pendulum and arm position control under dynamic state-feedback

5.7 Adaptive approach

In order to compare directly performance of the adaptive and the classical approach on a given task (pendulum position stabilization in this case), state-feedback controller will be assumed to have no internal dynamics, i.e., the analytically and the learned controllers will be described by

$$u[k] = -Lx[k], \tag{5.7.1}$$

at time k, so the difference will be in the approach for obtaining this controller, L.

Recall (4.2.40), so in this case Q-function is given by

$$Q(x_k, u_k) = x_k^\intercal Q x_k + u_k^\intercal R u_k + (Ax_k + Bu_k)^\intercal S (Ax_k + Bu_k), \tag{5.7.2}$$

with $\gamma = 1$, so the action-value function becomes undiscounted, and S being the algebraic riccati equation solution, so (5.7.2) can be written as

$$Q(x_k, u_k) = \begin{bmatrix} x_k \\ u_k \end{bmatrix}^\intercal \begin{bmatrix} A^\intercal SA + Q & B^\intercal SA \\ A^\intercal SB & B^\intercal SB + R \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix}, \tag{5.7.3}$$

so control law is given by

$$u[k] = S_{uu}^{-1} S_{ux} x[k], \tag{5.7.4}$$

with $S_{uu} = (B^\intercal SB + R)$, and $S_{ux} = B^\intercal SA$, so knowledge of the dynamics system would be required in order to perform policy improvement on policy iteration or value iteration algorithms. On the other hand, Q-function (and therefore kernel matrix S) can be determined online in real time without knowing system dynamics (A, B) by just using data measured along the system trajectories.

Figure 5.9 shows performance of the pendulum position control over this adaptive approach, where the learned controller, L_2 is given by

$$L_2 = \begin{bmatrix} 0 & -.17 & 2.24 & .28 & .005 \end{bmatrix} \cdot 10^3, \tag{5.7.5}$$

and the state-feedback controller obtained by means of solving the algebraic riccati equation is described by

$$L_1 = \begin{bmatrix} -.31 & -5.26 & 70.74 & 8.92 & .17 \end{bmatrix}, \tag{5.7.6}$$

with closed-loop poles given by

$$eig\,(A - B * L_1) = \{0.2140, 0.9394, 0.9852, 0.9852, 0.9998\}, \tag{5.7.7a}$$

$$eig\,(A - B * L_2) = \{0.0069, 0.2147, 0.9446, 0.9845, 0.9999\}, \tag{5.7.7b}$$

noting that the adaptive controller has closed-loop modes that disappear faster than the controller obtained then under the classical approach, and since the task is limited to stabilize the pendulum in the upward position, as the intuition suggests the arm position does not play any decisive role.

Solution of solving analytically the appropiate algebraic riccati equation:

$$P = \begin{bmatrix} .0046 & .0013 & -.0152 & -.0019 & 0 \\ \cdot & .0103 & -.1321 & -.0165 & -.0003 \\ \cdot & \cdot & 1.7318 & .2139 & .0036 \\ \cdot & \cdot & \cdot & .0268 & .0005 \\ \cdot & \cdot & \cdot & \cdot & 0 \end{bmatrix} \cdot 10^6, \tag{5.7.8}$$

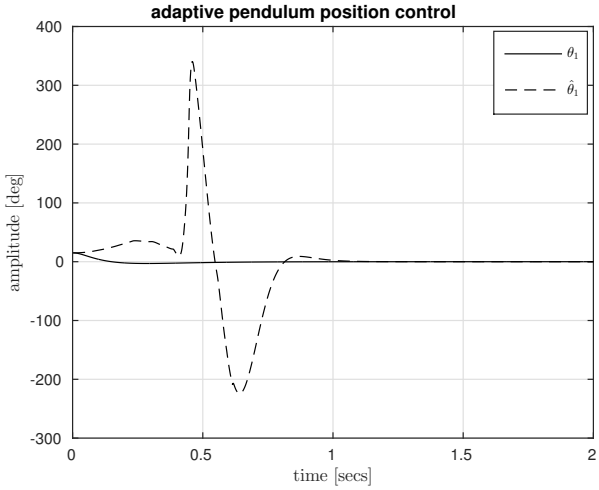


Figure 5.9: Pendulum position control

while the corresponding learned matrix from applying Algorithm 5 is given by

$$P = \begin{bmatrix} 0 & .0001 & -.0014 & -.0002 & 0 \\ 0 & -.0999 & 1.2757 & .1634 & .0028 \\ -.0001 & .3407 & -4.3501 & -.5572 & -.0097 \\ 0 & .1281 & -1.6356 & -.2095 & -.0037 \\ 0 & 0 & -.0003 & 0 & 0 \end{bmatrix} \cdot 10^{10}. \tag{5.7.9}$$

