People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University of Batna 2 – Mostefa Benboulaid
Faculty of Technology
Electronics Department

Dissertation

Prepared in LAAAS Laboratory

Presented in fulfillment of the requirement of the degree of
Doctor of Science in Electronics
Option: Robotics

Entitled:

Development of a robust fuzzy control and application to robotic systems

Presented by:

Nadia BOUNOUARA

Committee members:

Dr. Yassine ABDESSEMED	Prof.	University of Batna 2	Chairman
Dr. Kheireddine CHAFAA	Prof.	University of Batna 2	Advisor
Dr. Abdelaziz AOUICHE	MCA	University of Tebessa	Examiner
Dr. Abdelghani DJEDDI	MCA	University of Tebessa	Examiner

30/01/2023

Dedication

I dedicate this dissertation to my beloved father for always being there for me, supporting me, and believing in me.
In memory of my mother.

Acknowledgements

First and foremost, I would like to praise Allah for granting me countless blessings, including the overall well-being and good health which have particularly been essential for the completion of this work.

I would also like to express my deepest gratitude to my supervisor, **Prof. Kheireddine CHAFAA** from Mostefa Benboulaid; University of Batna 2, for his constant patience and motivation, his immense knowledge and unwavering support which guided me through each stage of this dissertation.

My heart heartfelt thanks equally go to the esteemed members of the Jury: the president **Prof. Yassine ABDESSEMED** from Mostefa Benboulaid; University of Batna 2, the members **Dr. Abdelaziz AOUICHE** from University of Tebessa, and **Dr. Abdelghani DJEDDI** from University of Tebessa for the time and consideration they dedicated to evaluate this humble work, and also for their invaluable comments and observations.

Last, but not least, I would like to extend my warmest appreciations to my family members and friends who have continuously encouraged and supported me during the years it took me to complete this dissertation.

Abstract

Our main contribution in this work concentrates on three objectives: 1) The synthesis of high gain observers for a class of uniformly observable nonlinear systems with sampled output and estimation of the high gain of the continuous–discrete time observer that corresponds to the minimum value of the cost function by using metaheuristic algorithms (BBO, PSO and GA). 2) Combination of PSO with Proportional-Derivative (PD) and Proportional-Integral-Derivative (PID) to design more efficient PD and PID controllers for robotic manipulators. Two PSO approaches were used: PSOCIW and PSOVIW. These approaches allow optimizing the controller parameters K_p (proportional gain), K_i (integral gain), and K_d (derivative gain) to achieve better performances. The proposed algorithms are performed in two steps: (1) First, PD and PID parameters are offline optimized by the PSO algorithm. (2) Second, the obtained optimal parameters are fed in the online control loop. Stability of the proposed scheme is established using the Lyapunov stability theorem, where we guarantee the global stability of the resulting closed-loop system, in the sense that all signals involved are uniformly bounded. 3) Proposition of an adaptive interval valued fuzzy controller for high performance direct vector-controlled induction motor drive. An interval valued controller compared with a type-1 fuzzy controller has the advantage that it can take into account the linguistic uncertainties present in the rules of the estimated models.

Résumé

Notre principale contribution dans ce travail se concentre sur trois objectifs: 1) La synthèse d'observateurs à grand gain pour une classe de systèmes non linéaires uniformément observables avec une sortie échantillonnée et l'estimation de la valeur du grand gain de l'observateur continu-discret qui correspond à la valeur minimale de la fonction objective en utilisant des algorithmes métaheuristiques (BBO, PSO et GA). 2) Combinaison du PSO avec le contrôleur Proportionnel-Dérivé (PD) et Proportionnel-Intégral-Dérivé (PID) pour concevoir des contrôleurs PD et PID plus efficaces pour les manipulateurs robotiques. Deux approches PSO ont été utilisées: PSOCIW et PSOVIW. Ces approches sont utilisées pour optimiser les paramètres du contrôleur K_p (gain proportionnel), K_i (gain intégral) et K_d (gain dérivé) pour obtenir de meilleures performances. Les algorithmes proposés sont exécutés en deux étapes: (1) Premièrement, les paramètres PD et PID sont optimisés hors ligne par l'algorithme PSO. (2) Deuxièmement, les paramètres optimaux obtenus sont ensuite introduits dans la boucle de contrôle en ligne. La stabilité du schéma proposé est établie en utilisant le théorème de stabilité de Lyapunov, où nous garantissons la stabilité globale du système en boucle fermée, dans le sens où tous les signaux impliqués sont uniformément bornés. 3) Proposition d'un contrôleur flou adaptatif à valeur d'intervalle pour l'entraînement de moteur à induction à commande vectorielle directe à hautes performances. Un contrôleur à valeurs d'intervalle par rapport à un contrôleur flou de type-1 a l'avantage de pouvoir prendre en compte les incertitudes linguistiques présentes dans les règles des modèles estimés.

ملخص

تركز مساهمتنا الرئيسية في هذا العمل على ثلاثة أهداف: ١) تركيب مراقبي المكاسب العالية لفئة من الأنظمة غير الخطية التي يمكن ملاحظتها بشكل موحد مع إخراج عينات وتقدير قيمة الكسب العالية لمراقب الزمن المستمر - المتقطع الذي يتوافق مع الحد الأدنى لقيمة دالة الهدف باستخدام خوارزميات اكتشاف الاستراتيجيات الجديدة (BBO ، PSO و GA). ٢) الجمع بين PSO والمشتق النسبي (PD) والمشتق النسبي المتكامل (PID) لتصميم وحدات التحكم PD و PID الأكثر كفاءة للروبوتات. تم استخدام طريقتين هما: $PSOCIW$ و $PSOVIW$. يتم استخدام هذه الأساليب لتحسين عوامل وحدة التحكم K_p (مكسب نسبي) ، K_i (مكسب متكامل) و K_d (مكسب مشتق) لتحقيق أداء أفضل. يتم تنفيذ الخوارزميات المقترحة في خطوتين: (١) أولاً ، تم تحسين عوامل كل من PD و PID بدون اتصال بواسطة خوارزمية PSO . (٢) ثانيًا ، يتم تغذية العوامل المثلى التي تم الحصول عليها في حلقة التحكم عن بعد. يتم إنشاء استقرار المخطط المقترح باستخدام نظرية الاستقرار $Lyapunov$ ، مع ضمان الاستقرار العام لنظام الحلقة المغلقة الناتج ، بحيث تكون جميع الإشارات المعنية محدودة بشكل موحد. ٣) اقتراح لفاصل زمني متكيف للتحكم الغامض من أجل محرك تحريضي عالي الأداء يتحكم فيه ناقل الحركة. تتميز وحدة التحكم ذات القيمة الفاصلة مقارنةً بوحدة التحكم الغامضة من النوع ١ بميزة أنها يمكن أن تأخذ في الاعتبار أوجه عدم اليقين اللغوية الموجودة في قواعد النماذج المقدرة.

List of Tables

List of Figures

Acronyms

BBO	Biogeography Based Optimization
COG	Centre Of Gravity
DOF	Degrees Of Freedom
EA	Evolutionary Algorithm
E	Emigration rate
FLC	Fuzzy Logic Control
FLS	Fuzzy Logic System
GA	Genetic Algorithm
HG	High Gain
HSI	High Suitability Index
I	Immigration rate
MAE	Mean Absolute Error
MIMO	Multiple Input Multiple Output
MSE	Mean Square Error
PSOCIW	PSO Constant Inertia Weight
PSOVIW	PSO Variable Inertia Weight
PSO	Particle Swarm Optimization
SC	Supervisory Control

Table of contents

GENERAL INTRODUCTION

A system is a combination of elements or parts which is considered as a single structure. These parts are generally defined with a particular set of variables, called the states of the system that completely determine the behavior of the system at a specific time. A dynamical system is a system which state changes with time. Specifically, the state of a dynamical system can be considered as an information storage or memory of past system events. The set of states of a dynamical system must be enough rich to completely determine the behavior of the system for any future time. Hence, a dynamical system consists of a set of possible states in a given space, together with a rule that determines the current state of the system in terms of past states.

A nonlinear system is a system that cannot be described by linear differential equations with constant coefficients. It is a set of nonlinear differential or difference equations, describing the temporal evolution of the variables constituting the system. This definition explains the complexity and diversity of nonlinear systems and the methods that apply to them. The modeling of nonlinear system depends on the physical nature of the system but also on the simplifying assumptions that it is possible to make.

Metaheuristic algorithms are computational intelligent models that have a wide range of applications in different fields of applied mathematics, engineering, and other sciences. A metaheuristic is a natural-inspired algorithm that contains a set of methods especially used for sophisticated solving optimization problems such as performance amelioration.

Metaheuristic algorithms like Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), and Biogeography-Based Optimization (BBO) are used for solving difficult optimization problems. The best solution is obtained throughout some parallel calculations on biological, animal or biogeographical populations.

A genetic algorithm is the most popular technique in evolutionary computation research and a heuristic method based on "Survival of the fittest". GAs are a sub-class of Evolutionary Algorithms (EAs) [1], they are inspired by natural selection and evolutionary genetics. GA was discovered as a useful tool for search and optimization problems.

Particle swarm optimization is one of the metaheuristic algorithms and a population-based stochastic optimization technique proposed in 1995 [2]. The aim of PSO is to search for the optimal solution in the search space. Its main idea is based on swarm intelligence. It uses two simple equations to explain the social attitude between a group of animals (fish, birds, etc.) [3, 4, 5, 6].

Biogeography based optimization is a biogeographical metaheuristic algorithm proposed in 2008 [7]. BBO is a new population based evolutionary algorithm and is based on an old theory of island biogeography that describes how biological species are distributed geographically. This method was inspired by the biogeographical concept of speciation or the evolution of new species, migration of species between islands and the extinction of species. An important study was performed in [8] where the authors provide a comparative study of these metaheuristic algorithms with all details of their overheads and complexities.

System states are very crucial in automatic control and their knowledge is essential for the implementation of control laws (state feedback or output feedback) and delivering relevant information on the state operation of the process. Usually, states are measured by sensors placed on the system. However, the system cost, volume and weight are augmented; also, the reliability of the whole system decreases with hard surroundings circumstances. Moreover, from a practical point of view, it is often very difficult to have access to all states in the cases when it is not always possible to reach them. Therefore, the necessity to use a supplementary dynamic system, called observer, which is responsible for estimating the state of the system is essential. Observers are excellent alternatives to physical sensors. Observer synthesis uses the relative data system, i.e., its dynamic model, its inputs and its measured outputs.

First articles dealing with the synthesis of observers for linear systems were published in the early 1960s by Kalman [9] and later by Luenberger [10], while a design of nonlinear observers began in the 1970s. These last years, designs of state observers have strongly mobilized the scientific community [11, 12, 13, 14]. An observer is a dynamic system either in continuous or discrete time that calculates the current states of a system from previous information considering both inputs and outputs of the system.

Among the most important observers that we can find in the literature is the high gain observer which has some advantages such as: (1) Design simplicity (2) Global or semi-global stability for large class of systems which means that their use can provide stability for any arbitrarily chosen initial conditions (3) Relatively fast (4) Robustness to modeling uncertainties and external disturbances. On the other side, there are some drawbacks of the high gain observers as their sensitivity to measurement noise and they suffer from the peaking phenomenon due to the high gain which produces an initial sharp spike in the response of the state estimates. This phenomenon can cause instability for some types of systems. For more details an interesting survey concerning high gain observers in feedback control can be found in [15].

The first contribution presented in this thesis is the application the above mentioned metaheuristic algorithms to adjust the high gain parameters of the continuous–discrete time observer in order to find the optimal estimation states. The framework of the proposed method is constituted of two steps. First, we present an optimized high gain structure which works in an offline manner which allows finding the optimal values of the high gain parameter. Second, obtained high gain value from step one is injected into the feedback linearization control loop running online for state estimation. The efficiency of optimization methods is investigated by presenting a short comparative study between BBO, PSO and GAs.

For decades, PD and PID controllers are the most widely used technique for controlling industrial processes. In this work, we introduce a new alternative to tune PD and PID parameters based on PSO optimization algorithm by optimizing the objective function defined by Mean Absolute Error (MAE). Minimizing the MAE is usually considered as a good performance index designing, and its optimization will adjust PD and PID parameters K_p, K_i and K_d. Note that optimization process is constrained in order to guarantee the stability of the system by using Lyapunov stability method. In this investigation we propose an alternative for the adaptation and optimization of K_p, K_i and K_d. For this propose we suggest to combine PSO with PID and PD in order to improve their performance.

The second contribution given in this dissertation concerns the optimization of PD and PID parameters using PSO algorithm. Inertia weight is a crucial parameter of the PSO algorithm which allows controlling its convergence. Two different inertia weights are considered in this paper: Constant Inertia Weight (CIW), and Variable Inertia Weight (VIW) which give us two strategies PSOCIW and PSOVIW. The aim of this investigation is to apply these two strategies for the optimization of PID (PD) free parameters to control a robotic system.

In the past decade, fuzzy logic control (FLC) strategy has been the focus of many studies and research for the control of nonlinear systems [16, 17, 18, 19]. One of the advantages of the fuzzy based control is that linguistic information can be directly incorporated into the controller without need an accurate mathematical model of the plant. Though, in presence of parameters variation of the plant, recourse to adaptive control is in most cases unavoidable. Adaptive fuzzy concept combines the robustness of fuzzy logic systems and the adaptation capabilities of adaptive control. Adaptive fuzzy controllers (AFC) provided an attracting approach to obtain the fuzzy parameters of a FLC by using a tuning algorithm. Model reference adaptive fuzzy control (MRAFC) technique has been applied usefully to control induction motor drives [20, 21, 22].

Type-2 fuzzy logic is an extension of type-1 fuzzy logic; it was introduced by Jerry M. Mendel [23] as an efficient tool which can outperform type-1 fuzzy logic in many situations, especially when uncertainties are present. Type-2 fuzzy logic was applied in many engineering areas, and the first application in adaptive control was proposed in [24, 25] where the authors gave how a type-2 fuzzy system can be used as a control system. Two approaches may be considered to reduce the computational burden while preserving the performance and the advantages of a type-2 fuzzy system: 1) Using a faster type-reduction method. Several algorithms are being developed for this purpose, including the modified enhanced Karnik-Mendel (MEKM) method [26], enhanced Karnik-Mendel (EKM) method, the enhanced iterative algorithm with stop condition (EIASC) method and many other methods reported in [27]. 2) Using a simple architecture with a reduced number of membership functions and rules.

The third contribution to this work is the development of a new indirect adaptive type-2 fuzzy controller (IAFC) for induction motors based on MEKM algorithm. The proposed scheme is based on the use of two controllers, the first one determines the feedback control by using type-2 fuzzy logic systems, and the second one generates the supervisory control (SC) action to stabilize the whole system when it tend to be unstable.

This thesis is composed of six chapters organized as follows:

The **first chapter** presents the general notions, stability and the dynamic models of nonlinear systems (the robot manipulator and the inverted pendulum-cart system).

In the **second chapter** we present observers design precisely the continuous-time high gain and the continuous discrete-time high gain observers of a class of MIMO nonlinear systems.

In the **third chapter** we present metaheuristic algorithms and the theory of optimization methods used in the present study.

The **fourth chapter** of this thesis presents the results of observer's high gain optimization by metaheuristic algorithms.

The **fifth chapter** presents the results of estimating the controller parameters K_p, K_i and K_d by metaheuristic algorithms to achieve better performances.

In the **sixth chapter** we present an indirect adaptive control based on type-2 fuzzy controller, with supervisory control to stabilize the nonlinear dynamical system.

Finally, we conclude this dissertation with some conclusions.

Bibliography

[1] Zhang L., Hu X., Wang Z., Sun F., and Dollell D.G. Fractional-order modeling and state-of-charge estimation for ultracapacitors. *Journal of Power Sources¡*. 314: 28–34, 2016. doi:10.1016/j.jpowsour.2016.01.066

[2] Kennedy J., and Eberhart R. Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*. 1942-1948, 1995. doi:10.1109/icnn.1995.488968

[3] Olivas F., Valdez F., and Castillo O. Dynamic parameter adaptation in particle swarm optimization using interval type-2 fuzzy logic. *Soft Computing*. 20(3): 1057–1070, 2014. doi:10.1007/s00500-014-1567-3

[4] Zeng N., Wang Z., Zhang H., and Alsaadi F.E. A novel switching delayed PSO algorithm for estimating unknown parameters of lateral flow immunoassay. *Cognitive Computation*. 8(2–8): 143–152, 2016. doi:10.1007/s12559-016-9396-6

[5] Zeng N., Hung Y.S., Li Y., and Du M. A novel switching local evolutionary PSO for quantitative analysis of lateral flow immunoassay. *Expert Systems with Applications*. 41(4): 1708–1715, 2014. doi:10.1016/j.eswa.2013.08.069

[6] Zeng N., Wang Z., Li Y., Du M., and Liu X. A hybrid EKF and switching PSO algorithm for joint state and Parameter estimation of lateral flow immunoassay models. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. 9(2): 321–329, 2012. doi:10.1109/tcbb.2011.140

[7] Simon D. Biogeography-Based Optimization. *IEEE Transactions on Evolutionary Computation*. 12(6): 702-713, 2008. doi:10.1109/tevc.2008.919004

[8] Kumar S., and Gaur P. Comparative response of filter using GA PSO BBO. *International Journal of Advanced Research in Computer and Communication Engineering.* 4(8): 83-86, 2015.

[9] Kalman R.E. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering.* 82(1): 35–45, 1960. doi:10.1115/1.3662552

[10] Luenberger D. An introduction to observers. *IEEE Transactions on Automatic Control.* 16(6): 596-602, 1971. doi:10.1109/tac.1971.1099826

[11] Farza M., M'Saad M., Ménard T., Fall M.L., Gehan O., and Pigeon E.Simple cascade observer for a class of nonlinear systems with long output delays. *IEEE Transactions on Automatic Control.* 60(12): 3338-3343, 2015. doi:10.1109/tac.2015.2427661

[12] Ma T., Che J., and Cao C. Control of Nonlinear Systems Using Filtered High-Gain Observer. *2016 International Symposium on Flexible Automation (ISFA).* 2016. doi:10.1109/isfa.2016.7790210

[13] Farza M., Ménard T., Ltaief A., Bouraoui I., M'Saad M., and Maatoug T. Extended high gain observer design for a class of MIMO non-uniformly observable systems. *Automatica.* 86: 138-146, 2017. doi:10.1016/j.automatica.2017.08.002

[14] Daldoul I., and Tlili A.S. Design of a High Gain Observer Optimization Method for the State Synchronization of Nonlinear Perturbed Chaotic Systems. *Journal of Computational and Nonlinear Dynamics.* 13(11), 2018. doi:10.1115/1.4041084

[15] Khalil H.K., and Laurent P. High-Gain Observers in Nonlinear Feedback Control. *International Journal of Robust and Nonlinear Control.* 24(6): 993–1015, 2013. doi:10.1002/rnc.3051

[16] Martinez F., Montiel H., and Jacinto E. Hybrid fuzzy-sliding grasp control for underactuated robotic hand. *TELKOMNIKA Telecommunication Computing Electronics and Control.* 17(4): 2070-2075, 2019.

[17] Lokriti A., Salhi I., Doubabi S., and Zidani Y. Induction motor speed drive improvement using fuzzy IP-self tuning controller, A real time implementation. *ISA transactions* 52(3): 406-417, 2013.

[18] Manickavasagam K. Fuzzy logic controller based single buck boost converter for solar PV cell. *International Journal of Applied Power Engineering.* 3(1): 1-8, 2014.

[19] Rafa S., Larabi A., Barazane L., Manceur M., Essounbouili N., and Hamzaoui A. Implementation of new fuzzy vector control of induction motor. *ISA transactions.* 53: 744-754, 2014.

[20] Gadoue S.M. , Giaouris D., and Finch J.W. MRAS sensorless vector control of an induction motor using new sliding-mode and fuzzy-logic adaptation mechanisms. *IEEE Transactions on Energy Conversion.* 25(2): 394-402, 2010.

[21] Bounar N., Boulkroune A., Boudjema F., M'Saad M., and Farza M. Adaptive fuzzy vector control for doubly-fed induction motor. *Neurocomputing.* 151: 756-769, 2015.

[22] Allaoui S., Chafaa K., Laamari Y., and Athamena B. Induction motor state estimation using tuned Extended Kalman Filter. *2015 4th International Conference on Electrical Engineering (ICEE), Boumerdes.* 1-5, 2015.

[23] Karnik N.N., Mendel J. M., and Liang Q. Type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems.* 7(6): 643-658, 1999.

[24] Chafaa K., Saidi L., Ghanai M., and Benmahammed K. Direct adaptive type-2 fuzzy control for nonlinear systems. *International Journal of Computaional Intelligence and Applications.* 6(3): 389-411, 2006.

[25] Chafaa K., Saidi L., Ghanai M., and Benmahammed K. Indirect adaptive interval type-2 fuzzy control for unknown nonlinear systems. *Int. Journal of Modelling, Identification and Control.* 2(2): 106-119, 2007.

[26] Salaken S.M., Khosravi A., and Nahavandi S. Modification on enhanced Karnik-Mendel algorithm. *Expert Systems with Applications.* 65: 283-291, 2016.

[27] Tai K., El-Sayed A.R., Biglarbegian M., Gonzalez C.I., Castillo O., and Mahmud S. Review of recent type-2 fuzzy controller applications. *Algorithms.* 9(2): 1-19, 2016.

Chapter 1

STABILITY AND DYNAMICS OF ROBOTIC SYSTEMS

Summary

1.1 Introduction

A nonlinear system is a system that is not linear. it is a system which output is not directly proportional to its input due to the interconnections and interdependencies within the system. Nonlinear systems theory is a modeling framework for describing nonlinear phenomena. There is not a general theory for these systems, but several methods were adapted to certain classes of nonlinear systems. Dynamic models are essential for understanding the nonlinear system dynamics in open-loop or for closed-loop control. These models are either derived empirically from data or from more fundamental relationships that rely on knowledge of the process.

Stability theory is crucial in dynamics and control systems. The stability of a dynamical system means that the system outputs and its internal signals are bounded within admissible limits or, the system outputs tend to an equilibrium state. A number of more accurate stability concepts, such as asymptotic stability, exponential stability, and global asymptotic stability.

Dynamic model determines the mathematical model which relates the input variables to the output variables. In general, such mathematical representation of the system is realized by ordinary differential equations. The system's mathematical model is obtained typically via one of the two following techniques.

- ***Analytical*** this procedure is based on physical laws of the system's motion. This methodology has the advantage of yielding a mathematical model as precise as is wanted.

- ***Experimental*** this procedure requires a certain amount of experimental data collected from the system itself. Typically one examines the system's behavior under specific input signals. The model so obtained is in general more imprecise than the analytic model since it largely depends on the inputs and the operating point. However, in many cases it has the advantage of being much easier and quicker to obtain.

This chapter presents the general concepts of nonlinear systems, stability theory and finally dynamic models of a manipulator robot and an inverted pendulum-cart system).

1.2 General notions of nonlinear systems

1.2.1 Nonlinear systems

Nonlinear systems can be modeled by a finite number of first-order ordinary differential equations [1]:

$$\begin{aligned}
\dot{x}_1 &= f_1\left(t, x_1, \ldots, x_n, u_1, \ldots, u_p\right)\\
\dot{x}_2 &= f_2\left(t, x_1, \ldots, x_n, u_1, \ldots, u_p\right)\\
&\vdots\\
\dot{x}_n &= f_n\left(t, x_1, \ldots, x_n, u_1, \ldots, u_p\right)\\
y &= h\left(t, x_1, \ldots, x_n, u_1, \ldots, u_p\right)
\end{aligned} \tag{1.1}$$

where

$\dot{x}_i$ denotes the derivative of x_i, with respect to the time variable t;

$u_1, \ldots, u_p$ are specified input variables;

$x_1, \ldots, x_n$ the state variables;

y is the system output.

The n first-order differential equations (1.1) can be expressed as one n-dimensional first-order vector differential equation called the state equation:

$$\begin{aligned}
\dot{x} &= f\left(x, t, u\right)\\
y &= h\left(t, x, u\right)
\end{aligned} \tag{1.2}$$

where

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, u = \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix}, f\left(t, x, u\right) = \begin{bmatrix} f_1\left(t, x, u\right) \\ \vdots \\ f_n\left(t, x, u\right) \end{bmatrix}$$

Equations (1.2) is called the state model.

Special Cases:

- An important special case of equation (1.2) is when the input u is identically zero. In this case, the equation takes the form

$$\dot{x} = f\left(x, t, 0\right) = f\left(x, t\right) \tag{1.3}$$

This equation is referred to as the unforced state equation.