5.8 Summary

A classic well-known underactuated system, the rotary inverted pendulum (also known as Furuta pendulum) has been studied and a linear model has been obtained. The switched strategy presented consists of a energy-based swing up task, and stabilization in the upward position due to a linear controller obtained by means of posing the stabilization problem as a linear quadratic regulator design, which also has proved to be of interest when controlling both degrees of freedom of the underactuated plant. Also, pendulum position control was implemented in an adaptive approach under a simulated environment, while implementation on the physical prototype remains as future work.

Chapter 6

CONCLUSIONS

In this thesis, the Linear Quadratic Tracking (LQT) problem was tackled, namely, the control problem where the output has to keep track of the trajectory of some exogenous reference signal assuming a linear model and a quadratic cost function. To this end, we used a (linear) dynamic state feedback controller, whose parameters were chosen by applying reinforcement learning techniques, proving to be specially useful when the model of the plant to be controlled is unknown or inaccurate, but still useful when dynamics are given, since hand-tuning of controller parameters could represent a time consuming task due to its number of degrees of freedom and its constraints. Simulation results showed that the proposed controller scheme performs the same as the classical one in terms of the tracking error, but this is achieved at a lower effort in terms of the control signal, at the cost of tuning more parameters. These simulation results were implemented on an arbitrary plant for both cases: state-feedback over the true state, and a feedback loop based on state estimations from a Kalman filter.

Later, two policy iteration algorithms were presented, one for the case when full knowledge of the model of the process to be controlled is assumed to be available, and the other one when there is no knowledge at all. Control problem was formulated as a Markovian Decision Process (MDP) and then used in an adaptive control approach using reinforcement learning, which is often interesting due to its practical applications, since it does not require complete knowledge of system dynamics. The case when knowledge of the model is assumed to be available is still useful for not having to tune by hand many parameters at a time, when the controller is allowed to have its own internal dynamics.

Then, a well-known underactuated system, the rotary inverted pendulum (also known as Furuta pendulum) was studied and a linear model was obtained. The switched strategy presented consists of a energy-based swing up task, and stabilization in the upward position due to a linear controller obtained by means of posing the stabilization problem as a linear quadratic regulator (LQR) design, which also has proved to be of interest when controlling both degrees of freedom of the underactuated plant. Also, pendulum position control was implemented in an adaptive approach under a simulated environment, while implementation on the physical prototype remains as future work.

Also, future work remains on the extension of these techniques to other underactuated systems, such as robotic manipulators or legged platforms since their dynamics are similar to a system of pendulum with several coupled links, representing also a platform rich enough in terms of dynamics complexity due to non-linearities.

Appendix A

APPENDIX

A.1 Useful matrix properties

This section will review some of the more important properties of matrices, which have been proved to be useful for derivations along this thesis.

Lemma A.1 (**Determinant of a block matrix**).
Consider a block matrix A given by

$$A = \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right], \tag{A.1.1}$$

where A_{ij} denotes the ij-th block. Then, $det(A)$ can be expressed as by the use of

$$C_1 = A_{11} - A_{12}A_{22}^{-1}A_{21}, \tag{A.1.2}$$
$$C_2 = A_{22} - A_{21}A_{11}^{-1}A_{12}, \tag{A.1.3}$$

as

$$det\left(\left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right]\right) = det(A_{22}) \cdot det(C_1), \tag{A.1.4}$$
$$= det(A_{11}) \cdot det(C_2), \tag{A.1.5}$$

where either A_{11} or A_{22} has to be invertible.