- The second special case is obtaind when $f(x,t)$ is not a function of time. In this case, the equation is as follows:

$$\dot{x} = f(x) \tag{1.4}$$

in this case the system is said to be *autonomous*. Autonomous systems are invariant to shifts in the time origin in the sense that changing the time variable from t to $\tau = t - \alpha$ does not change the right-hand side of the state equation. Otherwise, the system is called *nonautonomous*.

1.2.2 Equilibrium points of a nonlinear system

The equilibrium point is a crucial principle when dealing with the state equation.

Definition 1.2.2.1. *point $x = x_e$ in the state space is said to be an equilibrium point of the autonomous system*

$$\dot{x} = f(x)$$

if it has the property that whenever the state of the system starts at x_e, it remains at x_e for all future time.

According to this definition, the equilibrium points of (1.4) are the real roots of the equation $f(x_e) = 0$.

1.3 Stability of a nonlinear system

The analysis of the behavior of nonlinear systems, especially in the vicinity of the equilibrium points, is the study of their stability [2].

Definition 1.3.0.1. *If the system is initially "slightly" disturbed from its point of equilibrium the system remains "close" to this point of equilibrium.*

In the sense of studying local or global stability, there are two methods, namely:

- Phase plane
- Lyapunov stability

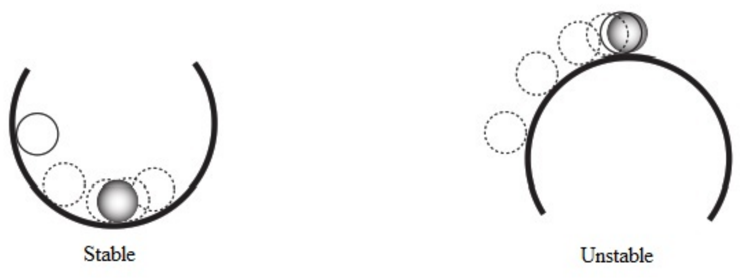


Figure 1.1: Illustration of stability and instability.

1.3.1 Phase plane

Phase plane analysis is a graphical method for studying second-order systems. The fundamental concept of the method is to generate motion trajectories corresponding to different initial conditions, and then look at the qualitative features of the trajectories.

The phase plane method is described by

$$\dot{x}_1 = f_1\left(x_1, x_2\right) \tag{1.5}$$

$$\dot{x}_2 = f_2\left(x_1, x_2\right) \tag{1.6}$$

where

x_1 and x_2 are the states of the system;

f_1, and f_2 are nonlinear functions of the states.

Due to initial conditions $x(0) = x_0$, Equations (1.5 and 1.6) define a solution $x(t)$. With time $t \in [0, \infty)$, the solution $x(t)$ can be represented geometrically as a curve in the phase plane. Such a curve is called a *phase plane trajectory*. A family of phase plane trajectories corresponding to different initial conditions is called a *phase portrait* of a system.

1.3.2 Lyapunov stability

System stability is distinguished by analyzing the response of a dynamical system to small disturbances in the system states. The most full contribution to the stability analysis of nonlinear dynamical systems was introduced in the late nineteenth century by the Russian mathematician *A.M. Lyapunov* in his work entitled *The General Problem of the Stability of Motion* [3, 4].

Definition 1.3.2.1. *A system is stable in the Lyapunov sense, if $\forall\varepsilon \succ 0$, $\exists\delta \succ 0$ such as $\|x_0\| \prec \delta \Rightarrow \|x(t)\| \prec \varepsilon$.*

This definition means that, regardless of the requirement ball of size ε, it is always possible to choose a certain sub-ball of size δ such that, for all the initial conditions included in this sub-ball, the resulting trajectories will be, at all times, included in the size ε requirement ball.

Definition 1.3.2.2. *A system is unstable in the sense of Lyapunov when it is not stable in the sense of definition (1.3.2.1).*

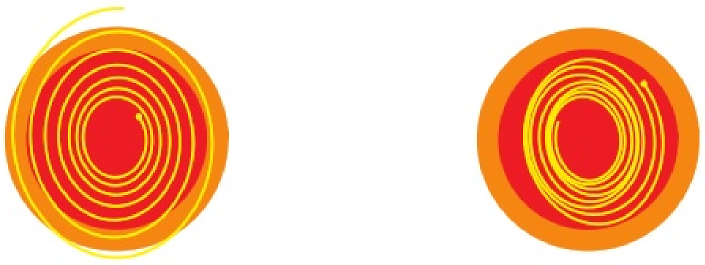


Figure 1.2: Unstable system (in the left), Stable system (in the right).

1.3.3 Lyapunov stability theory

We consider the general nonlinear autonomous dynamical system [5]

$$\dot{x}(t) = f(x(t)), \qquad x(0) = x_0, \qquad t \in [0, \tau) \tag{1.7}$$

where

$0 \leq \tau \leq \infty$;

$x(t) \in D \subseteq \Re^n$: is the system state vector;

D is an open set with $0 \in D$;

$f : D \to \Re^n$ is continuous on D. We assume that for every initial condition $x(0) \in D$ and every $\tau \succ 0$, the dynamical system (1.7) possesses a unique solution $x : [0, \tau) \to D$ on the interval $[0, \tau)$.

Definition 1.3.3.1. *i) The zero solution $x(t) \equiv 0$ to (1.7) is Lyapunov stable if, for all $\varepsilon \succ 0$, there exists $\delta = \delta(\varepsilon) \succ 0$ such that if $\|x(0)\| \prec \delta$, then $\|x(0)\| \prec \varepsilon$, $t \geq 0$ (see Figure 1.3).*

ii) The zero solution $x(t) \equiv 0$ to (1.7) is locally asymptotically stable if it is Lyapunov stable and there exists $\delta \succ 0$ such that if $\|x(0)\| \prec \delta$, then $\lim_{t\to\infty} x(t) = 0$ (see Figure 1.4).

iii) The zero solution $x(t) \equiv 0$ to (1.7) is globally asymptotically stable if it is Lyapunov stable and for all $x(0) \in \Re^n$, $\lim_{t\to\infty} x(t) = 0$.

iv) Finally, the zero solution $x(t) \equiv 0$ to (1.7) is unstable if it is not Lyapunov stable.

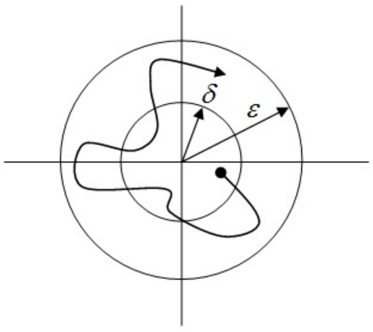


Figure 1.3: Lyapunov stability.

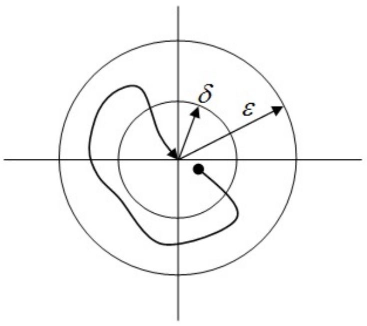


Figure 1.4: Asymptotic stability.

Figure 1.5 shows the asymptotic stability, Lyapunov stability, and instability notions of an equilibrium point. Clearly, exponential stability implies asymptotic stability and asymptotic stability implies Lyapunov stability.

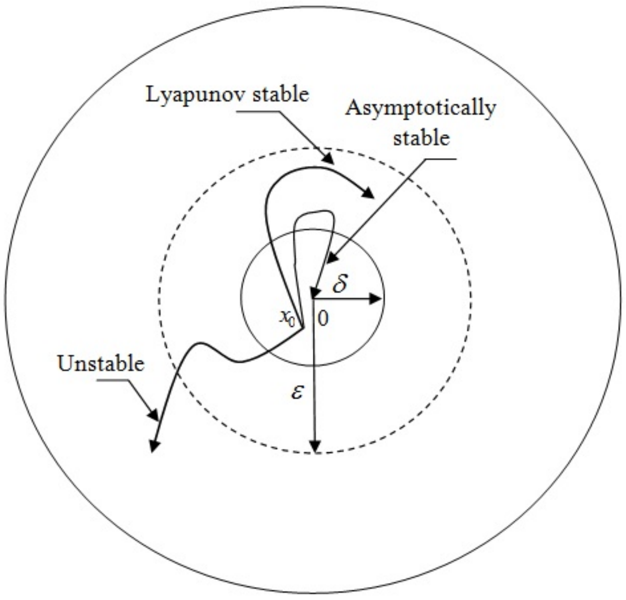


Figure 1.5: Asymptotic stability, Lyapunov stability, and unstability of an equilibrium point.

Theorem 1: Lyapunov's theorem

Consider the nonlinear dynamical system (1.7) and assume that there exists a continuously differentiable function $V : D \to \Re$ such that

$$V(0) = 0 \tag{1.8}$$

$$V(x) \succ 0, \qquad x \in D, \qquad x \neq 0 \tag{1.9}$$

$$V'(x) f(x) \leq 0, \qquad x \in D \tag{1.10}$$

Then the zero solution $x(t) \equiv 0$ of (1.7) is Lyapunov stable.
If, in addition,

$$V'(x) f(x) \prec 0, \qquad x \in D, \qquad x \neq 0 \tag{1.11}$$

Then the zero solution $x(t) \equiv 0$ to (1.7) is asymptotically stable.
Finally, if there exist scalars α, β, $\varepsilon \succ 0$, and $p \geq 1$, such that $V : D \to \Re$ satisfies

$$\alpha \|x\|^p \leq V(x) \leq \beta \|x\|^p, \qquad x \in D \tag{1.12}$$

$$V'(x) f(x) \leq -\varepsilon V(x), \qquad x \in D \tag{1.13}$$

Then the zero solution $x(t) \equiv 0$ of (1.7) is exponentially stable.

1.4 Dynamic models of robotic systems

The dynamic model of robot manipulators is typically derived in the analytic form, that is, using the laws of physics. Due to the mechanical nature of robot manipulators, the laws of physics involved are basically the laws of mechanics.
Robot manipulators are articulated mechanical systems composed of links connected by joints as illustrated in Figure 1.6. The joints are mainly of two types: revolute and prismatic [6].

On the other hand, from a dynamical systems viewpoint, an n DOF system may be considered as a multivariable nonlinear system. The term "multivariable" denotes the fact that the system has multiple (e.g. n) inputs (the forces and torques τ applied to the joints by the electromechanical, hydraulic or pneumatic actuators) and, multiple $(2n)$ state variables typically associated to the n positions q, and n joint velocities $\dot{q}$. In

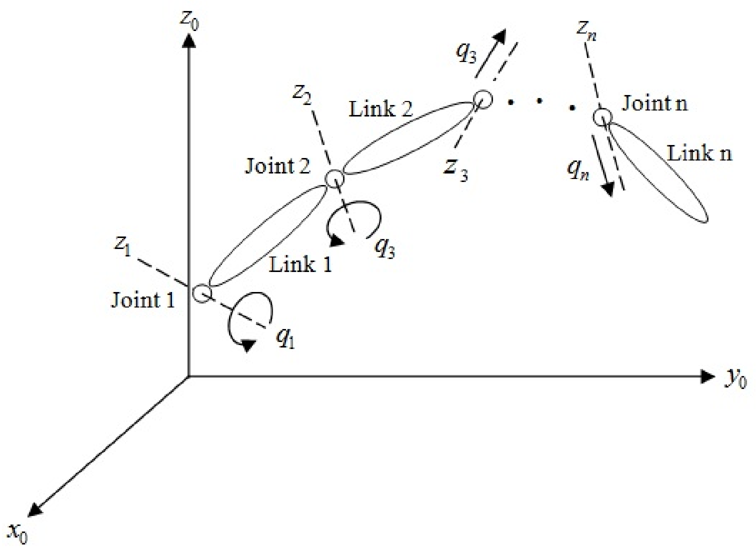


Figure 1.6: Diagram of an n DOF robot manipulator.

Figure 1.7 we depict the corresponding block-diagram assuming that the state variables also correspond to the outputs.

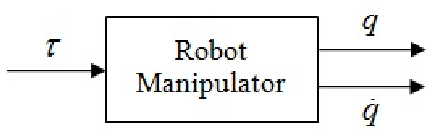


Figure 1.7: The input–output diagram of a robot manipulator.

1.4.1 Lagrange's equations of motion

One of the most common procedures followed in the computation of the dynamic model for robot manipulators, in closed form (i.e. not numerical), is the method which relies on the so-called *Lagrange's equations of motion* which was first reported in 1788.

The use of Lagrange's equations requires the notion of two important concepts: *kinetic* and *potential energies*.

Consider the robot manipulator with n links depicted in Figure 1.6. The total energy E of a robot manipulator of n DOF is the sum of the kinetic and potential energy functions, K and P respectively, i.e.

$$E\left(q, \dot{q}\right) = K\left(q, \dot{q}\right) + P\left(q\right)$$

where $q = \left[q_1, \ldots, q_n\right]^T$.

The *Lagrangian* $L\left(q, \dot{q}\right)$ of a robot manipulator of n DOF is the difference between its kinetic energy K and its potential energy P, that is,

$$L\left(q, \dot{q}\right) = K\left(q, \dot{q}\right) - P\left(q\right) \tag{1.14}$$

The Lagrange equations of motion for a manipulator of n DOF, are given by

$$\frac{d}{dt}\left[\frac{\partial L\left(q, \dot{q}\right)}{\partial \dot{q}}\right] - \frac{\partial L\left(q, \dot{q}\right)}{\partial q} = \tau$$

or in the equivalent form by

$$\frac{d}{dt}\left[\frac{\partial L\left(q, \dot{q}\right)}{\partial \dot{q}_i}\right] - \frac{\partial L\left(q, \dot{q}\right)}{\partial q_i} = \tau_i \qquad i = 1, \ldots, n \tag{1.15}$$

where τ_i correspond to the external forces and torques (delivered by the actuators). The use of Lagrange's equations in the derivation of the robot dynamics can be reduced to four main stages:

1) Computation of the kinetic energy function $K\left(q, \dot{q}\right)$.

2) Computation of the potential energy function $P\left(q\right)$.

3) Computation of the Lagrangian (1.14) $L\left(q, \dot{q}\right)$.

4) Computation of the Development of Lagrange's equations (1.15).

1.4.2 Dynamics of a two-link planar RR arm

We present in what follow an example of a Two-Link Planar RR Arm (2 DOF) shown in Figure 1.8 that illustrate the process of obtaining the robot dynamics by the use of Lagrange's equations of motion.

To determine its dynamics, examine Figure 1.8, where we have assumed that the link masses are concentrated at the ends of the links. The joint variable is [7]:

$$q = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T \tag{1.16}$$

and the generalized force vector is

$$\tau = \begin{bmatrix} \tau_1 & \tau_2 \end{bmatrix}^T \tag{1.17}$$

with τ_1, and τ_2 torques supplied by the actuators.

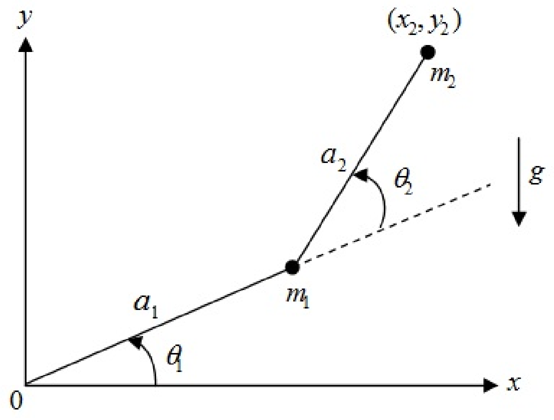


Figure 1.8: Two-link planar RR Arm.

a. Kinetic and potential energy

For link 1 the kinetic and potential energies are

$$K_1 = \frac{1}{2} m_1 a_1^2 \dot{\theta}_1^2 \tag{1.18}$$

$$P_1 = m_1 g a_1 \sin\theta_1 \tag{1.19}$$

For link 2 we have

$$x_2 = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) \tag{1.20}$$

$$y_2 = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2) \tag{1.21}$$

$$\dot{x}_2 = -a_1 \dot{\theta}_1 \sin\theta_1 - a_2 \left(\dot{\theta}_1 + \dot{\theta}_2\right) \sin(\theta_1 + \theta_2) \tag{1.22}$$

$$\dot{y}_2 = a_1 \dot{\theta}_1 \cos\theta_1 + a_2 \left(\dot{\theta}_1 + \dot{\theta}_2\right) \cos(\theta_1 + \theta_2) \tag{1.23}$$

so that the velocity squared is

$$\begin{aligned} \dot{v}_2 &= \dot{x}_2^2 + \dot{y}_2^2 \\ &= a_1^2 \dot{\theta}_1^2 + a_2^2 \left(\dot{\theta}_1 + \dot{\theta}_2\right)^2 + 2 a_1 a_2 \left(\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2\right) \cos\theta_2 \end{aligned} \tag{1.24}$$

Therefore, the kinetic energy for link 2 is

$$\begin{aligned} K_2 &= \tfrac{1}{2} m_2 v_2^2 \\ &= \tfrac{1}{2} m_2 a_1^2 \dot{\theta}_1^2 + \tfrac{1}{2} m_2 a_2^2 \left(\dot{\theta}_1 + \dot{\theta}_2\right)^2 + m_2 a_1 a_2 \left(\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2\right) \cos\theta_2 \end{aligned} \tag{1.25}$$

The potential energy for link 2 is

$$\begin{aligned} P_2 &= m_2 g y_2 \\ &= m_2 g \left[a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2)\right] \end{aligned} \tag{1.26}$$

b. Lagrange's equation

The Lagrangian for the entire arm is

$$\begin{aligned} L = K - P &= K_1 + K_2 - P_1 - P_2 \\ &= \tfrac{1}{2}(m_1+m_2)a_1^2\dot{\theta}_1^2 + \tfrac{1}{2}m_2a_2^2\left(\dot{\theta}_1+\dot{\theta}_2\right)^2 \\ &\quad + m_2a_1a_2\left(\dot{\theta}_1^2+\dot{\theta}_1\dot{\theta}_2\right)\cos\theta_2 \\ &\quad - (m_1+m_2)ga_1\sin\theta_1 - m_2ga_2\sin(\theta_1+\theta_2) \end{aligned} \tag{1.27}$$

The terms needed for (1.15) are

$$\frac{\partial L}{\partial\dot{\theta}_1} = (m_1+m_2)a_1^2\dot{\theta}_1 + m_2a_2^2\left(\dot{\theta}_1+\dot{\theta}_2\right) + m_2a_1a_2\left(2\dot{\theta}_1+\dot{\theta}_2\right)\cos\theta_2$$

$$\begin{aligned} \frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}_1} &= (m_1+m_2)a_1^2\ddot{\theta}_1 + m_2a_2^2\left(\ddot{\theta}_1+\ddot{\theta}_2\right) + m_2a_1a_2\left(2\ddot{\theta}_1+\ddot{\theta}_2\right)\cos\theta_2 \\ &\quad - m_2a_1a_2\left(2\dot{\theta}_1\dot{\theta}_2+\dot{\theta}_2^2\right)\sin\theta_2 \end{aligned}$$

$$\frac{\partial L}{\partial\theta_1} = -(m_1+m_2)ga_1\cos\theta_1 - m_2ga_2\cos(\theta_1+\theta_2)$$

$$\frac{\partial L}{\partial\dot{\theta}_2} = m_2a_2^2\left(\dot{\theta}_1+\dot{\theta}_2\right) + m_2a_1a_2\dot{\theta}_1\cos\theta_2$$

$$\frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}_2} = m_2a_2^2\left(\ddot{\theta}_1+\ddot{\theta}_2\right) + m_2a_1a_2\ddot{\theta}_1\cos\theta_2 - m_2a_1a_2\dot{\theta}_1\dot{\theta}_2\sin\theta_2$$

$$\frac{\partial L}{\partial\theta_2} = -m_2a_1a_2\left(\dot{\theta}_1^2+\dot{\theta}_1\dot{\theta}_2\right)\sin\theta_2 - m_2ga_2\cos(\theta_1+\theta_2)$$

Finally, according to Lagrange's equation, the arm dynamics are given by the two coupled nonlinear differential equations

$$\begin{aligned} \tau_1 &= \left[(m_1+m_2)a_1^2 + m_2a_2^2 + 2m_2a_1a_2\cos\theta_2\right]\ddot{\theta}_1 \\ &\quad + \left[m_2a_2^2 + m_2a_1a_2\cos\theta_2\right]\ddot{\theta}_2 - m_2a_1a_2\left(2\dot{\theta}_1\dot{\theta}_2+\dot{\theta}_2^2\right)\sin\theta_2 \\ &\quad + (m_1+m_2)ga_1\cos\theta_1 + m_2ga_2\cos(\theta_1+\theta_2) \end{aligned} \tag{1.28}$$

$$\begin{aligned} \tau_2 &= \left[m_2a_2^2 + m_2a_1a_2\cos\theta_2\right]\ddot{\theta}_1 + m_2a_2^2\ddot{\theta}_2 + m_2a_1a_2\dot{\theta}_1^2\sin\theta_2 \\ &\quad + m_2ga_2\cos(\theta_1+\theta_2) \end{aligned} \tag{1.29}$$

c. Manipulator dynamics

Writing the arm dynamics in vector form yields

$$\begin{aligned} & M(q)\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} m_2 a_1 a_2 \left(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2\right)\sin\theta_2 \\ -m_2 a_1 a_2 \dot{\theta}_1^2 \sin\theta_2 \end{bmatrix} \\ & - \begin{bmatrix} (m_1+m_2)\, g a_1 \cos\theta_1 + m_2 g a_2 \cos(\theta_1+\theta_2) \\ m_2 g a_2 \cos(\theta_1+\theta_2) \end{bmatrix} \\ & = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \end{aligned} \tag{1.30}$$

where

$$M(q) = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \tag{1.31}$$

with

$m_{11} = (m_1+m_2)\, a_1^2 + m_2 a_2^2 + 2m_2 a_1 a_2 \cos\theta_2$

$m_{12} = m_{21} = m_2 a_2^2 + m_2 a_1 a_2 \cos\theta_2$

$m_{22} = m_2 a_2^2$

These manipulator dynamics are in the standard form

$$M(q)\,\ddot{q} + C(q,\dot{q}) + G(q) = \tau \tag{1.32}$$

with $M(q)$ the inertia matrix, $C(q,\dot{q})$ the Coriolis/centripetal vector, and $G(q)$ the gravity vector.

d. State space modeling

A state space formulation of the system (see Figure 1.8) can be obtained by choosing a vector state

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} \quad \Rightarrow \quad \dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ \ddot{x}_1 \\ x_4 \\ \ddot{x}_3 \end{bmatrix}$$

where

$$\begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_3 \end{bmatrix} = M^{-1} \left(\begin{bmatrix} -m_2 a_1 a_2 \left(2x_2x_4 + x_4^2\right) \sin x_3 \\ m_2 a_1 a_2 x_2^2 \sin x_3 \end{bmatrix} + \begin{bmatrix} \left(m_1 + m_2\right) g a_1 \cos x_1 + m_2 g a_2 \cos\left(x_1 + x_3\right) \\ m_2 g a_2 \cos\left(x_1 + x_3\right) \end{bmatrix} \right) + M^{-1} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{1.33}$$

with

$m_{11} = \left(m_1 + m_2\right) a_1^2 + m_2 a_2^2 + 2 m_2 a_1 a_2 \cos x_3$

$m_{12} = m_{21} = m_2 a_2^2 + m_2 a_1 a_2 \cos x_3$

$m_{22} = m_2 a_2^2$

Simulations of positions tracking, speeds, errors and phases plane of the first and the second joint by the feedback linearization method are shown in Figure 1.9, Figure 1.10, Figure 1.11 and Figure 1.12.

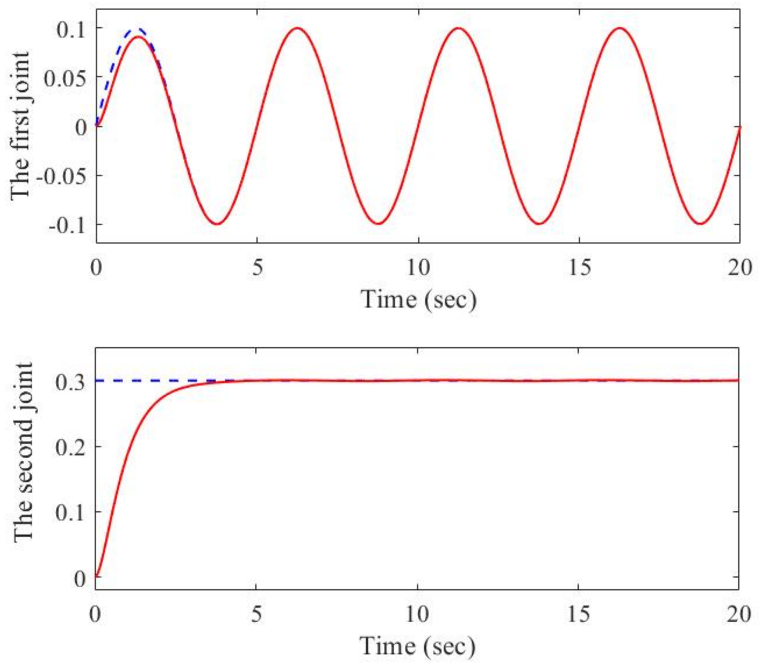


Figure 1.9: Positions tracking by feedback linearization.

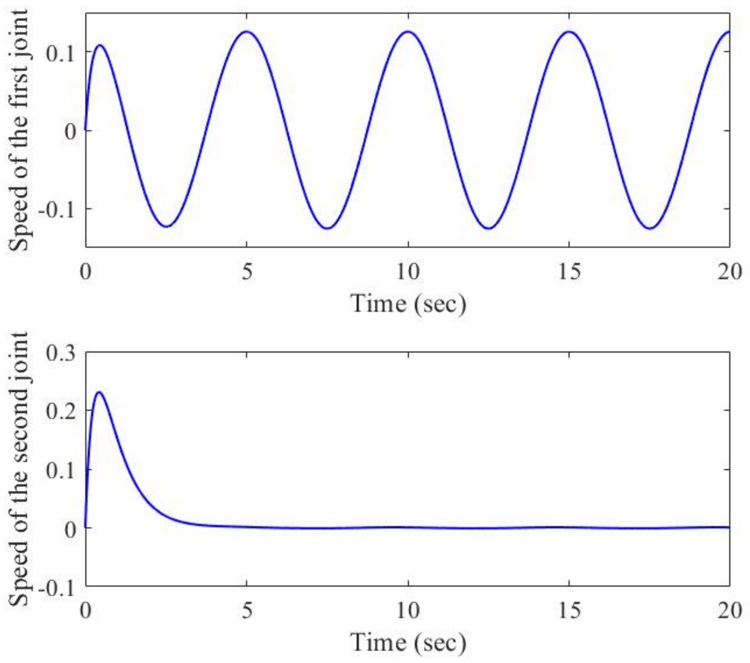


Figure 1.10: Speeds by feedback linearization.

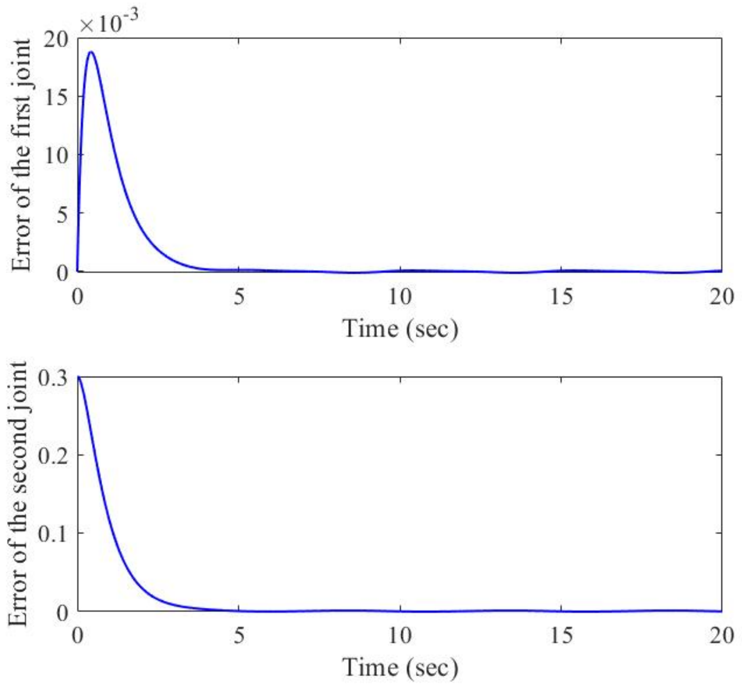


Figure 1.11: Errors by feedback linearization.

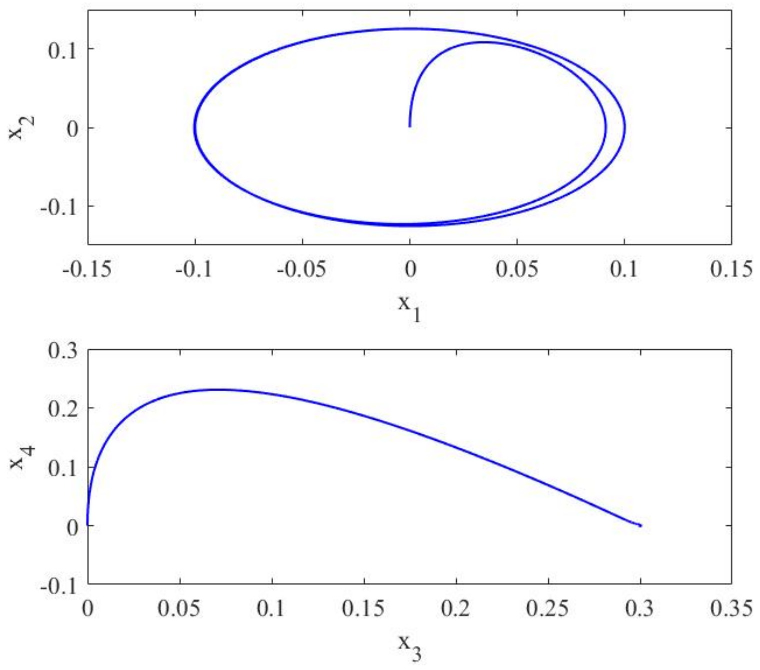


Figure 1.12: Phases plane by feedback linearization of the first and the second joint.

1.4.3 Dynamic model of the inverted pendulum-cart systems

Often inverted pendulums (1 DOF) are considered in combination with moving carts. The system of a single pendulum installed on a cart is drawn in Figure 1.13.

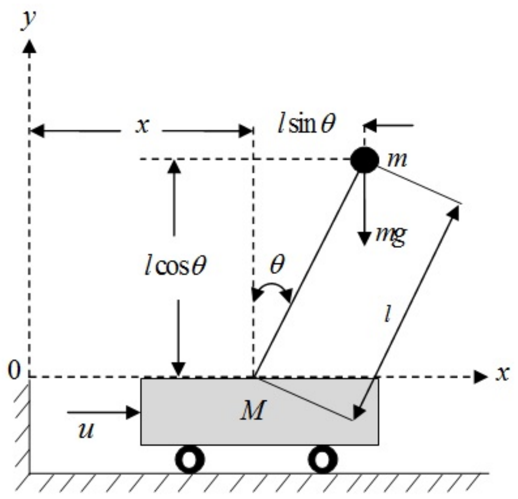


Figure 1.13: An inverted pendulum-cart system.

The dynamical model of the cart and the pendulum is often obtained by applying force analysis using free body diagrams and Newton's second law $F = ma$ or the

Lagrangian approach.

It is assumed that the pendulum rod is mass-less, and the hinge is frictionless. In such assumption, the whole pendulum mass is concentrated in the centre of gravity (COG) located at the center of the pendulum ball. For this case, the moment of inertia of the pendulum about its center of gravity is small (we assume $I = 0$). The cart mass and the ball point mass at the upper end of the inverted pendulum are denoted as M and m, respectively. There is an externally x-directed force on the cart, $u(t)$, and a gravity force acts on the point mass at all times. The coordinate system considered is shown in Figure 1.10, where $x(t)$ represents the cart position, and $\theta(t)$ is the tilt angle referenced to the vertically upward direction [8]. $\dot{x}$ and $\dot{\theta}$ represent velocity of the cart along the horizontal axis and angular velocity of the rod around the rod-cart connection point, respectively. Here (see 1.15),

$$\tau_1 = u, \qquad \tau_2 = 0 \tag{1.34}$$

The total kinetic energy of the pendulum–cart system can be written as

$$\begin{aligned} K &= \tfrac{1}{2}M\dot{x}^2 + \tfrac{1}{2}m\Big[\tfrac{d}{dt}\left(x + l\sin\theta\right)\Big]^2 + \tfrac{1}{2}m\Big[\tfrac{d}{dt}\left(l\cos\theta\right)\Big]^2 \\ &= \tfrac{1}{2}M\dot{x}^2 + \tfrac{1}{2}m\Big(\dot{x} + l\dot{\theta}\cos\theta\Big)^2 + \tfrac{1}{2}m\Big(-l\dot{\theta}\sin\theta\Big)^2 \\ &= \tfrac{1}{2}\left(M+m\right)\dot{x}^2 + ml\dot{x}\dot{\theta}\cos\theta + \tfrac{1}{2}ml^2\dot{\theta}^2 \end{aligned} \tag{1.35}$$

The total potential energy of the system, using the bottom of the pendulum rest position as the vertical reference point, can be written as

$$P = mgl\cos\theta \tag{1.36}$$

Therefore, the Lagrangian equation is given by

$$L = \frac{1}{2}\left(M+m\right)\dot{x}^2 + ml\dot{x}\dot{\theta}\cos\theta + \frac{1}{2}ml^2\dot{\theta}^2 - mgl\cos\theta \tag{1.37}$$

Substitute (1.37) into (1.15), we obtain [9]

$$\begin{aligned} &\left(M+m\right)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = u \\ &m\ddot{x}\cos\theta + ml\ddot{\theta} - mg\sin\theta = 0 \end{aligned} \tag{1.38}$$

The system model can be represented as follows

$$\begin{aligned} \ddot{x} &= \frac{ml\dot{\theta}^2\sin\theta - mg\sin\theta\cos\theta + u}{l\left(M+m-m\cos^2\theta\right)} \\ \ddot{\theta} &= \frac{-ml\dot{\theta}^2\sin\theta\cos\theta + Mg\sin\theta + mg\sin\theta - u\cos\theta}{l\left(M+m-m\cos^2\theta\right)} \end{aligned} \tag{1.39}$$

- **State space equations**

  Consider the state variables: $x_1 = \theta$ and $x_2 = \dot{\theta}$

  The state space equations for the inverted pendulum system can be written as

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{-mlx_2^2 \cos x_1 \sin x_1 + Mg \sin x_1 + mg \sin x_1}{l\,(M + m - m\cos^2\theta)} + \\ \qquad \dfrac{-\cos x_1}{l\,(M + m - m\cos^2\theta)} u \\ y = x_1 \end{cases} \tag{1.40}$$

Simulations of the tracking position, speed, error and phase plane by feedback linearization method are shown in Figure 1.14, Figure 1.15, Figure 1.16 and Figure 1.17.

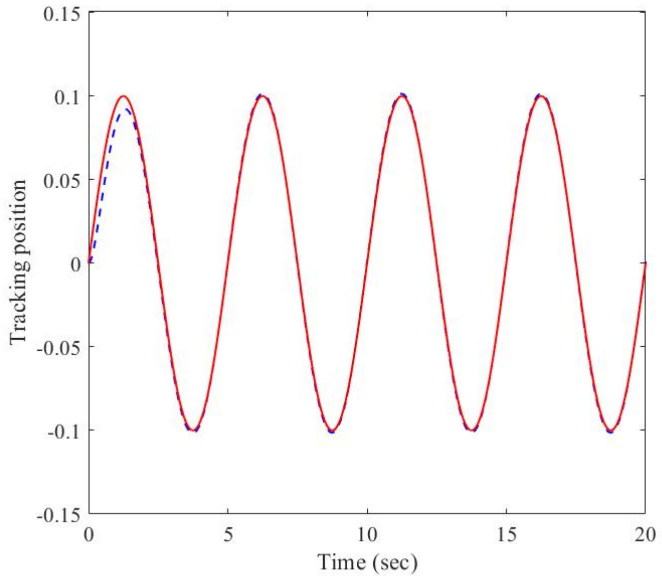


Figure 1.14: Tracking position by feedback linearization.

1.4.4 Three-phase induction motor

The three-phase induction motor (IM) was invented by *Mikhail Dolivo-Dobrovolsky* in 1889 [10]. Later, the squirrel-cage rotor was introduced by the same person. Induction motors have higher power densities compared to DC motors, and They are mechanically more robust which makes them the perfect motor in many applications.

The three-phase induction motor is an electromechanical system allowing the conversion of mechanical energy into electrical energy (generator mode) and the conversion of electrical energy into mechanical energy (motor mode). It consists of a stationary part, the stator, and a rotating part, the rotor. A stator with a three-phase distributed stator winding is shown in Figure 1.18.

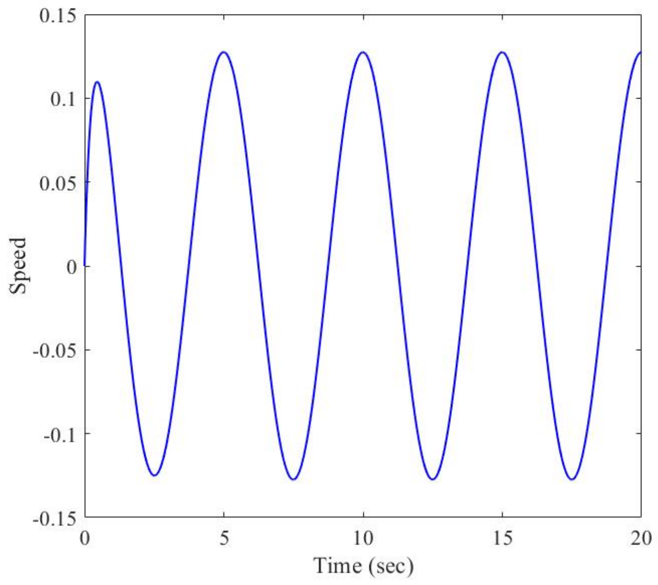


Figure 1.15: Speed by feedback linearization.

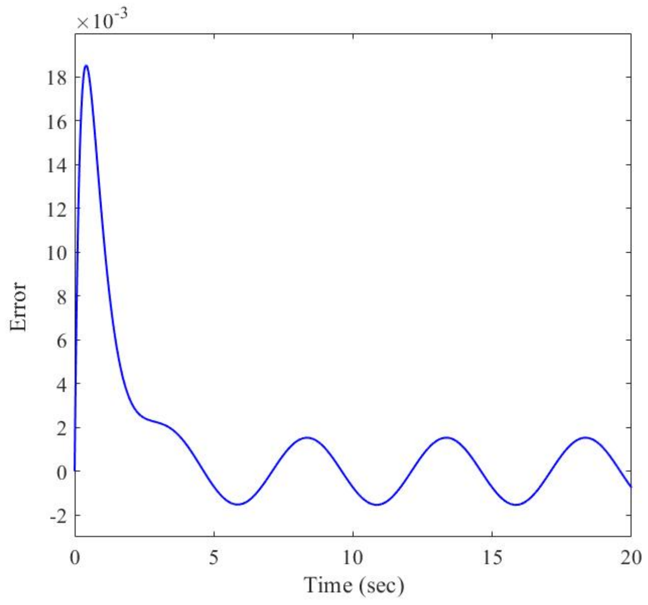


Figure 1.16: Error by feedback linearization.

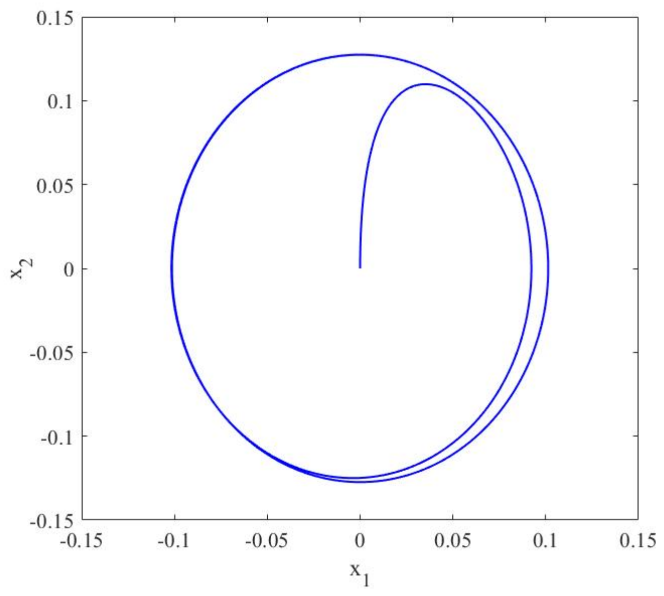


Figure 1.17: Phase plane.

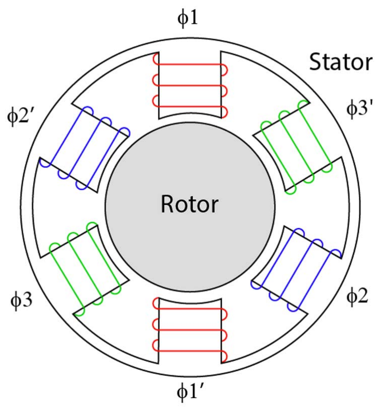


Figure 1.18: Illustration of the stator of an induction motor.

1.4.4.1 Operating principle

What allows the rotor to turn is the principle of rotating magnetic fields produced by alternating voltages. Three windings are arranged in the stator at 120° to each other, once powered three independent magnetic fields are created. The magnetic field rotating at a speed of rotation which is called speed of synchronism:

$$\Omega_s = \frac{\omega_s}{p} = 60\frac{f_s}{p}\,(tr/\min) \tag{1.41}$$

where

f_s : three-phase mains voltage frequency [Hz];

p : the number of pole pairs.

The rotor turns in the same direction as the rotating field, its speed of rotation is

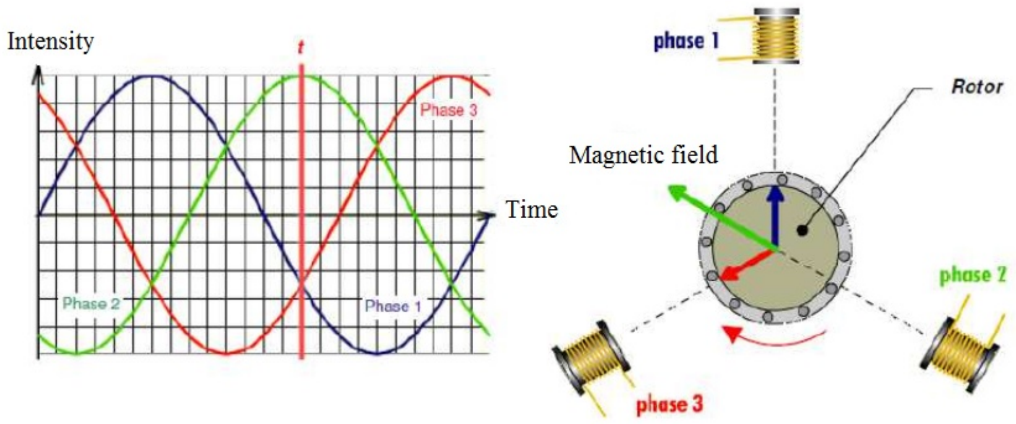


Figure 1.19: Diagram illustrating the operating principle of a three-phase induction motor.

slightly lower than that of the rotating field ($\Omega \prec \Omega_s$) [11].

Indeed, there is therefore always a difference in rotational speed between the stator and the rotor. This difference is called the slip (g) of the rotational speed of the rotor (Ω) compared to that of the rotating field (Ω_s) which characterizes asynchronous operation and has no unit:

$$g = \frac{\Omega_s - \Omega}{\Omega_s} \tag{1.42}$$

with: $\Omega = \frac{\omega}{p}$;

ω: rotor pulsation;

ω_s: stator pulsation;

Ω: angular rotational speed of the rotor;

Ω_s: angular speed of rotation of the rotating stator field.

1.4.4.2 Dynamic model

The dynamic modelling sets all the differential voltage, currents and flux linkages between the stationary stator as well as the moving rotor.

whether a three-phase induction motor with the rotor and the stator represented schematically by Figure 1.20 and whose phases are respectively marked a, b, c and A, B, C. the electrical angle θ, variable as a function of time, defines the instantaneous relave position between the magnetic axes of phases a and A chosen as deference axes [12].

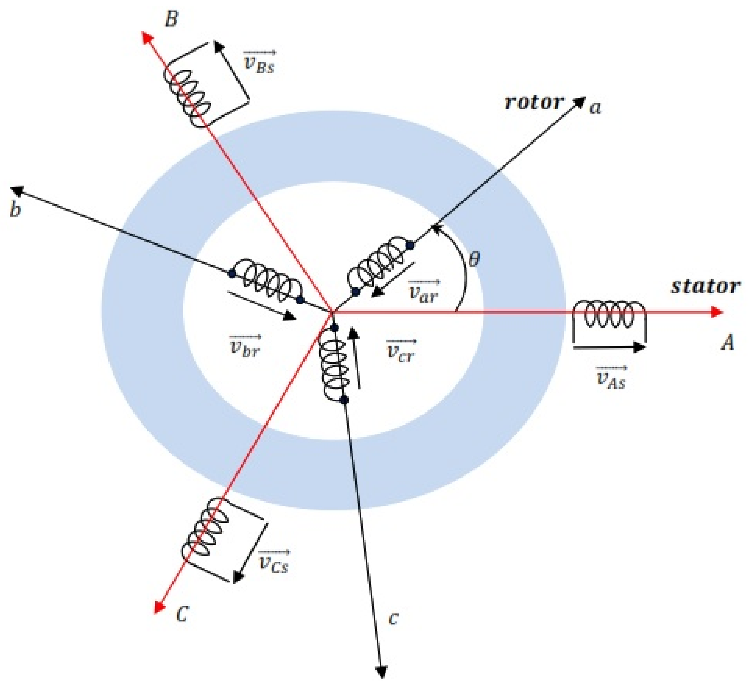

B
$\overrightarrow{v_{Bs}}$
rotor
a
b
θ
$\overrightarrow{v_{ar}}$
stator
A
$\overrightarrow{v_{br}}$
$\overrightarrow{v_{cr}}$
$\overrightarrow{v_{As}}$
$\overrightarrow{v_{Cs}}$
C
c

Figure 1.20: Schematic representation of a three-phase induction motor.

The equations of the machine are written as follows:

- **Electrical equations**

  By application of Faraday's law to each winding we have:

$$V = RI + \frac{d\phi}{dt} \tag{1.43}$$

  for all the phases of the machine represented by Figure 1.20, we deduce that:

  For the stator:

$$\begin{bmatrix} V_{As} \\ V_{Bs} \\ V_{Cs} \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} I_{As} \\ I_{Bs} \\ I_{Cs} \end{bmatrix} + \left(\frac{d}{dt}\right) \begin{bmatrix} \phi_{As} \\ \phi_{Bs} \\ \phi_{Cs} \end{bmatrix} \tag{1.44}$$

  For the rotor:

$$\begin{bmatrix} V_{ar} \\ V_{br} \\ V_{cr} \end{bmatrix} = \begin{bmatrix} R_r & 0 & 0 \\ 0 & R_r & 0 \\ 0 & 0 & R_r \end{bmatrix} \begin{bmatrix} I_{ar} \\ I_{br} \\ I_{cr} \end{bmatrix} + \left(\frac{d}{dt}\right) \begin{bmatrix} \phi_{ar} \\ \phi_{br} \\ \phi_{cr} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{1.45}$$

with

V, I, and ϕ: are voltage, current and flux respectively;

R_s and R_r: are the stator resistance and the rotor resistance, respectively.

- **Flux equations**

  The totalized fluxes coupled with the stator and rotor phases are expressed in matrix form as follows:

$$[\phi_{ABCs}] = [L_s]\,[I_{ABCs}] + [M_{sr}]\,[I_{abcr}] \tag{1.46}$$

$$[\phi_{abcr}] = [L_r]\,[I_{abcr}] + [M_{rs}]\,[I_{ABCs}] \tag{1.47}$$

where

$$[L_s] = \begin{bmatrix} l_s & m_s & m_s \\ m_s & l_s & m_s \\ m_s & m_s & l_s \end{bmatrix}, \quad [L_r] = \begin{bmatrix} l_r & m_r & m_r \\ m_r & l_r & m_r \\ m_r & m_r & l_r \end{bmatrix} \tag{1.48}$$

Due to the symmetry of the machine, we have:

$$[M_{sr}] = [M_{rs}]^T = M_0 \begin{bmatrix} \cos\theta & \cos\left(\theta + \frac{2\pi}{3}\right) & \cos\left(\theta - \frac{2\pi}{3}\right) \\ \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\theta & \cos\left(\theta + \frac{2\pi}{3}\right) \\ \cos\left(\theta + \frac{2\pi}{3}\right) & \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\theta \end{bmatrix} \tag{1.49}$$

with

l_s, (l_r): stator self inductance (rotor);

m_s, (m_r): mutual inductance between two stator windings (rotor);

M_0: maximum mutual inductance between a stator winding and a rotor winding.