■

Proof:

It is very straightforward that for every triangular block matrix, its determinant is given by the multiplication of the determinant of the blocks on the diagonal. Although A is not triangular, one could apply LU factorization, which turns A into

$$A = \left[\begin{array}{c|c} A_{11} & 0 \\ \hline A_{21} & I \end{array}\right] \cdot \left[\begin{array}{c|c} I & A_{11}^{-1}A_{12} \\ \hline 0 & A_{22} - A_{21}A_{11}^{-1}A_{12} \end{array}\right] \tag{A.1.6}$$
$$= \left[\begin{array}{c|c} I & A_{12} \\ \hline 0 & A_{22} \end{array}\right] \cdot \left[\begin{array}{c|c} A_{11} - A_{12}A_{22}^{-1}A_{21} & 0 \\ \hline A_{22}^{-1}A_{21} & I \end{array}\right] \tag{A.1.7}$$

Then, (A.1.6) and (A.1.7) yields (A.1.4) and (A.1.5) when A_{11} or A_{22} is invertible respectively.

□□

Lemma A.2 (Eigenvalues of two matrices product).
Consider a matrix A and B, such that their n eigenvalues are given by

$$eig(A) : \lambda_{A,1} \geq \lambda_{A,2} \geq ... \geq \lambda_{A,n}, \tag{A.1.8a}$$
$$eig(B) : \lambda_{B,1} \geq \lambda_{B,2} \geq ... \geq \lambda_{B,n}. \tag{A.1.8b}$$

Then, an upper bound for the k-th eigenvalue of the matrix product AB, i.e., $\lambda_{AB,k}$, is given by

$$\lambda_{AB,k} \leq \min_{i+j=k+1} \lambda_{A,i} \cdot \lambda_{B,j} \tag{A.1.9}$$

■

Proof: Let T be an operator of rank

$$rank(T) \leq i - 1, \tag{A.1.10}$$

such that $\lambda_{A,i} = \|A - T\|$. Similarly, let S be an operator of rank

$$rank(S) \leq j - 1, \tag{A.1.11}$$

such that $\lambda_{B,j} = \|B - S\|$.

Given that

$$\|(A - T) \cdot (B - S)\| \leq \lambda_{A,i} \cdot \lambda_{B,j}, \tag{A.1.12}$$

it remains to estimate the rank of $(A - T) \cdot (B - S) - AB$. By writing,

$$(A - T) \cdot (B - S) - AB = -T(B - S) - AS, \tag{A.1.13}$$

we see that its rank is at most

$$j - 1 + i - 1 = k - 1. \tag{A.1.14}$$

□□

A.2 State space discretization

When we describe the dynamics of a particular system by analyzing physical laws that relate its variables of interest, we often find derivatives, given that these physical laws often yield a continous time model, which in state space form corresponds to

$$\frac{dx}{dt} = Ax(t) + Bu(t), \tag{A.2.1a}$$
$$y(t) = Cx(t). \tag{A.2.1b}$$

Then, a fundamental problem is how to describe a continous-time system connected to a computer via Analog-to-Discrete and Discrete-to-Analog converters.

Definition A.1 (Zero-Order-Hold discretization, see [3])**.**
Assume the Discrete-to-Analog converter is constructed such that it holds the analog signal constant until a new conversion is commanded, and h denotes the sampling time, i.e. the time elapsed between these conversions.

Then, the continous-time system on (A.2.1) *is sampled into*

$$x[kh+h] = A_d x[kh] + B_d u[kh], \tag{A.2.2a}$$
$$y[kh] = Cx[kh], \tag{A.2.2b}$$

with $kh \leq t < kh+h$, and where the relation between A_d, B_d and A, B is given by

$$A_d = e^{Ah} \tag{A.2.3}$$
$$B_d = \int_0^h e^{As} ds B \tag{A.2.4}$$

■

A.3 Hardware setup

The signal acquisition task for measuring pendulum and arm position of the Furuta pendulum described on Chapter 5 was made over the National Instruments CB-68LPR, whose connections diagram is shown on Figure A.1.