Finally we get:

$$[V_{ABCs}] = [R_s]\,[I_{ABCs}] + \frac{d}{dt}\left([L_s]\,[I_{ABCs}] + [M_{sr}]\,[I_{abcr}]\right) \tag{1.50}$$

$$[V_{abcr}] = [R_r]\,[I_{abcr}] + \frac{d}{dt}\left([L_r]\,[I_{abcr}] + [M_{rs}]\,[I_{ABCs}]\right) \tag{1.51}$$

- **Mechanical equations**

  By applying the fundamental principle of dynamics we obtain:

$$J\frac{d\Omega}{dt} = C_{em} - f_v\Omega - C_r \tag{1.52}$$

where

C_{em}: electromagnetic torque;

$J\frac{d\Omega}{dt}$: inertia torque of the rotating masses referred to the diameter of the rotor;

$f_v\Omega$: Viscous friction torque;

C_r: resistant torque $[N.m]$;

J: total motor load inertia $[Kg.m^2]$;

ω: mechanical speed of rotation $\omega = \frac{\omega}{p}$ $[rad/s]$;

p: number of pole pairs of the motor;

f_v: viscous friction coefficient $[N.m/rad/s]$.

1.4.4.3 Machine model in axis system (d, q)

The set of previous equations (1.50), (1.51), and (1.52) reveal difficulties for analytical resolution because the trigonometric terms of the matrix of mutual inductances $[M_{sr}]$ vary according to the position θ. Solving this system of equations therefore comes up against insurmountable difficulties. This leads to the use of Park's transformation, which will make these terms independent of position and obtain a system of equations with constant coefficients which will facilitate its resolution [13].

- **Park Transform**

  The Park transform is a mathematical tool used to achieve a change of reference in a two-phase or three-phase axis system. It is generally used to go from a fixed frame linked to the stator of an electric machine to a rotating frame linked to its rotor or to the magnetic field (see Figure 1.21). A matrix $P(\theta_{obs})$ called Park, ensures this passage:

$$\begin{bmatrix} X_d \\ X_q \end{bmatrix} = P(\theta_{obs}) \begin{bmatrix} X_a \\ X_b \\ X_c \end{bmatrix} \tag{1.53}$$

with

$$P(\theta_{obs}) = \frac{\sqrt{2}}{3} \begin{bmatrix} \cos(\theta_{obs}) & \cos\left(\theta_{obs} - \frac{2\pi}{3}\right) & \cos\left(\theta_{obs} - \frac{4\pi}{3}\right) \\ -\sin(\theta_{obs}) & -\sin\left(\theta_{obs} - \frac{2\pi}{3}\right) & -\sin\left(\theta_{obs} - \frac{4\pi}{3}\right) \end{bmatrix} \tag{1.54}$$

The inverse transformation is given by:

$$P^{-1}(\theta_{obs}) = P^{T}(\theta_{obs}) = \frac{\sqrt{2}}{3}\begin{bmatrix} \cos(\theta_{obs}) & -\sin(\theta_{obs}) \\ \cos\left(\theta_{obs}-\frac{2\pi}{3}\right) & -\sin\left(\theta_{obs}-\frac{2\pi}{3}\right) \\ \cos\left(\theta_{obs}-\frac{4\pi}{3}\right) & -\sin\left(\theta_{obs}-\frac{4\pi}{3}\right) \end{bmatrix} \tag{1.55}$$

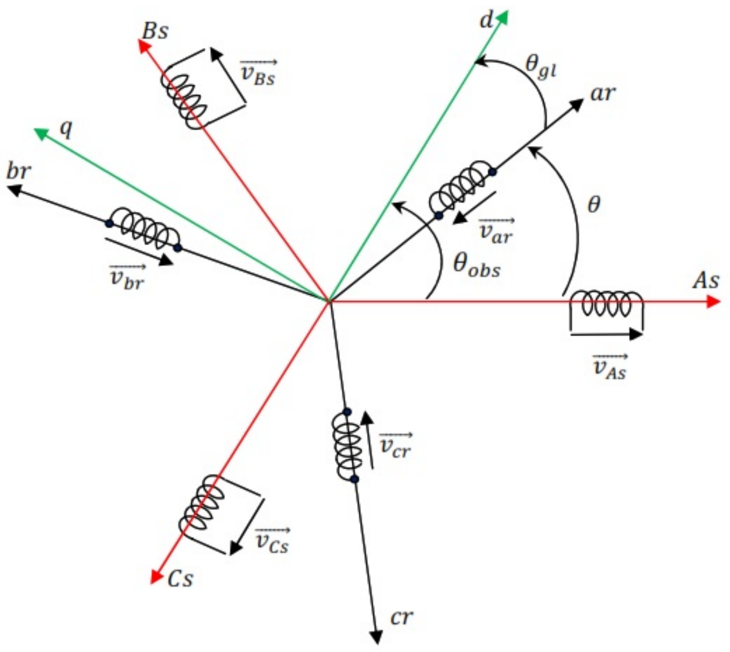

Bs
$\overrightarrow{v_{Bs}}$
d
θ_{gl}
ar
q
br
θ
$\overrightarrow{v_{ar}}$
θ_{obs}
$\overrightarrow{v_{br}}$
As
$\overrightarrow{v_{As}}$
$\overrightarrow{v_{cr}}$
$\overrightarrow{v_{Cs}}$
Cs
cr

Figure 1.21: Perform transformation from three-phase (ABC) to (dq) rotating reference frame.

The angle θ corresponds to the position of the coordinate system chosen for the transformation with:

- $\theta_{obs} = 0$ reference linked to the stator;
- $\theta_{obs} = \theta_s$ reference linked to the rotating field;
- $\theta_{obs} = \theta$ reference linked to the rotor.

- **Choice of the referential**

  There are different possibilities for choosing the orientation of the coordinate system (d, q) which generally depends on the objectives of the application. Depending on the choice of angular speed $\omega_{obs} = \frac{d\theta_{obs}}{dt}$, we obtain the following three frames of reference:

* $\frac{d\theta_{obs}}{dt} = \omega_{obs} = 0$: reference related to the stator.
* $\frac{d\theta_{obs}}{dt} = \omega_{obs} = \omega$: reference related to the rotor.
* $\frac{d\theta_{obs}}{dt} = \omega_{obs} = \omega_s$: reference related to the rotating field.

The design of vector control by flux orientation requires the last choice. The advantage of using this reference is to have constant quantities in steady-state, it is then easier to regulate it.

For a reference frame related to the rotating field we have:

$$\begin{cases} \frac{d\theta_s}{dt} = \omega_{obs} = \omega_s \\ \frac{d\theta_r}{dt} = \omega_{obs} - \omega = \omega_s - \omega = \omega_{gl} \end{cases} \tag{1.56}$$

The equations of stator and rotor voltages are written in Park's reference frame by:

$$\begin{cases} V_{ds} = R_s I_{ds} + \frac{d\phi_{ds}}{dt} - \omega_s \phi_{qs} \\ V_{qs} = R_s I_{qs} + \frac{d\phi_{qs}}{dt} + \omega_s \phi_{ds} \\ 0 = R_r I_{dr} + \frac{d\phi_{dr}}{dt} - (\omega_s - \omega)\, \phi_{qr} \\ 0 = R_r I_{qr} + \frac{d\phi_{qr}}{dt} - (\omega_s - \omega)\, \phi_{dr} \end{cases} \tag{1.57}$$

The components of stator and rotor fluxes are expressed by:

$$\begin{cases} \phi_{ds} = L_s I_{ds} + L_m I_{dr} \\ \phi_{qs} = L_s I_{qs} + L_m I_{qr} \\ \phi_{dr} = L_r I_{dr} + L_m I_{ds} \\ \phi_{qr} = L_r I_{qr} + L_m I_{qs} \end{cases} \tag{1.58}$$

The different expressions of the electromagnetic torque are expressed by the following equations as a function of the stator and rotor flux, and currents. The choice of which one to use depends on the chosen state vector.

$$\begin{cases} C_{em} = \phi_{ds} I_{qs} - \phi_{qs} I_{ds} \\ C_{em} = p\left(\phi_{qr} I_{dr} - \phi_{dr} I_{qr}\right) \\ C_{em} = p L_m \left(I_{qs} I_{ds} - I_{ds} I_{qr}\right) \\ C_{em} = p \frac{L_m}{L_r}\left(\phi_{dr} I_{qs} - \phi_{qr} I_{ds}\right) \end{cases} \tag{1.59}$$

1.4.4.4 State space representation

The induction motor can be modeled in the state space by a differential equations system of order 4 and a mechanical equation. By replacing the expression (1.58) in the equation (1.57) and after a long calculation we obtain the following system of equations:

$$\begin{cases} \frac{dI_{ds}}{dt} = -\left(\frac{1}{\sigma T_s} + \frac{1-\sigma}{\sigma T_r}\right) I_{ds} + \omega_s I_{qs} + \frac{1-\sigma}{\sigma L_m T_r}\phi_{dr} + \frac{1-\sigma}{\sigma L_m} p\Omega\phi_{qr} + \frac{1}{\sigma L_s} V_{ds} \\ \frac{dI_{qs}}{dt} = -\omega_s I_{ds} - \left(\frac{1}{\sigma T_s} + \frac{1-\sigma}{\sigma T_r}\right) I_{qs} - \frac{1-\sigma}{\sigma L_m} p\Omega\phi_{dr} + \frac{1-\sigma}{\sigma L_m T_r}\phi_{qr} + \frac{1}{\sigma L_s} V_{ds} \\ \frac{d\phi_{dr}}{dt} = \frac{L_m}{T_r} I_{ds} - \frac{1}{T_r}\phi_{dr} + (\omega_s - p\Omega)\,\phi_{qr} \\ \frac{d\phi_{qr}}{dt} = \frac{L_m}{T_r} I_{qs} + (\omega_s - p\Omega)\,\phi_{dr} - \frac{1}{T_r}\phi_{qr} \end{cases} \tag{1.60}$$

with

$\sigma = \left(1 - \frac{L_m^2}{L_s L_r}\right)$: is the dispersion coefficient;

$T_s = \frac{L_s}{R_s}$: stator time constant;

$T_r = \frac{L_r}{R_r}$: rotor time constant.

The general form of the equation of state space of the system is uniform and is written as follows:

$$\begin{cases} \dot{X} = AX + BU \\ Y = CX \end{cases} \tag{1.61}$$

where

X: is the state vector;

A: system state evolution matrix;

B: system control matrix;

U: control vector;

Y: output matrix;

C: output vector.

The model of the machine in the frame (d, q) linked to the rotating field for a vector of state $X = [I_{ds}, I_{qs}, \phi_{dr}, \phi_{qr}]^T$ and of control voltage $U = [V_{ds}, V_{qs}]^T$ is then defined by the triplet of the matrices A, B, C as follows:

$$A = \begin{bmatrix} -\left(\frac{1}{\sigma T_s} + \frac{1-\sigma}{\sigma T_r}\right) & \omega_s & \frac{1-\sigma}{\sigma L_m T_r} & \frac{1-\sigma}{\sigma L_m} p\Omega \\ -\omega_s & -\left(\frac{1}{\sigma T_s} + \frac{1-\sigma}{\sigma T_r}\right) & -\frac{1-\sigma}{\sigma L_m} p\Omega & \frac{1-\sigma}{\sigma L_m T_r} \\ \frac{L_m}{T_r} & 0 & \frac{-1}{T_r} & (\omega_s - p\Omega) \\ 0 & \frac{L_m}{T_r} & -(\omega_s - p\Omega) & \frac{-1}{T_r} \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{\sigma L_s} & 0 \\ 0 & \frac{1}{\sigma L_s} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The output vector is defined by:

$$Y = [I_{ds},\, I_{qs}]^T$$

1.5 Conclusion

In this chapter, we have described the nonlinear systems and their basic concepts. We have also studied the stability of a nonlinear system. In terms of stability, a system nonlinear is described as stable if starting the system somewhere near its desired operating point means that it will stay around the point ever after, that's why we have presented theorems related to the Lyapunov stability of the nonlinear dynamical systems. On the other hand, the dynamics model of the robot manipulator and the inverted pendulum-cart system have been discussed by the Lagrange equations of motion approach based on the Lagrange formulation, where the state spaces of both systems associated with these equations are described. To conclude, we have confirmed this by simulating tracking position, speed, error, and phase plane. Also the modeling of the induction motor was presented in the reference (d, q). This modeling is linked to the rotating field. The model presented has been given in a general two-phase benchmark in order to reduce its complexity. A so-called Park transformation was used to transform the three-phase machine into an equivalent two-phase machine.

Bibliography

[1] Marquez H. J. (2003). *Nonlinear Control Systems*. Wiley-Interscience.

[2] Müllhaupt P. (2007). *Introduction à l'analyse et à La Commande des systémes non linèaires*. Lausanne.

[3] Lyapunov A. M. (1892). *The General Problem of the Stability of Motion*. Kharkov, Russia: Kharkov Mathematical Society.

[4] Lyapunov A. M. (1992). *The General Problem of the Stability of Motion*. (A. T. Fuller, trans. and ed.). Washington, DC: Taylor and Francis.

[5] Haddad W. M., and Chellaboina V. S. (2008). *Nonlinear Dynamical Systems and control: A Lyapunov-based approach*. Princeton University Press.

[6] Kelly R., Santibanez V., and Loria A. (2005). *Control of robot manipulators in joint space*. Springer.

[7] Lewis F. L., Dawson D. M., and Abdallah C. T. (2004). *Robot manipulator control: Theory and practice*. Marcel Dekker.

[8] Prasad L. B., Tyagi B., and Gupta H. O. Optimal Control of Nonlinear Inverted Pendulum System Using PID Controller and LQR: Performance Analysis Without and With Disturbance Input. *International Journal of Automation and Computing*. 11(6): 661–670, 2014. doi:10.1007/s11633-014-0818-1

[9] Slotine J. E., and Li W. (1991). *Applied nonlinear control*. Englewood Cliffs, NJ: Prentice-Hall.

[10] Boldea I., and Nasar, S. (2010). *The induction machines design handbook*. Boca Raton, Fla: Taylor and Francis.

[11] Barret P. (1987). *Régimes transitoires des machines tournantes électriques.* Paris: Editions Eyrolles.

[12] Chatelain J. (1989). *Machines électriques.* Lausanne: PPUR Presses Polytechniques et Universitaires Romandes.

[13] Caron J., and Hautier J. (1995). *Modélisation et commande de la machine asynchrone.* Paris: Editions Technip.

CHAPTER 2

OBSERVERS DESIGN

Summary

2.1 Introduction

An observer or state reconstructor is a continuous or discrete-time dynamic system that calculates the estimations of current values of a system from previous information considering both inputs and outputs of the system. The observer is ,therefore, considered as a software sensor and its implementation allow the use of a minimum of physical sensor, or information redundancy, or diagnostics. In addition, it makes it possible to estimate quantities that are difficult to access or even not measurable. During the last decades, observer synthesis has undergone a very rapid increase for both linear ([1, 2]) and nonlinear systems ([3, 4, 5, 6, 7, 8, 9, 10]) with several different approaches. Over the last decades, observers have been one of the most interesting and investigated topics in the nonlinear systems community. In the observer theory, a key role is played by the high gain observers.

This chapter presents the objectives, principle and classification of observes and describes the design of the continuous-time high gain and discrete-time high gain observer for nonlinear systems.

2.2 Objectives of observers

An observer is a dynamic system that can be called a computing sensor because it is often installed on a calculator in order to reconstitute or estimate in real time the current state of a system; from available measurements; system inputs and prior knowledge of the model and system outputs. It allows us to follow the evolution of the condition as information about the system.

The need for internal information can be motivated by various objectives:

- **Monitoring (fault detection)** of the process through the differences between the behavior of the observer and that of the process.
- **Modeling (identification)** of the process by estimating constant quantities that define the model parameters.
- **Control** of the process which necessarily requires knowledge of its internal state.

All of these objectives are actually needed when trying to keep a system under control, as illustrated in Figure 2.1 [11].

2.3 Observers principle

The objective of an observer is to reconstruct quantities of which we cannot or do not wish to measure the state by a direct method (see Figure 2.2).

From this functional diagram of an observer (see Figure 2.3), we can implement all types of observers, their difference being only in the synthesis of the gain matrix L

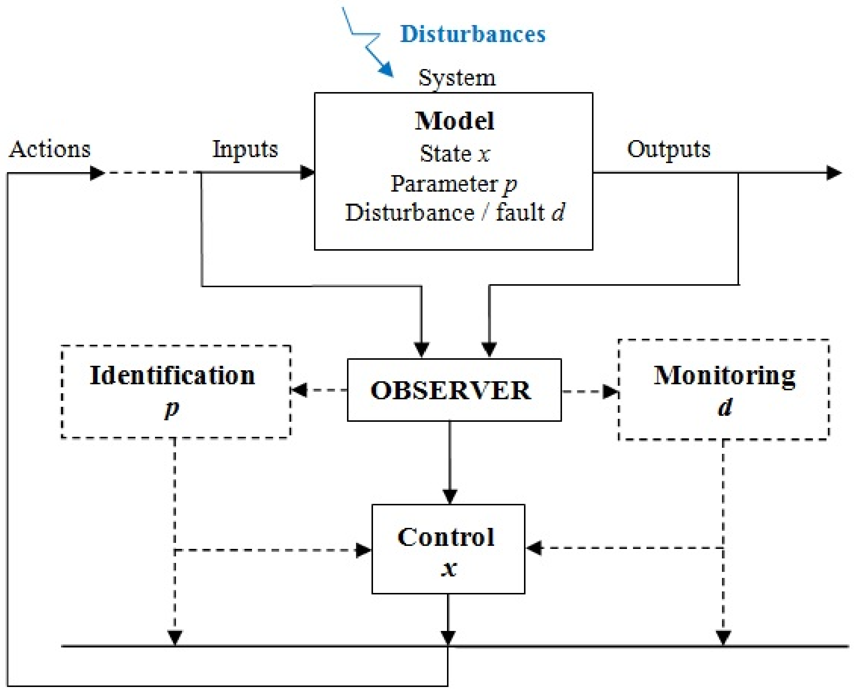


Figure 2.1: Observer: the essential part in the control.

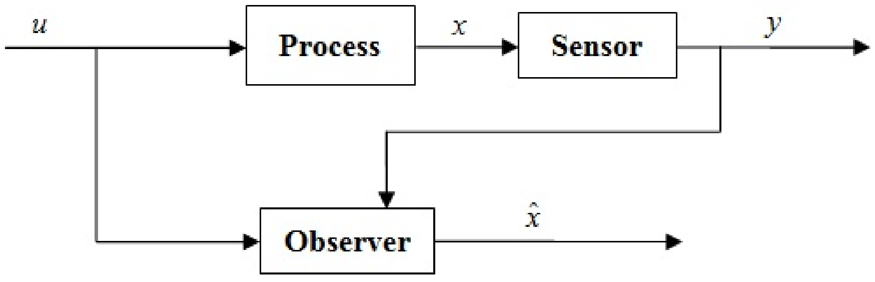


Figure 2.2: Principle of a state observer.

which must be adapted to the properties of the system whose we want to observe the states.

2.4 Classification of observers

There are many observation techniques. They differ depending on the environment considered (deterministic or stochastic), the nature of the system considered (linear or nonlinear), and finally, depending on the number of states to be observed (reduced order observers and full order observers). For reduced order observers we observe only

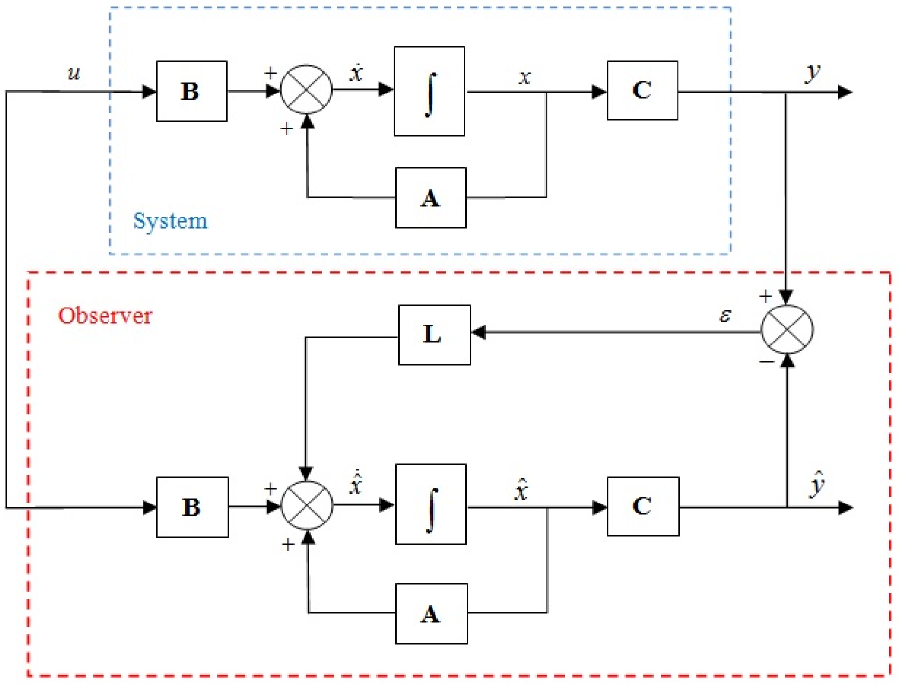


Figure 2.3: Functional diagram of an observer.

part of the state vector while for full order observers we observe the entire state vector.

In fact, whatever the classification criterion to consider, observers generally classified into two large families: deterministic and stochastic.

2.4.1 Deterministic observers

They are the observers who do not take into account the noise of measurements and the random fluctuations of state variables: the environment is deterministic. These observers are therefore characterized most of the time by sensitivity to disturbances and parametric variations. Among these observers we can cite the Luenberger observer, adaptive observer and the MRAS observer.

2.4.1.1 Luenberger observer

This observer makes it possible to reconstitute the state of an observable system from the measurement of its inputs and outputs. It also allows the estimation of variable or unknown parameters of a system. It is often used in feedback control, where the state vector is not known. Its operation is illustrated in the Figure 2.3.

The different quantities mentioned in Figure 2.3 represent:

u: input vector of the real system and of the observer;

x: state vector made up of the quantities to be observed;

y: output vector whose components are measurable;

$\hat{x}$ and $\hat{y}$: are respectively the estimate of the state and output vectors x, y, respectively.

In order to illustrate the principle of an observer, we consider a system described by the following equations of state:

$$\begin{cases} \dot{x}\,(t) = Ax(t) + Bu(t) \\ y\,(t) = Cx(t) \end{cases} \tag{2.1}$$

The observer represents a copy of the original system plus a gain term. So, it is described as follows:

$$\begin{cases} \dot{\hat{x}}\,(t) = A\hat{x}(t) + Bu(t) + L\varepsilon \\ \hat{y}\,(t) = C\hat{x}(t) \end{cases} \tag{2.2}$$

The output vector y is compared to the equivalent vector given by the observer to ensure closed-loop operation. Thus, we define a variable which is the error of the observation $\varepsilon = y(t) - \hat{y}(t)$. This later is multiplied by the matrix L and sent to the input of the observer to influence the estimated states $\hat{x}$. For a judicious choice of the matrix of gains L, one can modify the dynamics of the observer (which depends on the eigenvalues of the matrix $[A - LC]$), and consequently make evolve the speed of convergence of the error towards zero.

The dynamics of the estimation error; $e(t) = x(t) - \hat{x}(t)$ has the expression:

$$\dot{e}\,(t) = (A - LC)\,e(t) \tag{2.3}$$

2.4.1.2 Adaptive observer

An adaptive observer is composed of a state observer of a model whose parameters are unknown and an algorithm for online adaptation of these model parameters.

The structure of the adaptive observer is shown in Figure 2.4

2.4.1.3 ARMS observer (Adaptive Reference Model System)

The MRAS (Adaptive Reference Model System) is based on the comparison of the outputs of two estimators. The first, which does not introduce the quantity to be

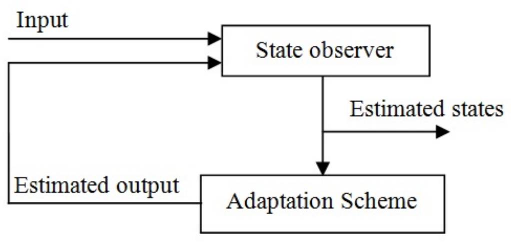


Figure 2.4: Structure of the adaptive observer.

estimated, is called the reference model and the second is the adjustable model. The error between these two models drives an adaptation mechanism. The latter is used in the adaptive model (see Figure 2.5).