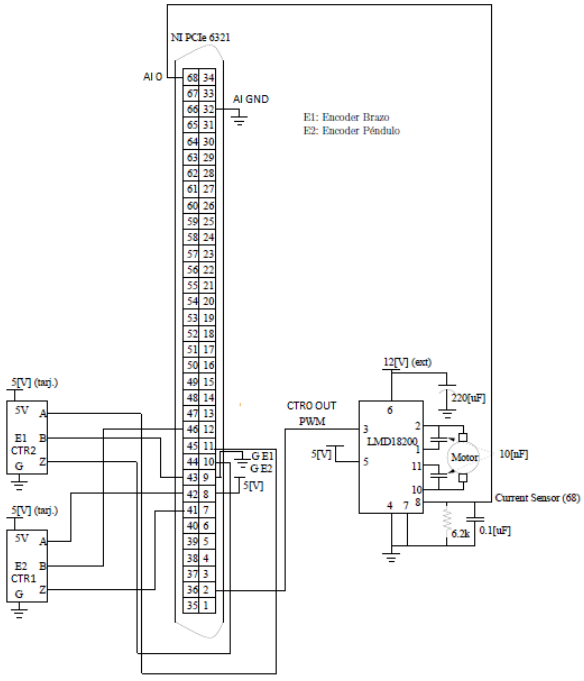


Figure A.1: Connections diagram

Then, before implementing the corresponding control program for the physical prototype using LabView, (National Instruments) Measurement & Automation Explorer has to be properly setup, as shown on Figure A.2.

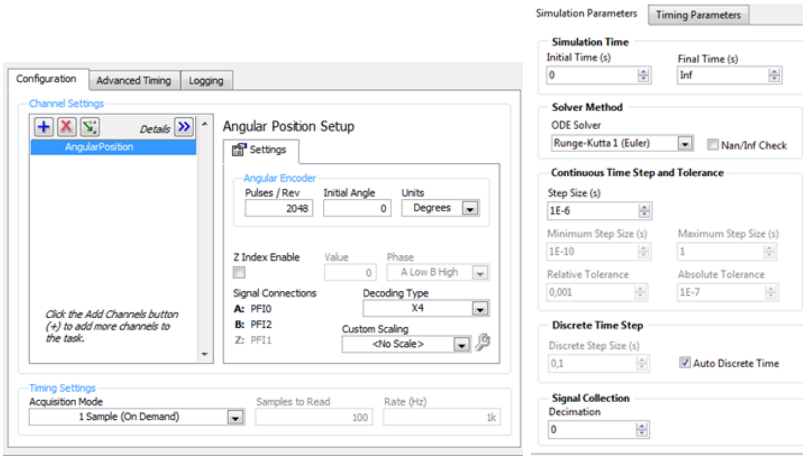


Figure A.2: NI Measurement & Automation Explorer

The subsystem corresponding to the dynamic state-feedback controller implemented in LabView is shown on Figure A.3, while the Front Panel of the final program that uses this subsystem is shown on Figure A.4.