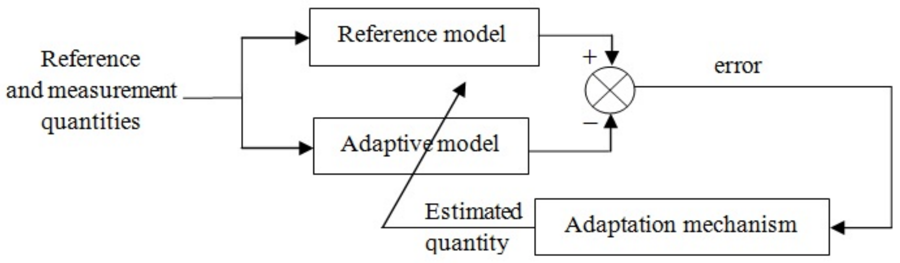


Figure 2.5: Diagram of the MRAS observer.

2.4.2 Stochastic observers

Stochastic observers give an optimal estimate of states based on stochastic criteria. Their observations are based on the presence of noise in the system. Among these observers, we cite the Kalman filter. This observer is characterized by taking into account measurement and state noises by stochastic algorithms tending to minimize the variance of the estimation error.

2.4.2.1 Kalman filter

The Kalman filter, introduced by *Rudolf Emil Kalman* in 1960, is one of the most interesting mathematical developments in linear estimation theory. It is a state reconstructor in a stochastic environment, when the variances of the noises are known, it is a linear estimator minimizing the variance of the estimation error. The applications of the Kalman filter are numerous. The Kalman filter makes it possible to give an estimate of the state of the system from a priori information on the evolution of this state (model) and real measurements. It will be used to estimate unknown initial conditions, predict trajectories, locate machines, implement control laws, etc [12].

2.4.2.2 Extended Kalman Filter (EKF)

The Kalman filter described previously is limited to linear systems. However, most physical systems are nonlinear. This nonlinearity can be associated with the process model, the observation model or both. The extended Kalman filter is a nonlinear extension of the conventional Kalman filter, which was developed specifically for systems with nonlinear dynamic models.

The EKF algorithm can also be decomposed into two phases: the prediction phase and the correction phase. We can schematize the extended Kalman filter by the following Figure 2.6:

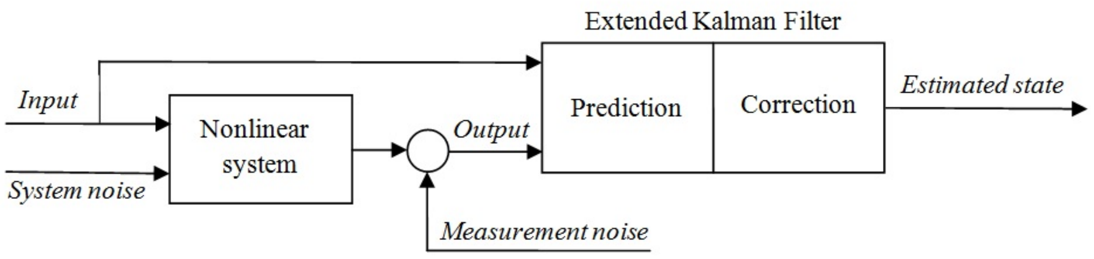


Figure 2.6: Block diagram of an Extended Kalman Filter.

2.5 Continuous-time high gain observer

2.5.1 Introduction

The high gain observer, initiated around the 1990s [13], has been proposed for nonlinear systems that can be put into the uniformly observable canonical form. Its advantage compared to the other observers previously developed is that it takes into account all the nonlinearities and nonstationarity of the systems. In addition, the nonlinearities can depend on the states, but must have a lower triangular structure.

The observer guarantees a good estimation of the states and the adjustment of the correction term is ensured with a single synthesis parameter.

2.5.2 Preliminary study

Definition 2.5.2.1. : ***Diffeomorphism***

A system is said to be diffeomorphic to another system if there is a differentiable bijective application from one set to another whose reciprocal bijection is also differentiable. This application allows to rewrite the system in a particular form.

Consider the MIMO systems and diffeomorphism to the following system:

$$\begin{cases} \dot{x}\left(t\right) = Ax(t) + \varphi\left(u(t), x(t)\right) \\ y\left(t\right) = Cx(t) = x^1(t) \end{cases} \tag{2.4}$$

With

$$x = \begin{pmatrix} x^1 \\ x^2 \\ \vdots \\ x^q \end{pmatrix} \quad ; \quad \varphi\left(u, x\right) = \begin{pmatrix} \varphi^1\left(u, x^1\right) \\ \varphi^2\left(u, x^1, x^2\right) \\ \vdots \\ \varphi^{q-1}\left(u, x^1, \dots, x^{q-1}\right) \\ \varphi^q\left(u, x^1, \dots, x^q\right) \end{pmatrix}$$

$$A = \begin{bmatrix} O_p & I_p & O_p & & O_p \\ \vdots & \ddots & I_p & & \\ O_p & & \ddots & \ddots & O_p \\ O_p & & & \ddots & I_p \\ O_p & \dots & & O_p & O_p \end{bmatrix} \quad ; \quad C = [I_p, O_p, \dots, O_p] \tag{2.5}$$

The state $x\left(t\right) \in \Re^n$; the $x^j \in \Re^p$, $j = \{1, \dots, q\}$ are state blocks ;
the input vector $u\left(t\right) \in U$ a compact subset of $\Re^s$;
$y \in \Re^p$ is the output available at all times t.
The system (2.4) is very special because the states x^j have all the same size p. The total dimension is $n = p \times q$. Referring to the work of [14], the system (2.4) is put into the *Brunovski*'s canonical form.

Definition 2.5.2.2. : ***Observability***

A system is said to be observable when all the states can be reconstructed from the knowledge of its inputs and outputs.

The concept of observability was introduced by *Kalman* for linear systems (see [15]).

Definition 2.5.2.3. *: : **Uniform observability***

A system is uniformly observable if it is observable for any input.

We note that the class of uniformly observable systems is the natural extension of the class of linear systems.

The observation problem for dynamical systems can be interpreted as a trajectory tracking problem. The condition of which to fulfill to ensure convergence is described by the following property:

Property 1: Convergence condition of an observer

For a system (see 2.4) whose state vector is $x(t)$, the essential property that must fill the observer (see 2.9) is:

$$\lim_{t\to\infty} \|\hat{x}(t) - x(t)\| = 0 \tag{2.6}$$

This property ensures an asymptotic convergence towards zero of the observation error.

2.5.3 Continuous-time high gain observer design

In [16] the author has proposed a high gain observer for a class of nonlinear systems having a triangular structure. This observer converges exponentially and guarantees the robustness of the estimations despite the presence of disturbances and measurement noise.

Consider the following class of multivariable (MIMO) nonlinear systems that are diffeomorphic and uniformly observable:

$$\begin{cases} \dot{x}\left(t\right) = Ax\left(t\right) + \varphi\left(u\left(t\right), x\left(t\right)\right) + \beta\varepsilon\left(t\right) \\ y\left(t\right) = Cx\left(t\right) + \omega\left(t\right) = x^{1}\left(t\right) + \omega\left(t\right) \end{cases} \tag{2.7}$$

with

$$x = \begin{pmatrix} x^1 \\ x^2 \\ \vdots \\ x^q \end{pmatrix} \quad ; \quad \varphi(u,x) = \begin{pmatrix} \varphi^1(u,x^1) \\ \varphi^2(u,x^1,x^2) \\ \vdots \\ \varphi^{q-1}(u,x^1,\dots,x^{q-1}) \\ \varphi^q(u,x^1,\dots,x^q) \end{pmatrix}$$
$$A = \begin{bmatrix} O_p & I_p & O_p & & O_p \\ \vdots & \ddots & I_p & & \\ O_p & & \ddots & \ddots & O_p \\ O_p & & & \ddots & I_p \\ O_p & \dots & & O_p & O_p \end{bmatrix} \quad ; \quad B = \begin{bmatrix} I_p \\ O_p \\ \vdots \\ O_p \end{bmatrix} \quad ; \quad C = [I_p, O_p, \dots, O_p] \tag{2.8}$$

$x^i \in \Re^p$ are the state variables for $i \in [1,q]$;

$u(t) \in U\, in\, \Re^m$ is the system input;

$y \in \Re^p$ is the system output;

$\omega(t)$ the measurement noise;

$\varepsilon : \Re^+ \rightarrow \Re^p$ is an unknown function describing the system uncertainties and may depend on the state, the input and uncertain parameters.

The associated continuous-time observer is proposed in [17] as follows:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + \varphi(u(t), \hat{x}(t)) - \theta\Delta_\theta^{-1} K (C\hat{x}(t) - y(t)) \tag{2.9}$$

where $\hat{x} = \begin{pmatrix} \hat{x}^1 \\ \hat{x}^2 \\ \vdots \\ \hat{x}^q \end{pmatrix} \in \Re^n$; $K = \begin{pmatrix} K^1 \\ K^2 \\ \vdots \\ K^q \end{pmatrix} \in \Re^n$ is the gain matrix chosen such that the matrix $\bar{A} \triangleq A - KC$ is Hurwitz, there exist a positive definite symmetric matrix P and a positive real μ such that:

$$P\bar{A} + \bar{A}^T P \leq -2\mu I_n \tag{2.10}$$

Δ_θ is a diagonal matrix defined as follows:

$$\Delta_\theta = diag(I_p, \frac{1}{\theta} I_p, \dots, \frac{1}{\theta^{(q-1)}} I_p)$$

$\theta \geq 1$ is a scalar design parameter which makes it possible to determine the speed of convergence of the estimates.

The synthesis of the proposed observer necessitates some of the following assumptions:

A1. The state $x(t)$ is bounded, i.e. there exists a compact set $\Omega \in \Re^n$ such that $\forall t \geq 0$, $x(t) \in \Omega$.

A2. The functions φ^i for $i \in [1, q]$ are Lipschitz with respect to x uniformly in u, i.e. $\forall \rho \succ 0$; $\exists L \succ 0$; $\forall u\, s.t.\ \|u\| \leq \rho$;
$\forall (x, \bar{x}) \in \Omega \times \Omega : \|\varphi^i(u, x) - \varphi^i(u, \bar{x})\| \leq L \|x - \bar{x}\|$.

A3. The unknown function ϵ is essentially bounded, i.e.
$\exists \delta_\varepsilon \succ 0;\ \sup\limits_{t \geq 0} Ess \|\varepsilon(t)\| \leq \delta_\varepsilon$.

A4. The noise signal ω is essentially bounded, i.e.
$\exists \delta_\omega \succ 0;\ \sup\limits_{t \geq 0} Ess \|\omega(t)\| \leq \delta_\omega$.

2.6 Continuous discrete-time high-gain observer

2.6.1 Introduction

The objective of this section consists to determine a state observer that guarantees the robustness of the estimates despite the discretization of the measurements and the presence of disturbances and uncertainties. We, therefore, seek to reconstruct all the states of the continuous system from the measurements available at sampling instants t_k. For the synthesis, we base on the elements of the theory of the continuous high gain observer and we use the approach of Lyapunov to prove the exponential convergence of the proposed observer.

Consider the MIMO uniformly observable and diffeomorphic nonlinear systems to the following system:

$$\begin{cases} \dot{x}(t) = Ax(t) + \varphi(u(t), x(t)) + \beta\varepsilon(t) \\ y(t_k) = Cx(t_k) + \omega(t_k) = x^1(t_k) + \omega(t_k) \end{cases} \tag{2.11}$$

where the state $x(t) \in \Re^n$, the input vector $u(t)$, the matrix A and the Lipschitzian function $\varphi(u(t), x(t))$, ω, ϵ are defined in (2.7), (2.8), and $y(t_k)$ is the output measured at time t_k, satisfying the following inequality:

$$0 \leq t_0 \prec \cdots \prec t_k \prec t_{k+1} \prec \cdots \quad avec \quad \lim_{k\to\infty} t_k = \infty \tag{2.12}$$

Define the interval between two measurement instants bounded by τ_m and τ_M:

$$0 < \tau_m \leq \tau_k = t_{k+1} - t_k \leq \tau_M, \qquad \forall k \geq 0 \tag{2.13}$$

2.6.2 Preliminary study

Definition 2.6.2.1.

- *τ_M denotes the maximum admissible value of the sampling period (2.13) for which the exponential convergence towards zero of the observation error is guaranteed.*
- *τ_m denotes the minimum value of the sampling period.*

$$0 < \tau_m \leq \tau_k = t_{k+1} - t_k \leq \tau_M$$

2.6.3 Continuous discrete-time high gain observer design

We add the following hypothesis on the boundedness of the noise samples $\omega(t_k)$:

A5. For all t_k, the samples $\omega(t_k)$ are bounded by δ_ω where δ_ω is the essential bound given by Assumption A4.

The dynamics of the continuous-discrete observer proposed for the class of systems (2.11) is written as follows [17]:

$$\begin{aligned} \dot{\hat{x}}(t) &= A\hat{x}(t) + \varphi(u(t), \hat{x}(t)) - \theta\Delta_\theta^{-1} K e^{-\theta k_1 (t-t_k)} \\ &\times (C\hat{x}(t_k) - y(t_k)), \quad t \in [t_k, \ t_{k+1}[\end{aligned} \tag{2.14}$$

where
$\hat{x} = \begin{pmatrix} x^1 \\ \vdots \\ x^q \end{pmatrix}$, $K = \begin{pmatrix} k_1 I_p \\ \vdots \\ k_q I_p \end{pmatrix}$ is the gain matrix where the k_i's, $i = 1, \ldots, q$ are chosen such that the matrix $\bar{A} \triangleq A - KC$ is Hurwitz.

Δ_θ is the diagonal matrix defined before with $\theta \geq 1$.

The observer contains an exponential function varying in time independently of the error which is updated only at the sampling instants t_k. For the convergence, if the maximum allowable value of the sampling period satisfies a certain condition then the observer converges exponentially to 0 [17].

2.7 Conclusion

In this chapter, we have presented the continuous-time high gain and the continuous discrete-time high gain observers for a class of uniformly observable MIMO nonlinear systems with the presence of disturbances and uncertainties. The gain of the observer will be determined by an optimization algorithm which will be mentioned in the next chapter.

Bibliography

[1] Kalman R.E. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering.* 82(1): 35–45, 1960. doi:10.1115/1.3662552

[2] Luenberger D. An introduction to observers. *IEEE Transactions on Automatic Control.* 16(6): 596-602, 1971. doi:10.1109/tac.1971.1099826

[3] Fridman L., Shtessel Y., Edwards C., and Yan X. Higher-order sliding-mode observer for state estimation and input reconstruction in nonlinear systems. *International Journal of Robust and Nonlinear Control.* 18(4-5): 399-412, 2008. doi:10.1002/rnc.1198

[4] Krener A.J., and Isidori A. Linearization by output injection and nonlinear observers. *Systems and Control Letters.* 3(1): 47-52, 1983. doi:10.1016/0167-6911(83)90037-3

[5] Busawon K., Farza M., and Hammouri H. A simple observer for a class of nonlinear systems. *Applied Mathematics Letters.* 11(3): 27-31, 1998. doi:10.1016/s0893-9659(98)00029-9

[6] Bastin G., and Gevers M. Stable adaptive observers for nonlinear time-varying systems. *IEEE Transactions on Automatic Control.* 33(7): 650-658, 1988. doi:10.1109/9.1273

[7] Marino R., Peresada S., and Valigi P. Adaptive input-output linearizing control of induction motors. *IEEE Transactions on Automatic Control.* 38(2): 208-221, 1993. doi:10.1109/9.250510

[8] Bornard G., and Hammouri H. A high gain observer for a class of uniformly observable systems. *Proceedings of the 30th IEEE Conference on Decision and Control.* 1494–1496, 1991. doi:10.1109/cdc.1991.261650

[9] Gauthier J., and Bornard G. Observability for any u(t) of a class of nonlinear systems. *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*. 910–915, 1980. doi:10.1109/cdc.1980.271933

[10] Barbot J., Boukhobza T., and Djemai M. Sliding mode observer for triangular input form. *Proceedings of 35th IEEE Conference on Decision and Control*. 2: 1489–1490, 1996. doi:10.1109/cdc.1996.572727

[11] Besançon G. 2007. *Nonlinear observers and applications*. Saint Martin d'Héres: Gipsa-lab.

[12] Chafaa K. *Structures d'identification et de commande des systemes non linéaire basees sur les techniques floues*. PhD thesis, University of Batna, 2006.

[13] Gauthier J.P., Hammouri H., and Othmanet S. A Simple Observer for Nonlinear Systems Applications to Bioreactors. *IEEE Transactions on Automatic Control*. 37(6): 875–880, 1992. doi:10.1109/9.256352

[14] Naghshtabrizi P, Hespanha J.P., and Teel A.R. Exponential Stability of Impulsive Systems with Application to Uncertain Sampled-Data Systems. *Systems and Control Letters*. 57(5): 378–385, 2008. doi:10.1016/j.sysconle.2007.10.009

[15] Kalman R. On the General Theory of Control Systems. *IRE Transactions on Automatic Control*. 4(3): 110–110, 1959. doi:10.1109/tac.1959.1104873

[16] Farza M., M'saad M., and Rossignol L. Observer design for a class of MIMO nonlinear systems. Automatica. *Automatica*. 40(1): 135–143, 2004. doi:10.1016/j.automatica.2003.08.008

[17] Bouraoui I., Farza M., Ménard T., Ben Abdennour R., M'saad M., and Mosrati H. Observer design for a class of uncertain nonlinear systems with sampled outputs—Application to the estimation of kinetic rates in bioreactors. *Automatica*. 55: 78–87, 2015. doi:10.1016/j.automatica.2015.02.036

CHAPTER 3

METAHEURISTIC OPTIMIZATION

Summary

3.1 Introduction

Metaheuristic algorithms have found many applications in different fields of applied mathematics, engineering and other sciences. A metaheuristic is an algorithm inspired by nature that contains a set of methods, which include evolutionary algorithms, to solve known problems as performance improvement. Metaheuristics are based of principles, which make possible the design of solution algorithms. Optimization is the act of obtaining the best result; which gives the maximum or minimum value of a function; under given circumstances.

This chapter provides a brief overview of metaheuristic optimization algorithms called GA, PSO and BBO.

3.2 Optimization

Optimization is a very important tool in engineering; it is the act of obtaining the best result under given circumstances such as design, construction or maintenance. Optimization is the organized search for such designs and operating modes. It determines the set of actions or elements that must be implemented to achieve optimized systems. In the simplest case, optimization seeks the maximum or minimum value of an objective function corresponding to variables defined in a feasible range or space. More generally, optimization is the search of the set of variables that produces the best values of one or more objective functions while complying with multiple constraints. A single-objective optimization model embodies several mathematical expressions including an objective function and constraints as follows:

$$Optimize\ \ f(X), \quad X = (x_1, x_2, \ldots, x_i, x_N) \tag{3.1}$$

subject to

$$g_j(X) \prec b_j, \quad j = 1, 2, \ldots, m \tag{3.2}$$

$$x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, N \tag{3.3}$$

where
$f(x)$ is the objective function;
X is a set of decision variables x_i that constitutes a possible solution to the optimization problem;
x_i is the i^{th} decision variable;
N is the number of decision variables that determines the dimension of the optimization problem;
g_j is the j^{th} constraint;
b_j is a constant of the j^{th} constraint;
m is the total number of constraints;
$x_i^{(L)}$ is the lower bound of the i^{th} decision variable;
and $x_i^{(U)}$ is the upper bound of the i^{th} decision variable.

The term optimisation refers to both minimisation and maximisation tasks. A task involving the maximisation of the function f is equivalent to the task of minimising $-f$

(see Figure 3.1), therefore the terms minimisation, maximisation and optimisation are used interchangeably.

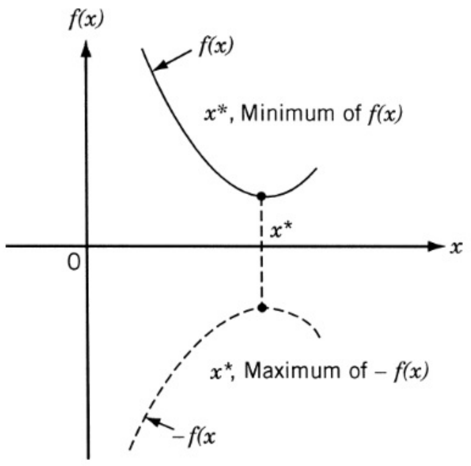


Figure 3.1: Minimum of $f(x)$ is same as maximum of $-f(x)$.

3.2.1 Stochastic optimization

Stochastic optimization using meta-heuristics is well adapted for solving problems for which it is difficult to find a global optimum or good local optima using classical methods. This type of optimization has three characteristics that are often decisive in global optimization:

- Optimization using metaheuristics does not require us to know the gradient of the function to be minimized, the only constraint being that we must be able to evaluate the latter, which can therefore have any form;
- It is not necessary to use a “good” initial point, the initialization being carried out at random in the search space;
- Finally, this type of optimization is stochastic, which makes it possible to overcome the combinatorial explosion of possibilities and limits trapping in the local optima.

3.2.2 Objective function

The objective function constitutes the goal of an optimization problem. That goal could be maximized or minimized by choosing variables, or decision variables for the set of parameters that satisfy all constraints is called a *feasible solution*. Feasible solutions with objective function values as good as the values of any other feasible solutions are called *optimal solutions* [1].

3.2.3 Decision variables

The decision variables determine the value of the objective function. In each optimization problem we search for the decision variables that yield the best value of the objective function or optimum.

3.2.4 Decision space

The set of decision variables that satisfy the constraints of an optimization problem is called the *feasible decision space*. In an N-dimensional problem, each possible solution is an N-vector variable with N elements. Each element of this vector is a decision variable. Optimization algorithms search for a point (i.e., a vector of decision variables) or points (i.e., more than one vector of decision variables) in the decision space that optimizes the objective function.

3.2.5 Local and global optima

It has been established that a well-defined optimization problem has a well defined decision space. Each point of the decision space defines a value of the objective function. A local optimum refers to a solution that has the best objective function in its neighborhood. In a one-dimensional optimization problem, a feasible decision variable X^* is a local optimum of a maximization problem if the following condition holds:

$$f\left(X^*\right) \geq f\left(X\right), \quad X^* - \varepsilon \leq X \leq X^* + \varepsilon \tag{3.4}$$

In a minimization problem the local optimum condition becomes

$$f\left(X^*\right) \leq f\left(X\right), \quad X^* - \varepsilon \leq X \leq X^* + \varepsilon \tag{3.5}$$

where

X^* is a local optimum;

ε is the limited length in the neighborhood about the local optimum X^*.

Figure 3.1 illustrates global and local optima for a one-dimensional maximization problem.

L_1, L_2, and L_3 in Figure 3.2 are local optima, and G denotes the global optimum with the largest value of the objective function.

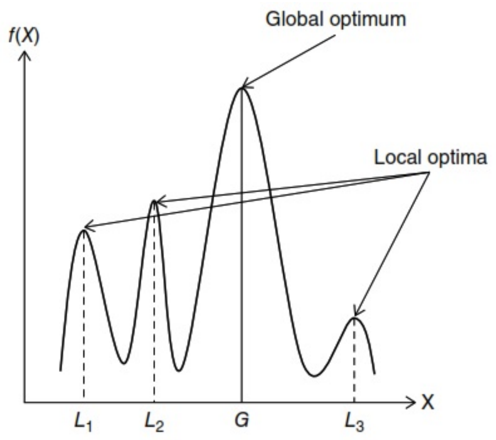


Figure 3.2: Schematic of global and local optimums in a one-dimensional maximizing optimization problem.

3.3 Metaheuristic algorithms

3.3.1 Definition of metaheuristics and algorithms

The words *meta* and *heuristic* both have their origin in the old Greek: meta means upper level, and heuristic denotes the art of discovering new strategies [2]. The term metaheuristic was coined by *Glover* in 1986 [3] to refer to a set of methodologies conceptually ranked above heuristics in the sense that they guide the design of heuristics. A metaheuristic is a higher level procedure or heuristic designed to find, generate, or select a lower level procedure or heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem. By searching over a large set of feasible solutions, metaheuristics can often find good solutions with less computational effort than calculus-based methods, or simple heuristics.

An *algorithm* refers to a sequence of operations that are performed to solve a problem. Algorithms are made of iterative operations or steps that are terminated when a stated convergence criterion is reached. Each step may be refined into more refined detail in terms of simple operations. Figure 3.3 shows a general schematic of an algorithm [4].