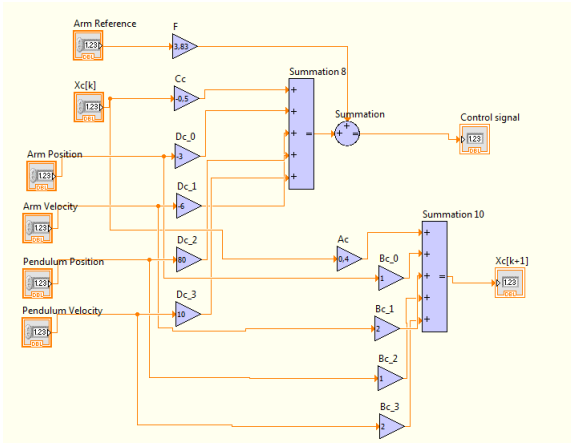


Figure A.3: Dynamic state-feedback controller in LabView

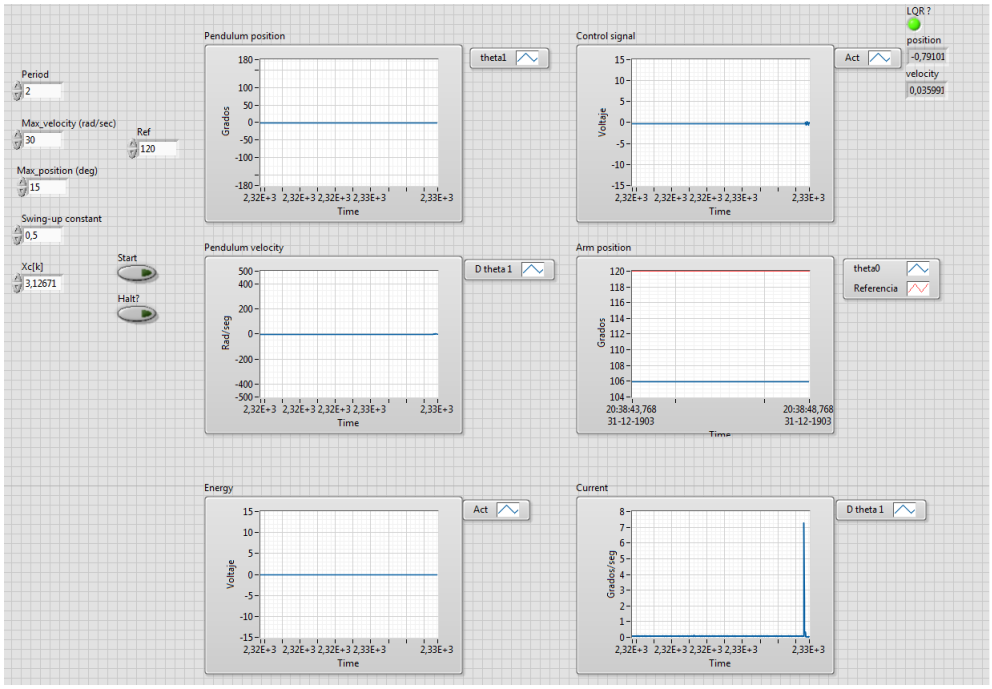


Figure A.4: Front panel for Furuta pendulum using dynamic state-feedback

A.4 Adaptive LQR code used on Furuta pendulum simulation

```
clc; clear all;

dt = 0.002;
time_horizon = 12/dt;
len_targets = 6;

A = [0 1 0 0 0; 0 -0.31 2.99 -0.02 81.47; 0 0 0 1 0; 0 -0.02 61.49 -0.5 63.88; 0
B = [0;0;0;0;100];
C = [1 0 0 0 0;0 0 1 0 0;0 0 0 0 1];

sys = ss(A, B, C, 0);
sysd = c2d(sys, 0.002, 'zoh');
A = sysd.a;
B = sysd.b;
C = sysd.c;

L = [-.31 -5.26 70.74 8.92 0.17]; % para Q solo control pendulo
%L = [-3.1 -6.07 80.02 10.08 0.18]; % para Q control_brazo
Q = [1 0 0 0 0; 0 10 0 0 0; 0 0 100 0 0; 0 0 0 10 0; 0 0 0 0 1];
R = 10;

ref_brazo = zeros(1,time_horizon);
u = zeros(2,time_horizon);
x = zeros(size(A,1),time_horizon);
x(:,1) = [0; 0; 15; 0; 0];

x_ml = zeros(size(A,1),time_horizon);
x_ml(:,1) = [0; 0; 15; 0; 0];

P = zeros(size(A,1),size(A,1));
L2 = -inv(R + B'*P*B)*B'*P*A;
d_target(1:len_targets,1:size(A,1)) = 0;

for i = 1:1:time_horizon-1
%     if 1/dt < i && i < 4/dt,
%         ref_brazo(1,i) = 90;
%     else
%         ref_brazo(1,i) = 0;
%     end
    x(:,i+1) = A*[x(1,i)-ref_brazo(1,i);x(2:5,i)] - B*L*[x(1,i)-ref_brazo(1,i);x(
    u(1,i) = -L*[x(1,i)-ref_brazo(1,i);x(2:5,i)];
    u(2,i) = L2*[x_ml(1,i)-ref_brazo(1,i);x_ml(2:5,i)];
```

```
    x_ml(:, i+1) = A*[x_ml(1,i)-ref_brazo(1,i);x_ml(2:5,i)] + B*u(2,i);
    d_target(1:len_targets-1, :) = d_target(2:len_targets, :);
    %d_target(5, :) = 2*x_ml(:, i)'*Q + 2*u'*R*L2 + 2*x_ml(:, i+1)'*P*(A + B*L2);
    d_target(len_targets, :) = [x_ml(1,i)-ref_brazo(1,i);x_ml(2:5,i)]'*Q + u(2,i)
    if mod(i,len_targets) == 0
        C = 2*[x_ml(1, i-len_targets+1:i)-ref_brazo(1,i-len_targets+1:i);x_ml(2:5
        q = [x_ml(1, i-len_targets+1:i)-ref_brazo(1,i-len_targets+1:i);x_ml(2:5, 
        P = inv(C)*q;
        L2 = -inv(R + B'*P*B)*B'*P*A;
    end
end

t = 1:1:time_horizon;
plot(dt.*t,x(3,:),'k');
hold on; grid on; title('pendulum')
plot(dt.*t,x_ml(3,:),'k--')

figure
plot(dt.*t,x(1,:),'k');
hold on; grid on; title('arm')
plot(dt.*t,mod(x_ml(1,:),360),'k--')

% figure
% plot(dt.*t,u(1,:),'k');
% hold on; grid on; title('control signal')
% plot(dt.*t,u(2,:),'k--')
```

REFERENCES

[1] Gabriel A Ahumada, Cristobal J Nettle, and Miguel A Solis. Accelerating q-learning through kalman filter estimations applied in a robocup ssl simulation. In *Robotics Symposium and Competition (LARS/LARC), 2013 Latin American*, pages 112–117. IEEE, 2013.

[2] Brian DO Anderson and John B Moore. *Optimal control: linear quadratic methods*. Courier Dover Publications, 2007.

[3] Karl J Astrom and Bjorn Wittenmark. Computer controlled systems: Theory and design, 1994.

[4] Karl Johan Åström and Katsuhisa Furuta. Swinging up a pendulum by energy control. *Automatica*, 36(2):287–295, 2000.

[5] KJ Åström. Introduction to stochastic control theory. 1970.

[6] Ahmad Taher Azar and Fernando E Serrano. Adaptive sliding mode control of the furuta pendulum. In *Advances and Applications in Sliding Mode Control systems*, pages 1–42. Springer, 2015.

[7] Enrique Barbieri and Rocio Alba-Flores. On the infinite-horizon lq tracker. *Systems & Control Letters*, 40(2):77–82, 2000.

[8] Richard Bellman. E. 1957. dynamic programming. *Princeton UniversityPress. Bellman-Dynamic programming1957*, 1957.

[9] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[10] Lionel Birglen, Thierry Laliberté, and Clément M Gosselin. *Underactuated robotic hands*, volume 40. Springer, 2007.

[11] Steven J Bradtke, B Erik Ydstie, and Andrew G Barto. Adaptive linear quadratic control using policy iteration. In *American Control Conference, 1994*, volume 3, pages 3475–3479. IEEE, 1994.

[12] Shuping Chen, Xunjing Li, and Xun Yu Zhou. Stochastic linear quadratic regulators with indefinite control weight costs. *SIAM Journal on Control and Optimization*, 36(5):1685–1702, 1998.

[13] Carlos E de Souza and Marcelo D Fragoso. On the existence of maximal solution for generalized algebraic riccati equations arising in stochastic control. *Systems & Control Letters*, 14(3):233–239, 1990.

[14] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. *IEEE computational intelligence magazine*, 1(4):28–39, 2006.

[15] Graham Clifford Goodwin, Stefan F Graebe, and Mario E Salgado. *Control system design*, volume 240. Prentice Hall New Jersey, 2001.

[16] Iraj Hassanzadeh and Saleh Mobayen. Controller design for rotary inverted pendulum system using evolutionary algorithms. *Mathematical Problems in Engineering*, 2011, 2011.

[17] Pingan He and Sarangapani Jagannathan. Reinforcement learning-based output feedback control of nonlinear systems with input constraints. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(1):150–154, 2005.

[18] Yulin Huang, Weihai Zhang, and Huanshui Zhang. Infinite horizon linear quadratic optimal control for discrete-time stochastic systems. *Asian Journal of Control*, 10(5):608–615, 2008.

[19] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *arXiv preprint cs/9605103*, 1996.

[20] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of Fluids Engineering*, 83(1):95–108, 1961.