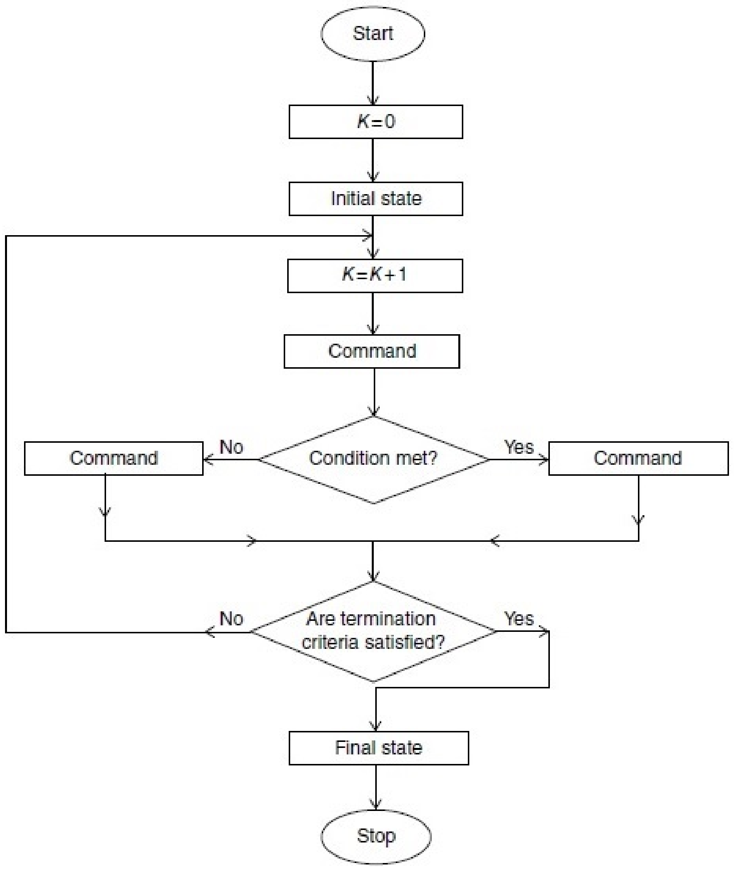


Figure 3.3: General schematic of a simple algorithm; K denotes the counter of iterations.

3.3.2 Modern metaheuristics of optimization

In recent years, some optimization methods that are conceptually different from the traditional mathematical programming techniques have been developed. These methods are labeled as modern or nontraditional methods of optimization. Most of these methods are based on certain characteristics and behavior of biological. The following metaheuristics are described in this part:

- Genetic algorithms
- Particle swarm optimization
- Biogeography-Based Optimization

3.3.2.1 Genetic algorithm

Genetic algorithm (GA) was invented by *J. Holland* and developed this idea in his book "*Adaptation in natural and artificial systems*" in 1975 [5]. Also, *John Holland* introduced the term *genetic algorithm*. Thus, a genetic algorithm is a technique for simulating the natural process of microscopic evolution and adaptation specific to biological systems [6]. He described how to apply the principles of natural evolution to optimization problems and built the first Genetic Algorithms. *Holland*'s theory has been further developed and now Genetic Algorithms stand up as a powerful tool for solving search and optimization problems. Genetic algorithms are based on the principle of genetics and evolution.

GA operates with a collection of chromosomes, called a population of individuals (where each individual in the population represents a candidate solution to the optimization problem). The population is normally randomly initialized. As the search evolves, the population includes fitter and fitter solutions, and eventually, it converges, meaning that it is dominated by a single solution. *Holland* also presented proof of convergence to the global optimum where chromosomes are binary vectors. In the most general case, the fitness of an individual determines the probability of its survival for the next generation [7].

GA uses three (genetic) operations to generate new solutions from existing ones : *crossover*, *mutation* and *inversion*.

- *Crossover* is a genetic operation that can be described as exchanging two chromosomes, called parents, together to form new chromosomes, called offspring.
- *Mutation* is a random change into characteristics of chromosomes. It's generally applied at the gene level.
- *Inversion* is a genetic operation that produces a change in the concatenation of the genes in a certain area of chromosome, so that the new gene sequence (series) is inverted with respect to the initial sequence.

The genetic algorithm loops are an iteration process to make the population evolve. Each consists of the following steps:

- *Selection* The first step consists in selecting individuals for reproduction. This selection is done randomly with a probability depending on the relative fitness of the individuals so that best ones are often chosen for reproduction than poor ones.
- *Reproduction* In the second step, offspring is bred by the selected individuals. For generating new chromosomes, the algorithm can use both recombination and mutations.
- *Evaluation* Then the fitness of the new chromosomes is evaluated.
- *Replacement* During the last step, individuals from the old population are killed and replaced by the new ones.

Figure 3.4 shows the simplified iterative operation of a genetic algorithm that works through a simple cycle of steps [8]:

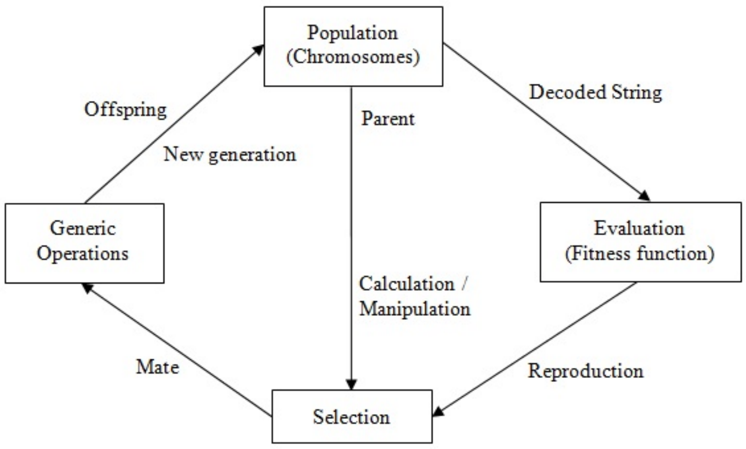


Figure 3.4: General operation of a GA.

3.3.2.2 Particle swarm optimization algorithm

Particle swarm optimization (PSO) is an evolutionary computation technique inspired by social behavior of groups like bird flocking (Figure 3.5), fish schooling (Figure 3.6) or colonies of insects (Figure 3.7); because it is known that a group can effectively achieve an objective by using the common information of every element. PSO algorithm was first introduced in 1995 by *Eberhart* and *Kennedy* [9, 10] as an alternative to population based search approaches (like genetic algorithms) in order to solve optimization problems.

Figure 3.5: A flock of birds.

Figure 3.6: A school of fish.

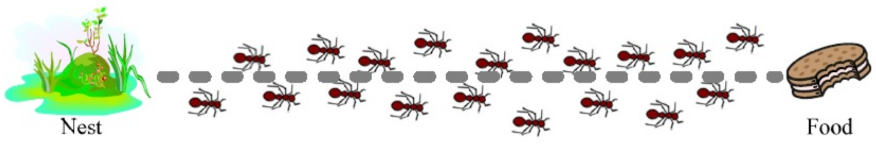


Figure 3.7: The path of the ants.

In this algorithm the elements of the population are called particles, and each particle (a bird or a fish) is a candidate for the solution. Each particle is considered as a moving point in the N-dimensional search space with a certain velocity. The velocity of each particle is constantly adjusted according to its own experience and the experience of its companions hopping to fly towards better solution area.The displacement of a particle is influenced by three components (Figure 3.8) [11]:

- *A physical component* the particle tends to follow its current direction of displacement;
- *A cognitive component* the particle tends to move towards the best site by which it has already passed;
- *A social component* the particle tends to rely on the experience of its congeners and, thus, to move towards the best site already reached by its neighbors.

In PSO, each state of particle presents a position and velocity, which is initialized with a population generation by a random process. Note that each particle is described by three features:

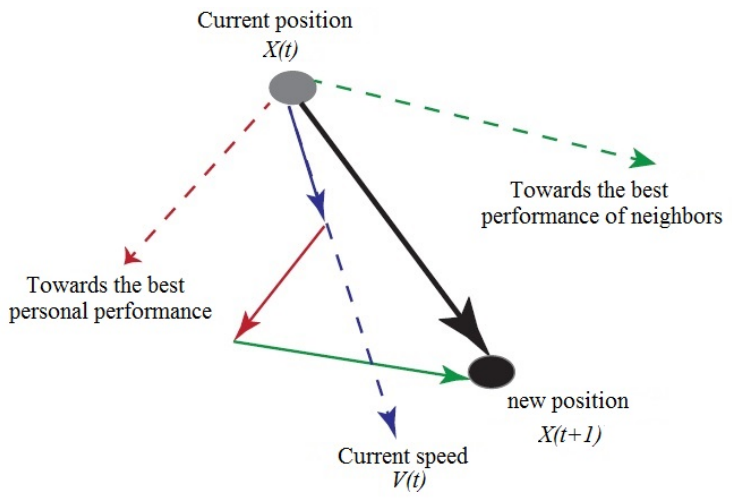


Figure 3.8: Displacement of a particle.

x_k^i: i^{th} particle vector position at time k;

v_k^i: i^{th} particle velocity at time k, which represents the search direction and used to update the position vector;

$f\left(x_k^i\right)$: fitness or objective, determines the best position of each particle over time.

Mathematically, the particle velocities are updates according to the following equations:

$$v_{k+1}^i = wv_k^i + c_1 rand(p^i - x_k^i) + c_2 rand(p_k^g - x_k^i) \tag{3.6}$$

where

v_{k+1}^i: the new velocity;

w: the inertia factor;

c_1: positive constant (self confidence);

c_2: positive constant (swarm confidence);

g: represents the index of best particle among all the particles in the population;

p^i i^{th}: particle best position (the best position in the swarm);

p_k^g: particle best global position (best particle among all the particles in the population) until time k (so, p_g will be the last best global position);

$rand$: is a random number uniformly distributed in $[0, 1]$.

Particle positions are the updates by velocity (3.6) as

$$x_{k+1}^i = x_k^i + v_{k+1}^i \tag{3.7}$$

The PSO principle consists of, at each time step, regulating the velocity and location of each particle toward its p_i and p^g locations according to equations (3.6) and (3.7) until a maximum change in the fitness function will be smaller than a specified tolerance ε which gives us the following stopping criteria (3.8):

$$\left| f\left(p^g_{k+1}\right) - f\left(p^g_k\right) \right| \leq \varepsilon \tag{3.8}$$

The PSO algorithm flowchart is shown in Figure 3.9.

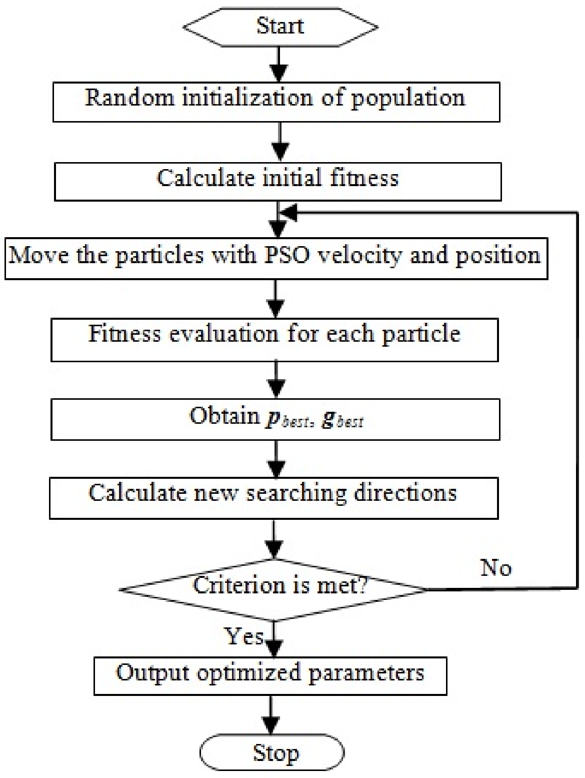

Start
Random initialization of population
Calculate initial fitness
Move the particles with PSO velocity and position
Fitness evaluation for each particle
Obtain p_{best}, g_{best}
Calculate new searching directions
Criterion is met?
No
Yes
Output optimized parameters
Stop

Figure 3.9: PSO algorithm flowchart.

3.3.2.3 Biogeography-based optimization

Biogeography-Based Optimization technique (BBO) is a novel biological optimization technique and one of the metaheuristic algorithms which simulates the biogeography of nearby islands. Each island has a high suitability index (HSI) which determines the number of species (S_i) that will be able to live there. Mathematical models of BBO describe how species migrate from one island (habitat) to another, how new species arise and how species become extinct. It is inspired by mathematical models of biogeography and the first original was introduced by *Dan Simon* in 2008 [12].

A good habitat has a high HSI, while a poor habitat has a low HSI. This means that good habitats have more good aspects than the poor ones. Habitats with high HSI have a high immigration rate due to their good aspects, whereas poor habitats have a low immigration rate but a high emigration rate, unlike good ones. The migration rates are directly related to the number of species in a habitat. So, a habitat with many species has a high emigration rate, because it is almost saturated, while habitats with few species have high immigration rate because do not have good conditions to live in. This migration process increases the diversity of the habitat and the miscegenation and contributes to the species information sharing and the mutation probability. Figure 3.10 represents emigration and immigration as a function of the number of species. In Figure 3.10, I and E represent the maximum rates of immigration and emigration, respectively, and S denotes the number of species [13].

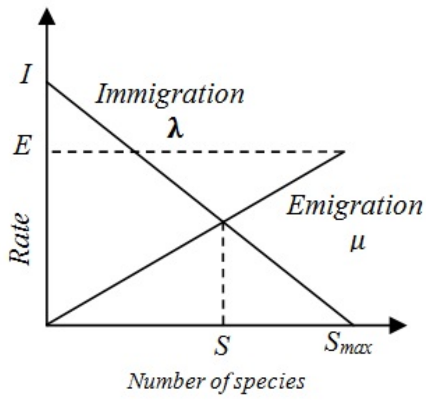


Figure 3.10: Emigration and immigration rates.

- ***BBO algorithm***

  The basic algorithm of BBO is as follows:

  – *Step 1:* Initialize the parameters used in the algorithm: S_{max} maximum number of species, E emigration rate, I the immigration rate, and m_{max} the maximum mutation rate.

  – *Step 2:* Calculate the probability for each value of the number of species as follows:

$$P_j = \frac{1}{S_{\max}} \tag{3.9}$$

where

$j = 1, \ldots, S_{\max}$ and P is the probability for the j^{th} habitat.

- *Step 3:* Generate an initial random set of habitats according to the constraints of the problem.
- *Step 4:* Start the loop:
  - *(4.i)* Generate the immigration and emigration rates:

$$\lambda_j = I\left(1 - \frac{j}{S_{\max}}\right) \tag{3.10}$$

$$\mu_j = E\frac{j}{S_{\max}} \tag{3.11}$$

where

λ_j and μ_j are the immigration and the emigration rates for the j^{th} habitat.

  - *(4.ii)* Calculate the derivative probability:

$$\begin{cases} \overset{\bullet}{\underset{s}{P}} = -\left(\lambda_s + \mu_s\right) P_s + \mu_{s+1} P_{s+1} & s = 0 \\ \overset{\bullet}{\underset{s}{P}} = -\left(\lambda_s + \mu_s\right) P_s + \lambda_{s-1} P_{s-1} + \mu_{s+1} P_{s+1} & 1 \leq s \prec S_{\max} \\ \overset{\bullet}{\underset{s}{P}} = -\left(\lambda_s + \mu_s\right) P_s + \lambda_{s-1} P_{s-1} & s = S_{\max} \end{cases} \tag{3.12}$$

  - *(4.iii)* Update the probability:

$$P_j = P_j + \overset{\bullet}{P_j}\, dt \tag{3.13}$$

$$P_j = \frac{P_j}{\sum_{i=0}^{S_{\max}} P_i} \tag{3.14}$$

where

dt is the derivative step.

  - *(4.iv)* Use the immigration and emigration rates to modify each habitat and probabilistically mutate the individuals.
  - *(4.v)* Evaluate the habitats to make sure that the constraints of the problem are satisfied.
  - *(4.vi)* Calculate the fitness of each habitat and return to the beginning of the loop until a stopping criterion is achieved.

The BBO algorithm flowchart is shown in Figure 3.11.

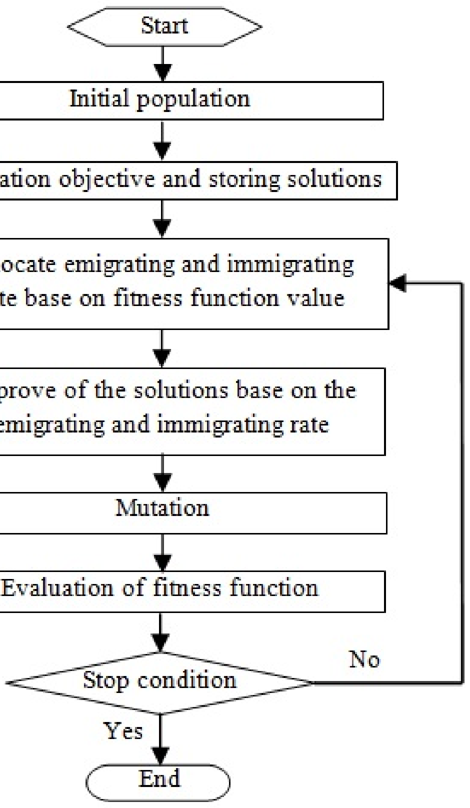

Start
Initial population
Evaluation objective and storing solutions
Allocate emigrating and immigrating rate base on fitness function value
Improve of the solutions base on the emigrating and immigrating rate
Mutation
Evaluation of fitness function
Stop condition
No
Yes
End

Figure 3.11: BBO algorithm flowchart.

3.4 Conclusion

This chapter is intended to provide an overview of metaheuristic algorithms and the theory of optimization methods that will be used later, namely genetic algorithm, particle swarm optimization and biogeography-based optimization.

Bibliography

[1] Rardin R.L. (1998). *Optimization in operations research.* Upper Saddle River, NJ: Prentice Hall.

[2] Talbi E. (2009). *Metaheuristics: From design to implementation.* Oxford: Wiley.

[3] Glover F. Future paths for integer programming and links to artificial intelligence.*Computers and Operations Research.* 13(5): 533-549, 1986. doi:10.1016/0305-0548(86)90048-1

[4] Bozorg-Haddad O., Solgi M., and Loaiciga H.A. (2017). *Meta-heuristic and evolutionary algorithms for engineering optimization.* Hoboken, NJ: John Wiley and Sons.

[5] Holland J.H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* Ann Arbor: University of Michigan Press.

[6] Stefanoiu D., Borne P., Popescu D., Filip, F.G., and Kamel, A.E. (2014). *Optimization in engineering sciences: Metaheuristics, stochastic methods and decision support.* London: ISTE.

[7] Kumar S., gaur P. Comparative Response of FIR Filter Using GA PSO BBO. *International Journal of Advanced Research in Computer and Communication Engineering.* 4(4): 83–86, 2015.

[8] Sivanandam S.N., Deepa S.N. (2008). *Introduction to genetic algorithms.* Berlin: Springer.

[9] Kennedy J., and Eberhart R. Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks.* 1942-1948. 1995. doi:10.1109/icnn.1995.488968

[10] Eberhart R., Kennedy J. A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science.* 39-43. 1995. doi:10.1109/mhs.1995.494215

[11] Ouali M.A. *Modélisation, Débruitage, Extraction des caractéristiques et Classification des signaux électrocardiogrammes ECG.* PhD thesis, University of Batna-2, 2018.

[12] Simon D. Biogeography-Based Optimization. *IEEE Transactions on Evolutionary Computation.* 12(6): 702-713, 2008.

[13] Silva M.A.C., Silveira C.C., and Coelho L.S. A New Biogeography-Based Optimization Approach Based on Shannon-Wiener Diversity Index to PID Tuning in Multivariable System. *ABCM Symposium Series in Mechatronics.* 5: 591-601, 2012.

CHAPTER 4

METAHEURISTIC OPTIMIZATION OF HIGH GAIN OBSERVER

Summary

4.1 Introduction

The high gain observer is one of the most important observers in the literature. It has been used extensively in the design of output feedback control of nonlinear systems, which is responsible for estimating the system states. It has several advantages, including: stability and robustness against modeling errors and external disturbances.

This chapter presents the simulation results of optimization methods that are applied on nonlinear dynamical system (the two link robot) to estimate the observer's high gain.

4.2 Learning observer parameters and control

To apply high gain observer, its parameters have to be determined which is a difficult task especially that the stochastic properties of the corresponding noises disturbing the robot are unknown. To avoid this difficulty, the high gain value will be considered as a free parameter to be tuned. In the literature, this parameter was determined manually by a constrained free choice. In this thesis, we propose to do a tuning for these parameters. The simplest tuning can be released by trial and error method which is a very laborious task. To surmount this problem and to avoid trial and error, we propose to tune these parameters by BBO algorithm. The framework of the method is constituted of two steps.

1) In the first step represented in Figure 4.1, we present a BBO-HG structure which works in an offline manner and allows finding the optimal value of the gain.

2) In the second step, obtained parameter value from step 1 is injected into the estimation-control loop running online to control the two link robot.

The framework of the BBO-HG parameter estimation system is illustrated in Figure 4.1. The system input $u = [\tau_1, \tau_2]$ and the measured response $y = [\theta_1, \theta_2]$ are used by the high gain observer, where input u is applied to both two link robot and high gain observer. Actual (measured) angles of the robot and estimated angles of HG observer are set to be inputs to a performance evaluator through a comparator. Note that the optimization will be impossible if the angles cannot be measured. The performance evaluator calculates the fitness function which is a mean square error (MSE) criterion between y and $\hat{y}$. Then, obtained MSE will be applied to the BBO optimizer. Based on MSE values, BBO optimizer will calculate and optimize the unknown parameters gain observer by updating the solutions according to BBO algorithm to provide better sets. The new solutions are then used for the adaptation of the HG observer for the next iteration until a preset number of iterations have been reached, and then optimal values of the gain are obtained. Finally, optimized values are injected into HG observer running online to estimate the robot states. Note that the BBO-HG algorithm is implemented offline because BBO needs several iterations to obtain acceptable solutions. For each iteration, BBO-HG estimator has to be executed once; consequently, the BBO-HG must be executed several times allowing the optimization of the parameters from each measurement.

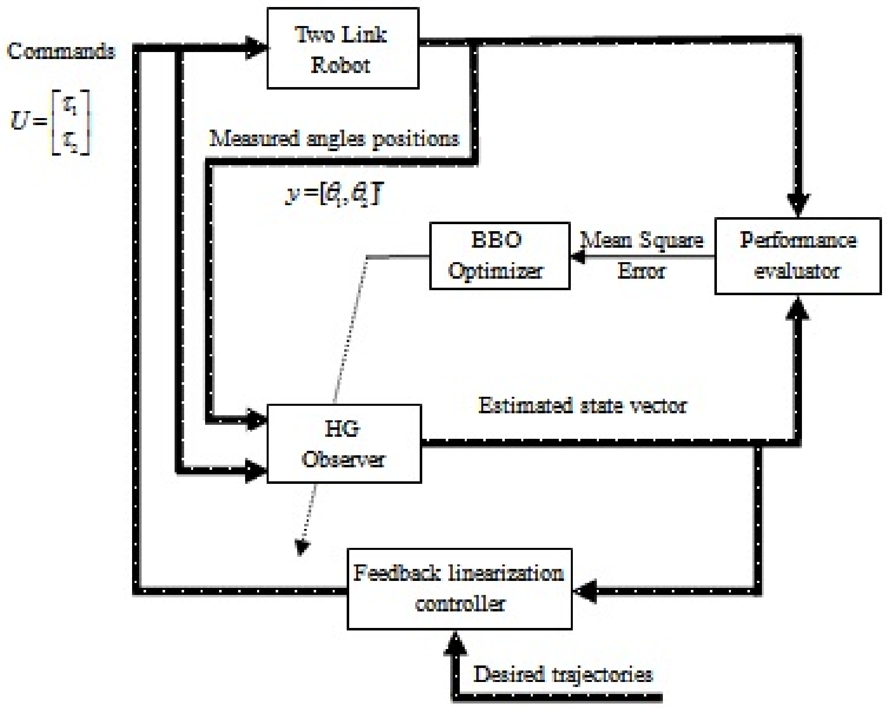

Commands
$U=\begin{bmatrix}\tau_1\\ \tau_2\end{bmatrix}$
Two Link Robot
Measured angles positions
$y=[\theta_1,\theta_2]^T$
BBO Optimizer
Mean Square Error
Performance evaluator
Estimated state vector
HG Observer
Feedback linearization controller
Desired trajectories

Figure 4.1: Framework of the proposed state observer optimization with feedback linearization.

4.2.1 Controller design

For control purposes, system (2.11) in chapter 2 can be rewritten as an n^{th} order nonlinear dynamical system represented in the controllable canonical form:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \vdots \\ \dot{x}_n = f(x_1, x_2, ..., x_n) + g(x_1, x_2, ..., x_n)u \\ y = x_1 \end{cases} \tag{4.1}$$

or, equivalently

$$\begin{cases} x^{(n)} = f(x_1, x_2, ..., x_n) + g(x_1, x_2, ..., x_n)u \\ y = x \end{cases} \tag{4.2}$$

where f and g are real continuous functions, $u \in \Re$ and $y \in \Re$ are the input and output of the system, respectively. We assume that the state vector $x = (x_1, x_2, ..., x_n)^T = (x, \dot{x}, ..., x^{(n-1)})^T \in \Re^n$ is not available for measurement. The controllability of (4.2) requires that $g(x) \neq 0$ for all $\underline{x}$ in certain controllability region $U_c \subset \Re^n$. Since $g(x)$ is continuous, without loss of generality, we assume that $0 < g(x) < \infty$ for all $x \in U_c$. In addition, we assume that the functions f and g are bounded. The control objective is to find a feedback control law $u = u(x)$ such that to make the state $\underline{x}(t)$ track a given desired bounded reference trajectory $\underline{y}_m(t) = \left(y_m(t), \dot{y}_m(t), ..., y_m^{(n-1)}(t)\right)^T$. But since $x(t)$ is not available, it will be replaced by its estimate $\hat{x}(t)$. Therefore, the control law becomes $u = u(\hat{x})$, where $\hat{x}(t)$ is the state estimate provided by high gain observer.
System of the form (4.2) can be then controlled by the so-called feedback linearization method [1, 2]. In this method, $f(\hat{x}(t))$ and $g(\hat{x}(t))$ are used to construct the following feedback controller:

$$u = u(\hat{x}) = \frac{1}{g(\hat{x})}\left[-f(\hat{x}) + y_m^{(n)}(t) + \underline{k}^T\underline{e}\right] \tag{4.3}$$

where $e = e(t) \triangleq y_m(t) - y(t)$ is the tracking error $\underline{e} = \underline{e}(t) \triangleq (e, \dot{e}, ..., e^{(n-1)})^T$, and $\underline{k} \triangleq (k_n, ..., k_2, k_1)^T \in \Re^n$ is chosen such that all roots of the polynomial $h(s) \triangleq s^n + k_1 s^{n-1} + ... + k_n$ are in the open left-half of the complex plane. Applying the control law (4.3) to the system (4.2) we obtain the following error dynamics

$$e^{(n)} + k_1 e^{(n-1)} + ... + k_n e = 0 \tag{4.4}$$

where the main objective of the control is $\lim\limits_{t\to\infty} e(t) = 0$. However, construction of estimation $\underline{x}(t)$ by the high gain observer can gives us good values for $f(\hat{x})$ and $g(\hat{x})$ which will allow the construction of the control law (4.3).

4.3 Experimentations and simulation results

Throughout this section, experimental simulations are performed on an $Intel® Core^{TM}\ i7-7500U\ CPU@2.70GHz\ 2.90GHz$ under $Matlab\ R2018b$ environment. Note that all of our codes are written in Matlab language in M-files with time step size $2.5e{-}04s$.

The effectiveness of the proposed method is tested on a highly nonlinear dynamical system: The two link robot. This system is naturally unstable and has to be persistently balanced by control actions to hold it in stable positions. The control is guaranteed by a feedback linearization control technique.

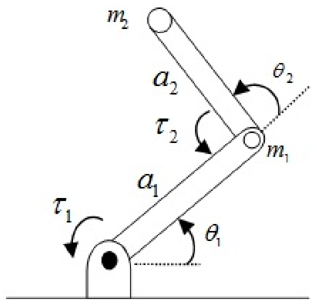


Figure 4.2: Two-link planar RR arm.

The used two link robot manipulator is shown in Figure 4.2. Its dynamic is given by the following differential equation [3, 4]:

$$H(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + g(q) = \tau \tag{4.5}$$

Where $H(q) = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}$ is symmetric, positive definite mass matrix such that

$h_{11} = (m_1 + m_2)\,a_1^2 + m_2 a_2^2 + 2m_2 a_1 a_2 + \cos\theta_2$

$h_{12} = h_{21} = m_2 a_2^2 + m_2 a_1 a_2 \cos\theta_2$

$h_{22} = m_2 a_2^2$

The parameters m_i and a_i, $i = 1, 2$ are masses and lengths of the robot taken $m_1 = m_2 = 1\,kg$ and $a_1 = a_2 = 1\,m$.

$\tau = \begin{bmatrix} \tau_1 & \tau_2 \end{bmatrix}^T$ is the vector of joint torques supplied by the actuators;

$q = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T$ is the vector of joint displacements;

$C(q,\,\dot{q}) = \begin{bmatrix} -2m_2 a_1 a_2 \dot{\theta}_2 \sin\theta_2 & -m_2 a_1 a_2 \dot{\theta}_2 \sin\theta_2 \\ m_2 a_1 a_2 \dot{\theta}_1 \sin\theta_2 & 0 \end{bmatrix}$ is the Centrifugal and Coriolis forces matrix;

$g(q) = \begin{bmatrix} (m_1 + m_2)\,g a_1 \theta_1 + m_2 g a_2 \cos(\theta_1 + \theta_2) \\ m_2 g a_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$ is the gravitational forces matrix with $g = 9.8\,m/s^2$ is the acceleration due to gravity.

The state variables are chosen to be: $x_1 = \theta_1$, $x_2 = \dot{\theta}_1$, $x_3 = \theta_2$ and $x_4 = \dot{\theta}_2$

Parameters of equation (2.11) in chapter 2 are defined as follows:

$$x^1 = \begin{pmatrix} x_1 \\ x_3 \end{pmatrix}, x^2 = \begin{pmatrix} x_2 \\ x_4 \end{pmatrix}, \varphi^2 = \begin{pmatrix} \varphi_2\left(u, x^1(t), x^2(t)\right) \\ \varphi_4\left(u, x^1(t), x^2(t)\right) \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix};$$

$$\varphi^2 = \begin{pmatrix} \varphi_2\left(u, x^1(t), x^2(t)\right) \\ \varphi_4\left(u, x^1(t), x^2(t)\right) \end{pmatrix} = M^{-1}\left[-C\dot{q} - G\right] + M^{-1}\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix};$$

$$M(q) = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}$$

with

$$m_{11} = (m_1 + m_2)\,a_1^2 + m_2 a_2^2 + 2m_2 a_1 a_2 + \cos x_3$$

$$m_{12} = m_{21} = m_2 a_2^2 + m_2 a_1 a_2 \cos x_3$$

$$m_{22} = m_2 a_2^2$$

In order of the matrix $A - KC$ to take all its eigenvalues -1; we determined the gain of the observers (2.14) in chapter 2 as $K^1 = 2I_2$ and $K^2 = I_2$.

Initial conditions have been initialized as follows:

- Robot state initial conditions: $x_1(0) = x_2(0) = x_3(0) = x_4(0) = 0\ rad$.
- Estimator initial conditions: $\hat{x}_1(0) = 0.1\ rad$, $\hat{x}_2(0) = 0\ rad$, $\hat{x}_3(0) = 0.2\ rad$ and $\hat{x}_4(0) = 0\ rad$.

Reference input trajectories are supposed to be a sinusoidal with amplitude $0.1\ rad$ for the first joint and a step input with amplitude $0.3\ rad$ for the second joint as shown in Figure 4.3.

Two cases of experimentations are treated:

1. without stochastic perturbations (noise free case).
2. with stochastic perturbations (noisy output case).

In our experimentations, the two-link robot is controlled by a feedback linearization controller for which the state vector is estimated by a high gain observer. High gain observer parameter is optimized under the two cited environments. Note that optimization process is done during an off-line phase.

We give in Figures 4.4 and 4.5 simulation results of estimation errors that show the influence of the observer gain and the nature of the observer (continuous time or continuous-discrete time). In Figure 4.4, we see that observation errors are less in the

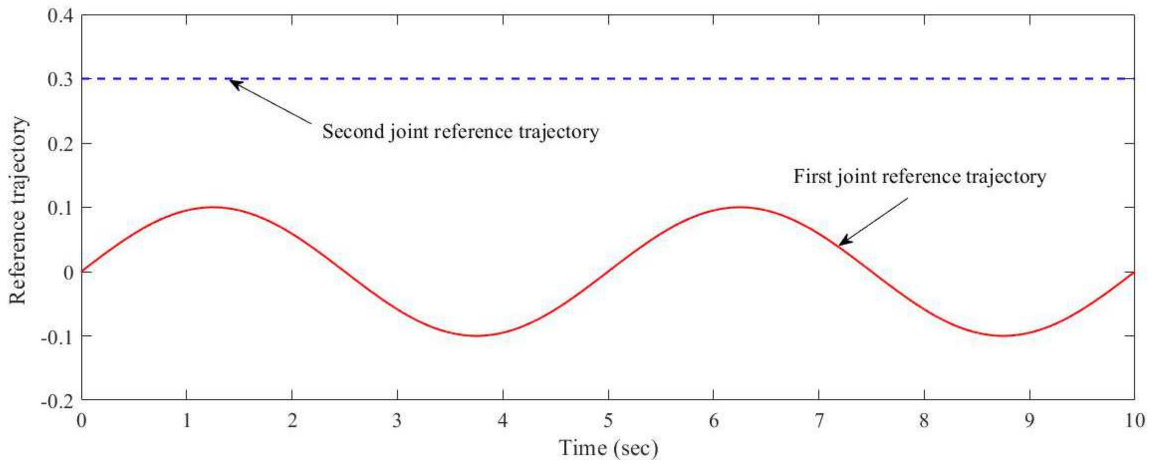


Figure 4.3: Input reference trajectories.

case when the gain is high (estimation errors decrease when θ increases). Likewise, in Figure 4.5 we notice that the obtained errors are smaller in the case of continuous observer (estimation errors provided by the continuous-discrete time observer are bigger than those of continuous-time observer). By this comparison, we confirm that the continuous-discrete time observer will be equivalent to the continuous-time observer when the sampling period tends to zero. The same fact is obtained with the second error e_2 of the second articulation.

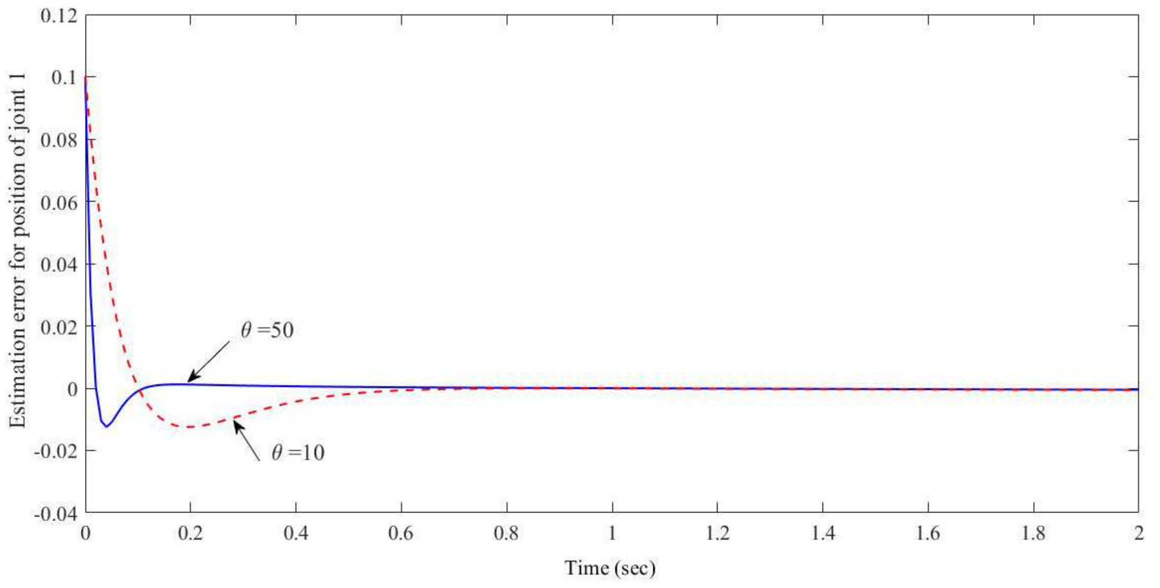


Figure 4.4: Influence of the observer gain on the estimation error e_1 of the first joint.

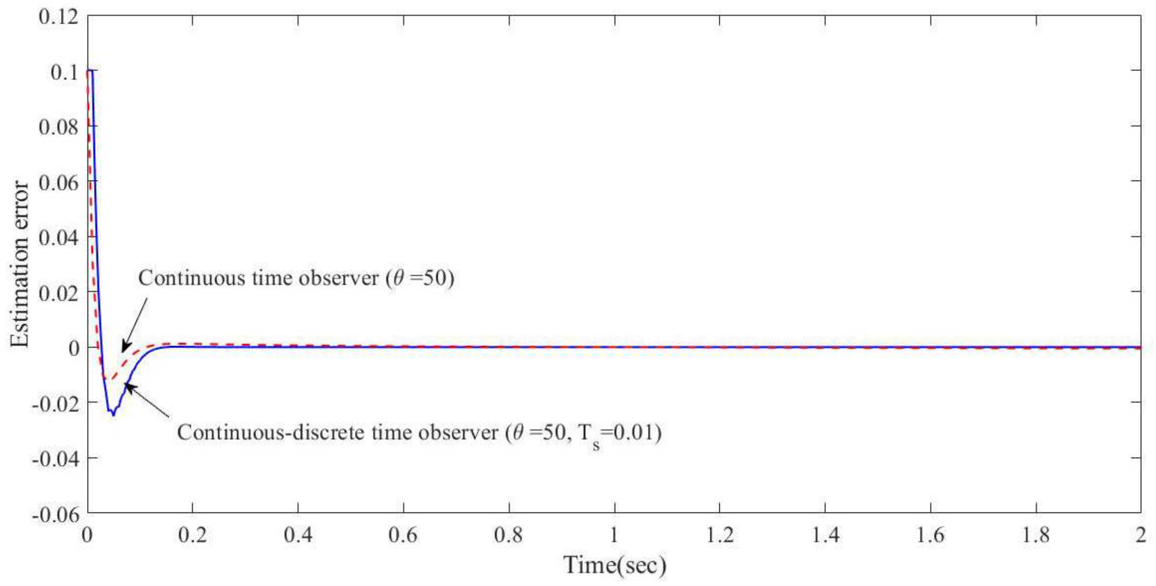


Figure 4.5: Superposition of estimation errors e_1 with continuous time and continuous-discrete time observers.

A. The noise-free case

Note that the observer gain value can be chosen arbitrary (by trial and error method) in its solution region. In this situation we say that the high gain θ is un-optimized. The key problem of this choice is that it affects greatly the estimation result. The un-optimized high gain θ obtained by trial and error is posted in Table 4.1.

Table 4.1: Un-optimized high gain for robot manipulator for noise free case.

Sampling period (T_s)	$\theta_{un-optimized}$	MSE
0.01	10	8.8129e-03
	50	4.0549e-03

Table 4.1 shows chosen parameters values with their resultant Mean Square Error, obtained by a trial and error method. The manual choice is easy to perform, but the method takes longtime to find good values. Therefore, to obtain reasonable estimation, an experienced expert has to do a great effort because it is difficult to infer a correlation between the values of the chosen parameter and the best state estimation. We denote good MSE performance for the second choice in the table. Notice that if these parameters are poorly chosen, this can cause big errors in the estimation. The corresponding results

of joint positions for joint 1 and 2 are presented in Figures 4.6 and 4.7, and their respective errors are shown in Figures 4.8 and 4.9.

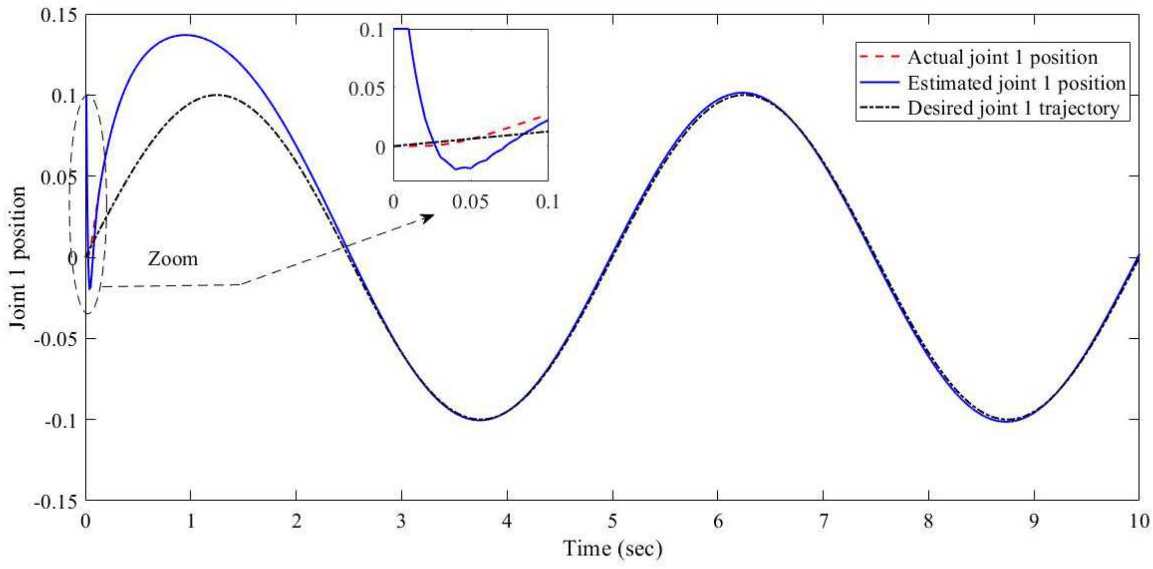


Figure 4.6: Tracking position of the first joint for sampled outputs with gain $\theta = 50$ and sampling time $T_s = 0.01$ sec.

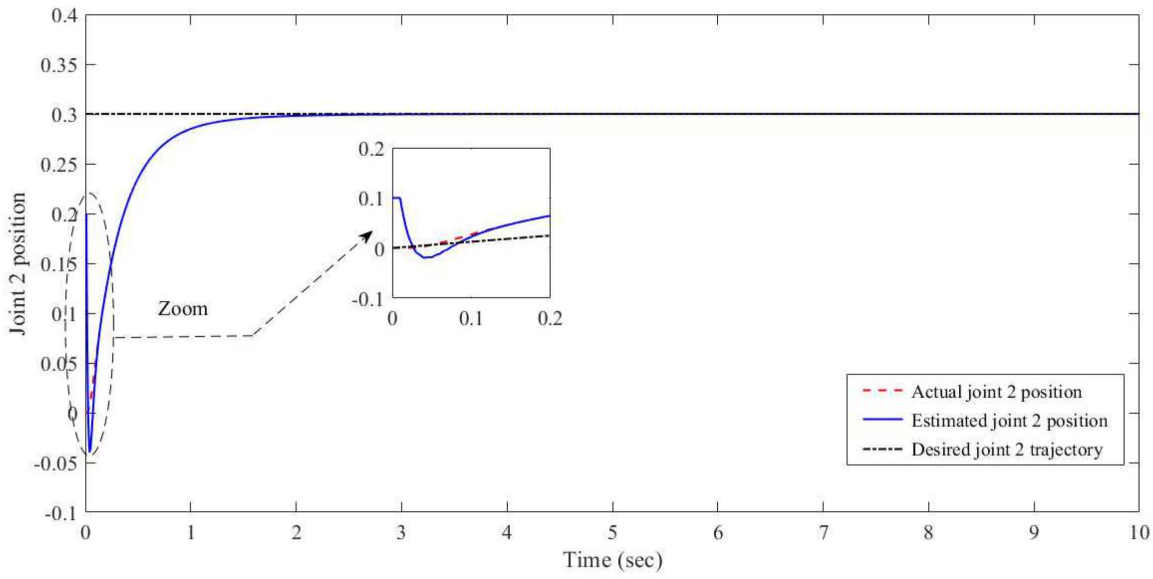


Figure 4.7: Tracking position of the second joint for sampled outputs with gain $\theta = 50$ and sampling time $T_s = 0.01$ sec.

In what follows, we will consider the optimization of the free parameter of the observer. The corresponding optimized high gain θ obtained by algorithms BBO, GA and PSO are presented in Table 4.2. The sampling period was chosen to be $T_s = 0.01$ sec.

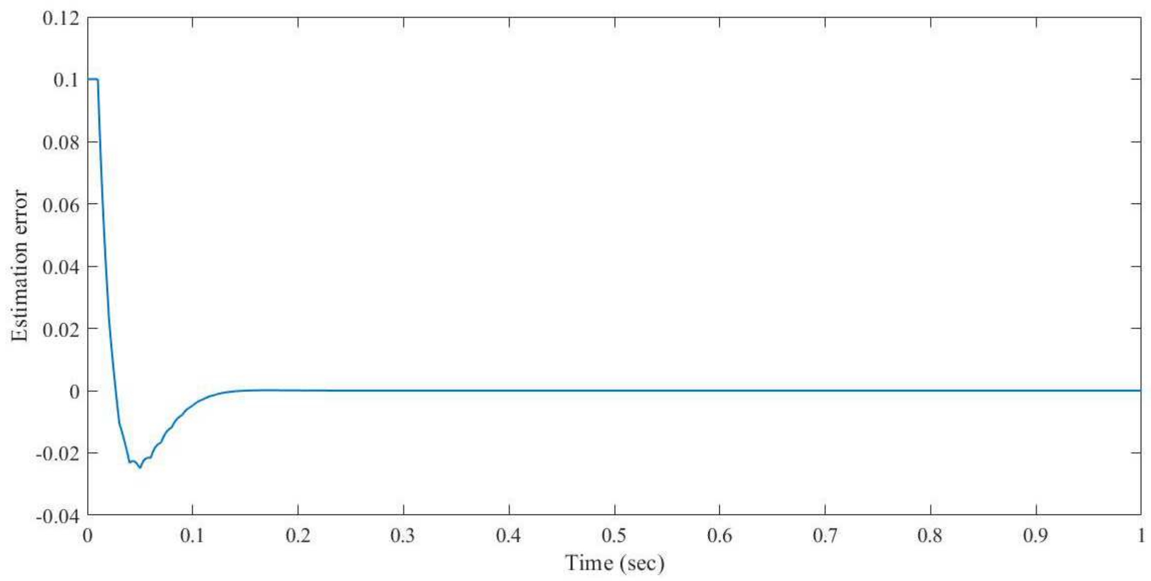


Figure 4.8: Estimation error e_1 of the first joint for sampled outputs with gain $\theta = 50$ and sampling time $T_s = 0.01$ sec.

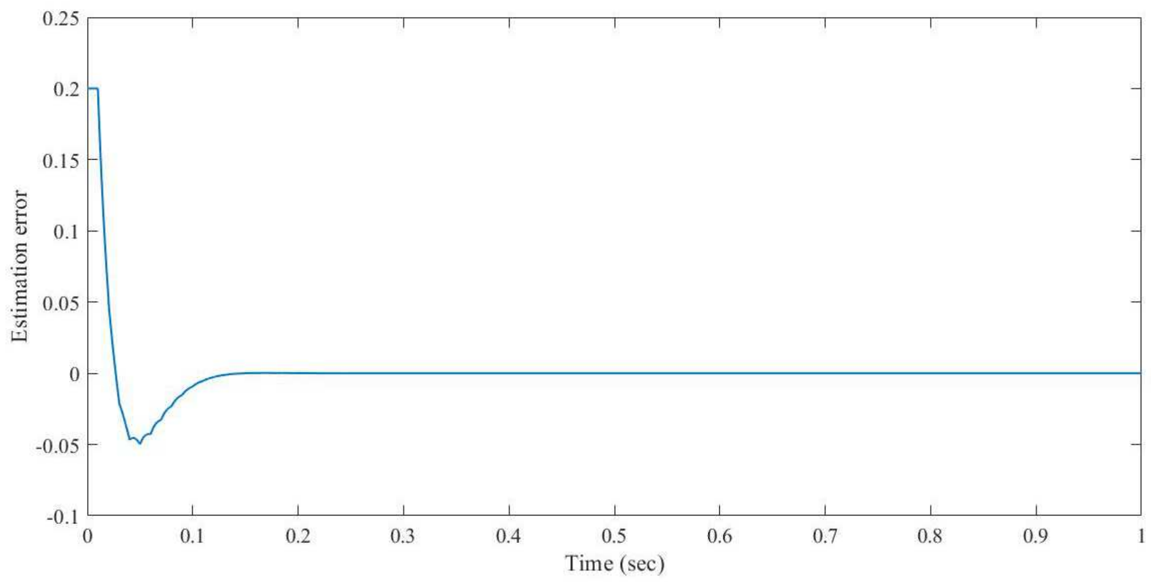


Figure 4.9: Estimation error e_2 of the second joint for sampled outputs with gain $\theta = 50$ and sampling time $T_s = 0.01$ sec.