[21] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[22] Bahare Kiumarsi, Frank L Lewis, Hamidreza Modares, Ali Karimpour, and Mohammad-Bagher Naghibi-Sistani. Reinforcement q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica*, 50(4):1167–1175, 2014.

[23] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[24] N Kumaresan and P Balasubramaniam. Optimal control for stochastic linear quadratic singular system using neural networks. *Journal of Process Control*, 19(3):482–488, 2009.

[25] Frank L Lewis and Derong Liu. *Reinforcement learning and approximate dynamic programming for feedback control*, volume 17. John Wiley & Sons, 2013.

[26] Frank L Lewis and Kyriakos G Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(1):14–25, 2011.

[27] Rémi Munos. Error bounds for approximate policy iteration. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 560–567, 2003.

[28] Reza Olfati-Saber. *Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles*. PhD thesis, Citeseer, 2000.

[29] Manuel Olivares and Pedro Albertos. A switched swing-up and stabilization control strategy for the rotating flywheel pendulum. In *Intelligent Control and Automation (WCICA), 2014 11th World Congress on*, pages 3874–3880. IEEE, 2014.

[30] Chunbin Qin, Huaguang Zhang, and Yanhong Luo. Online optimal tracking control of continuous-time linear systems with unknown dynamics by using adaptive dynamic programming. *International Journal of Control*, 87(5):1000–1009, 2014.

[31] M Ramírez-Neria, H Sira-Ramírez, R Garrido-Moctezuma, and Alberto Luviano-Juarez. Linear active disturbance rejection control of underactuated systems: The case of the furuta pendulum. *ISA transactions*, 53(4):920–928, 2014.

[32] Wilson J Rugh. *Linear system theory*, volume 2. prentice hall Upper Saddle River, NJ, 1996.

[33] Oscar A Silva and Miguel A Solis. Evolutionary function approximation for gait generation on legged robots. In *Nature-Inspired Computing for Control Systems*, pages 265–289. Springer, 2016.

[34] Sigurd Skogestad and Ian Postlethwaite. *Multivariable feedback control: analysis and design*, volume 2. Wiley New York, 2007.

[35] Torsten Söderström. *Discrete-time stochastic systems: estimation and control*. Springer, 2002.

[36] Miguel A Solis, Manuel Olivares, and Hector Allende. Stabilizing dynamic state feedback controller synthesis: A reinforcement learning approach. *Studies in Informatics and Control*, 25(2):245–254, 2016.

[37] Mark W Spong. Underactuated mechanical systems. In *Control problems in robotics and automation*, pages 135–150. Springer, 1998.

[38] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*. MIT Press, 1998.

[39] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

[40] Richard S Sutton, Andrew G Barto, and Ronald J Williams. Reinforcement learning is direct adaptive optimal control. *Control Systems, IEEE*, 12(2):19–22, 1992.

[41] Stephan Ten Hagen and Ben Kröse. Linear quadratic regulation using reinforcement learning. In *8th Belgian-Dutch Conference on Machine learning*, pages 39–46, 1998.

[42] Sebastian B Thrun. The role of exploration in learning control. *Handbook of intelligent control: Neural, fuzzy and adaptive approaches*, 1992.

[43] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards.* PhD thesis, University of Cambridge, 1989.

[44] Paul J Werbos. Neural networks for control and system identification. In *Decision and Control, 1989., Proceedings of the 28th IEEE Conference on*, pages 260–265. IEEE, 1989.

[45] Paul J Werbos. Approximate dynamic programming for real-time control and neural modeling. *Handbook of intelligent control: Neural, fuzzy, and adaptive approaches*, 15:493–525, 1992.

[46] Ronald J Williams and Leemon C Baird. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, Citeseer, 1993.

[47] William M Wonham. On a matrix riccati equation of stochastic control. *SIAM Journal on Control*, 6(4):681–697, 1968.

[48] H Yu, Y Liu, and T Yang. Closed-loop tracking control of a pendulum-driven cart-pole underactuated system. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 222(2):109–125, 2008.

[49] Weihai Zhang and Guiling Li. Discrete-time indefinite stochastic linear quadratic optimal control with second moment constraints. *Mathematical Problems in Engineering*, 2014, 2014.