Notice that the best estimation of the gain is $\theta = 100$ obtained in case 2 by BBO algorithm (see $MSE = 3.2608e{-}03s$) which leads to the most good estimates for system state variables. Simulation result of the first and second joint positions are presented in Figures 4.10 and 4.11, respectively. We denote that in both cases the tracking between the actual position and the estimated one is very acceptable.

Table 4.2: Optimized high gain for robot manipulator for noise free case.

Sampling period (T_s)	Optimization method	Case	number of iterations (I)	$\theta_{optimized}$	MSE
0.01	BBO	1	10	97.1760	3.3519e-03
		2	30	100	3.2608e-03
	GA	3	10	94.999	3.8003e-03
		4	30	95.8820	3.7805e-03
	PSO	5	10	86.6576	3.6658e-03
		6	30	92.6146	3.5846e-03

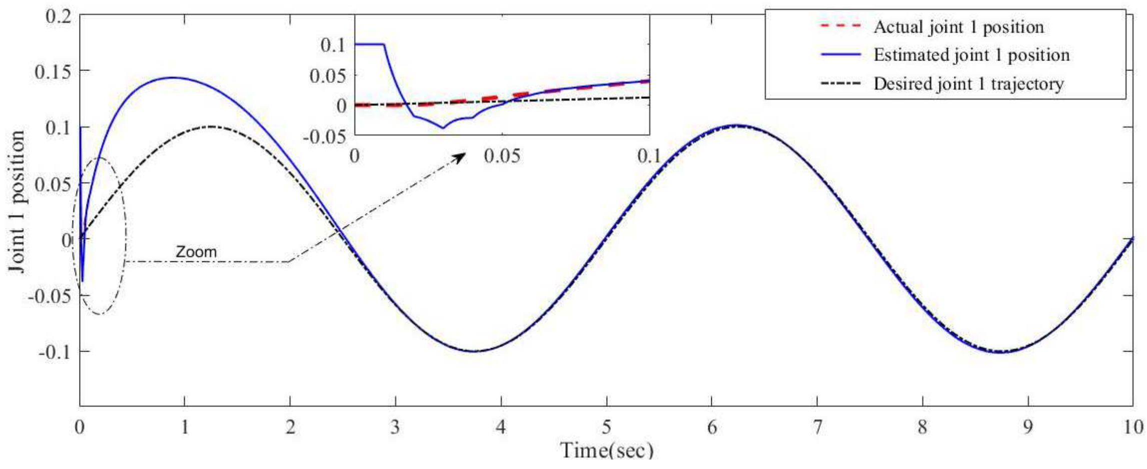


Figure 4.10: Tracking position of the first joint for sampled outputs with gain $\theta_{BBO} = 100$ and sampling time $T_s = 0.01$ sec.

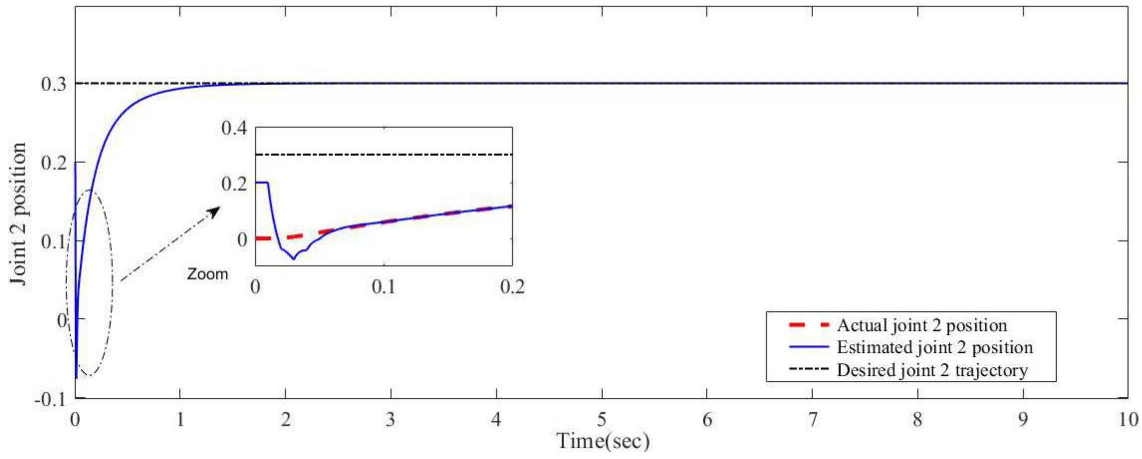


Figure 4.11: Tracking position of the second joint for sampled outputs with gain $\theta_{BBO} = 100$ and sampling time $T_s = 0.01$ sec.

B. The noisy outputs case

In this case, Gaussian noise with variance equal to 0.01 and a zero mean is added to the system output (see equation (2.11) in chapter 2).

As previously, we will present two cases, the un-optimized case and the optimized case. We give in Table 4.3, some un-optimized values of θ with their performances.

As in the un-optimized case, in the optimized case, the high gain θ is optimized by GA, PSO and BBO algorithms in order to improve the results obtained in Table 4.3. Optimized parameter values are given in Table 4.4, for two different sampling periods ($T_s = 0.01$ sec and $T_s = 0.1$ sec). Also, we see that the best value of θ is obtained with BBO algorithm given in case 2 ($T_s = 0.01$ sec) and case 8 ($T_s = 0.1$ sec). Notice that optimized θ values are lower than values obtained in the noise-free case due to the introduced stochastic environment (noise).

Simulation results relative to BBO learning for the robot joints positions are shown in Figures 4.12, 4.13, 4.14 and 4.15. In Figures 4.12 and 4.13, the estimation results are relative to sampling period $T_s = 0.01$ sec, and in Figure 4.14 and 4.15 results are relative to $T_s = 0.1$ sec. We denote that in both cases the tracking between the actual positions and the estimated ones is very acceptable.

Table 4.3: Un-optimized high gain for robot manipulator for noisy outputs case.

Sampling period (T_s)	$\theta_{un-optimized}$	MSE
0.01	5	6.0319e-02
	15	4.1121e-02
0.1	5	7.8154e-02
	15	5.6979e-02

Table 4.4: Optimized high gain for robot manipulator for noisy outputs case.

Sampling period (T_s)	Optimization method	Case	number of iterations	$\theta_{optimized}$	MSE
0.01	BBO	1	10	23.2898	1.1161e-02
		2	30	21.7166	1.0641e-02
	GA	3	10	24.7890	1.1550e-02
		4	30	23.8950	1.1479e-02
	PSO	5	10	24.0878	1.1459e-02
		6	30	24.4772	1.1214e-02
0.1	BBO	7	10	9.6712	5.1673e-02
		8	30	7.4206	5.0068e-02
	GA	9	10	9.0800	5.3148e-02
		10	30	7.3160	5.2693e-02
	PSO	11	10	8.3325	5.2151e-02
		12	30	8.2878	5.2009e-02

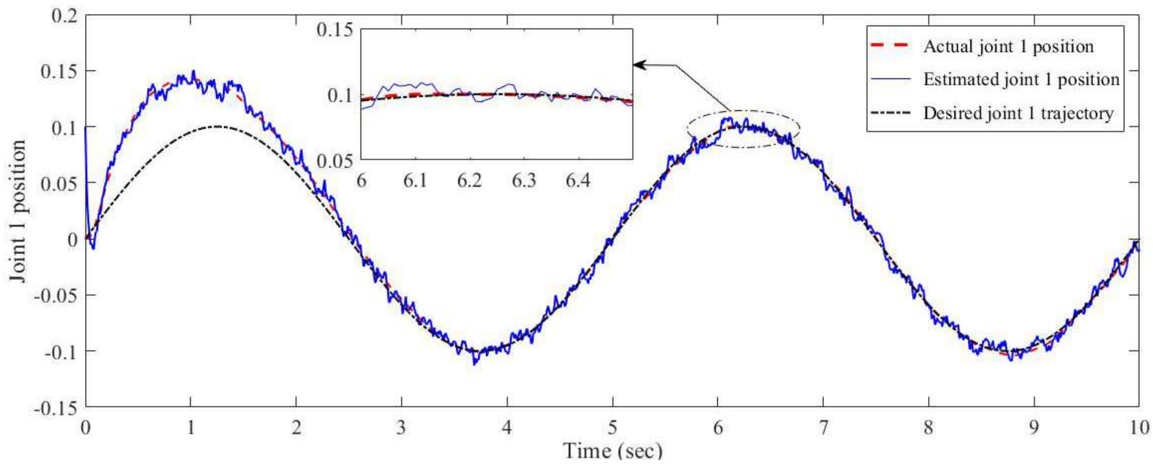


Figure 4.12: Tracking position of the first joint for sampled outputs with gain $\theta_{BBO} = 21.7166$ and sampling time $T_s = 0.01$ sec for the noisy case.

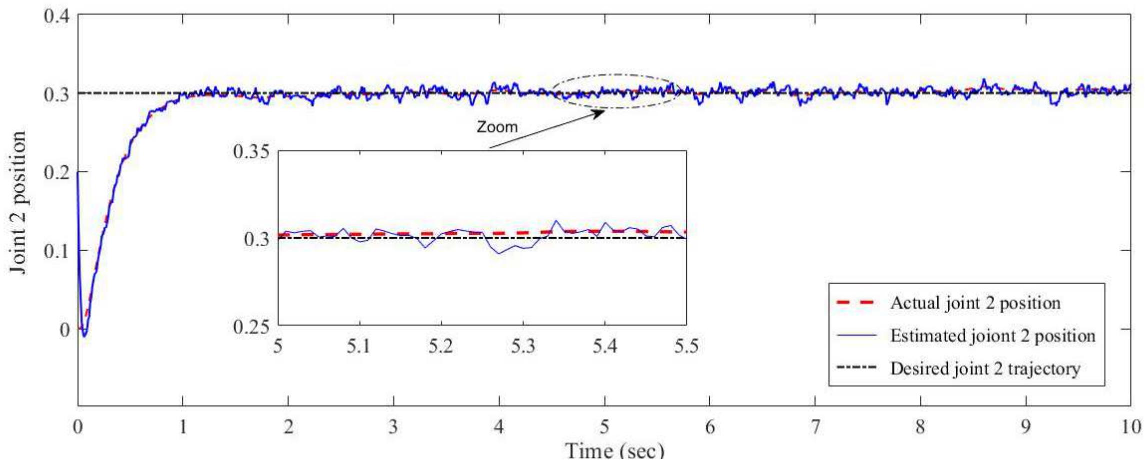


Figure 4.13: Tracking position of the second joint for sampled outputs with gain $\theta_{BBO} = 21.7166$ and sampling time $T_s = 0.01$ sec for the noisy case.

To confirm the efficiency of the proposed observer, we present in the following a statistical comparison between the best obtained results between the un-optimized and the optimized parameters. As a statistical analysis tool, we are going to use the error bars for the estimated states to evaluate the accuracy of the estimation quality. This technique is a graphical representation of the variability of the estimated variables on graphs to indicate the estimation uncertainty and provides a general idea of the precision of the estimation values. If the bars are large, this means we have bad estimation (high variability or high uncertainty), contrary, if the bars are narrow, then the estimation quality is better (less uncertainty).

We present in Figures 4.16 and 4.17 error bars comparison between the un-optimized

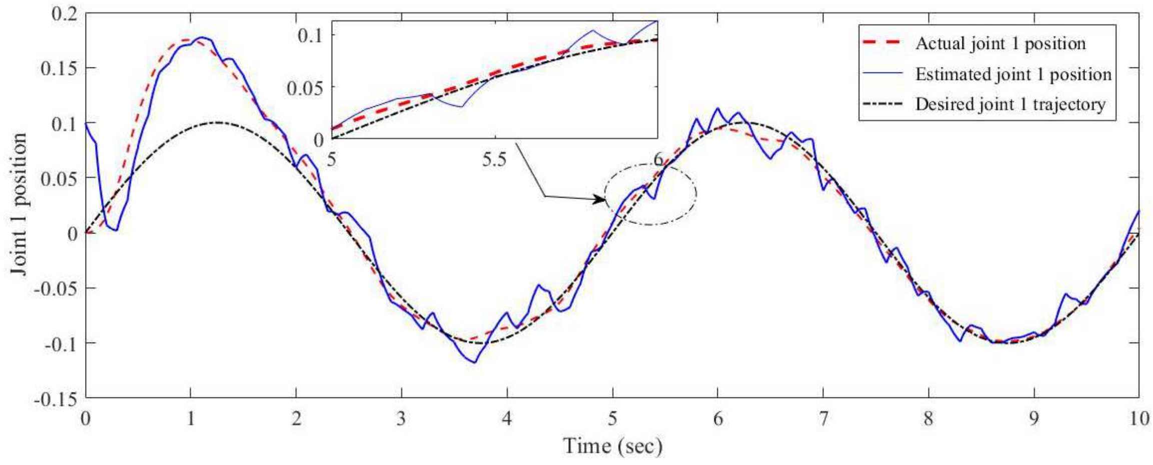


Figure 4.14: Tracking position of the first joint for sampled outputs with gain $\theta_{BBO} = 7.4206$ and sampling time $T_s = 0.1$ sec for the noisy case.

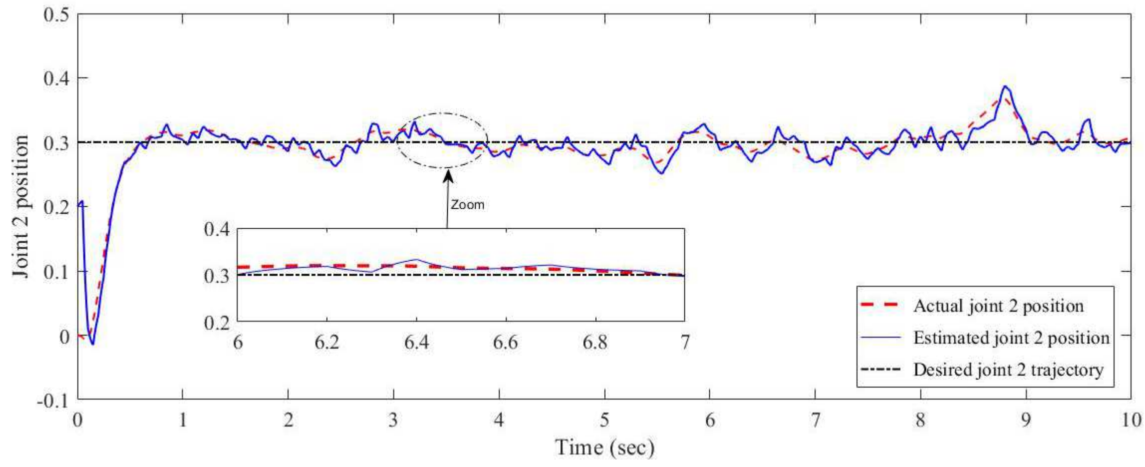


Figure 4.15: Tracking position of the second joint for sampled outputs with gain $\theta_{BBO} = 7.4206$ and sampling time $T_s = 0.1$ sec for the noisy case.

and the optimized states of the first and the second joints of the robot using parameter $\theta_{un-optimized} = 15$, $\theta_{BBO\,optimized} = 21.7166$. Notice that these parameter values are relative to the noisy case, and they are the best ones for the un-optimized and optimized cases. Figures 4.16 and 4.17 suggest clearly that widths of the error bars for the BBO method are the narrowest and more centered compared to the un-optimized case, which asserts optimization efficiency.

In Figures 4.18 and 4.19 we present error bars comparison between GA method and BBO method using the best parameters for both cases $\theta_{GA\,optimized} = 23.8950$ and $\theta_{BBO\,optimized} = 21.7166$. Error bars in Figure 4.18 and 4.19 confirm the efficiency of BBO method compared to GA method where we notice that BBO estimation variance is tighter and well centered.

Figures 4.20 and 4.21 show error bars comparison between PSO and BBO methods considering the best parameters for both cases $\theta_{PSO\,optimized} = 24.4772$ and $\theta_{BBO\,optimized} = 21.7166$. Notice that the bars for the estimation in this case confirm again the superiority of BBO method compared to PSO method (small estimation variance and good centering). We confirm by this short comparative study that BBO algorithm preserves its superiority compared to PSO and GA for the optimization of high gain observers for the estimation of state vector of a two link robotic manipulator.

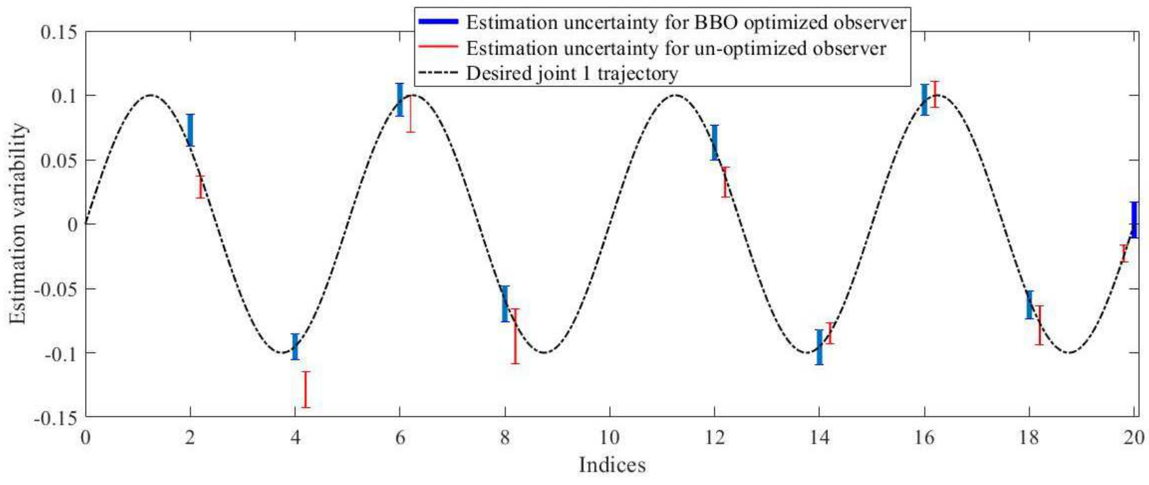


Figure 4.16: Error bar comparison of estimation variability of the first joint for $T_s = 0.01$ sec ($\theta_{un-optimized} = 15$, $\theta_{BBO\,optimized} = 21.7166$).

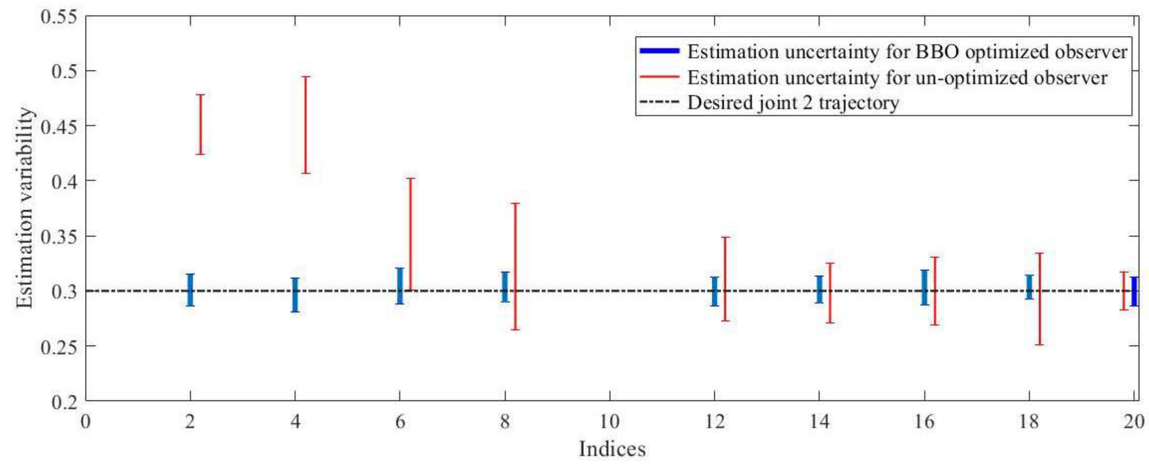


Figure 4.17: Error bar comparison of estimation variability of the second joint for $T_s = 0.01$ sec ($\theta_{un-optimized} = 15$, $\theta_{BBO\,optimized} = 21.7166$).

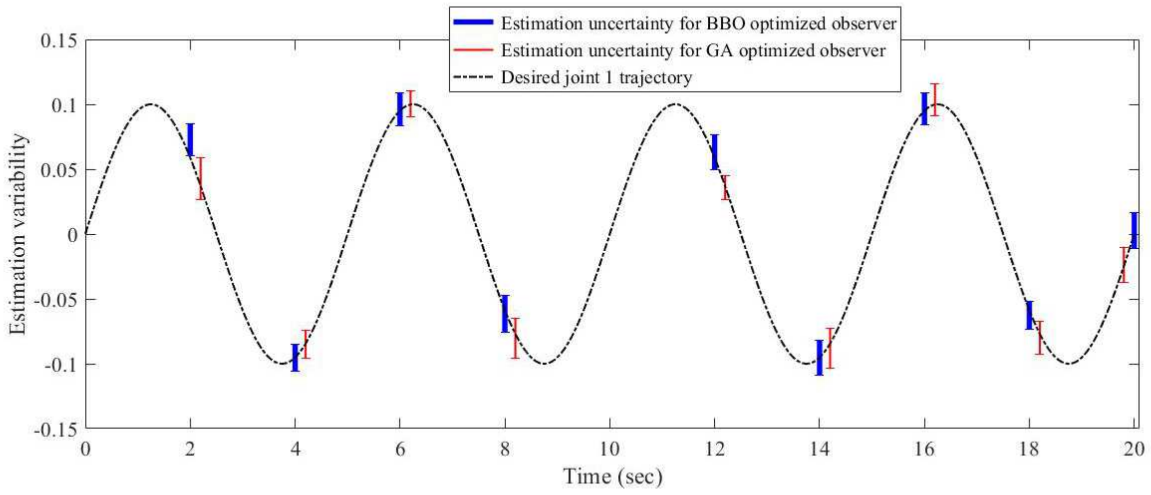


Figure 4.18: Error bar comparison of estimation variability of the first joint for $T_s = 0.01$ sec ($\theta_{GA\,optimized} = 23.8950$, $\theta_{BBO\,optimized} = 21.7166$).

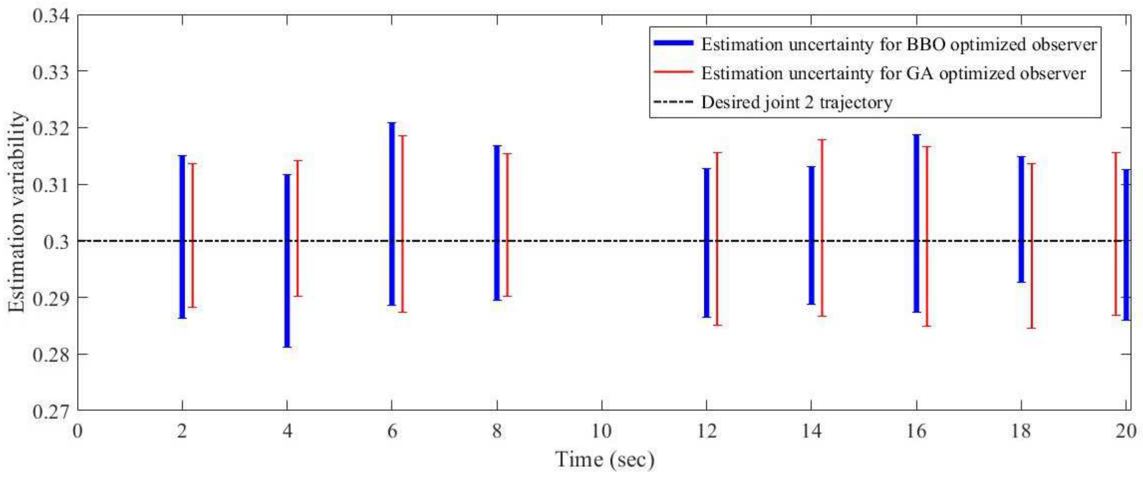


Figure 4.19: Error bar comparison of estimation variability of the second joint for $T_s = 0.01$ sec ($\theta_{GA\,optimized} = 23.8950$, $\theta_{BBO\,optimized} = 21.7166$).

4.4 Conclusion

In this chapter, the observer gain has been optimized by a relatively new optimization method called biogeography-based optimization that is employed to find the optimal estimation of the system states. System states were used to generate control actions via a feedback linearization controller. The efficiency of the proposed method was proved on a highly nonlinear and multi input multi output dynamical system which is the two link robot manipulator. Simulation results show that the optimal estimations obtained by BBO are much better than the best solutions obtained by PSO and GA algorithms.

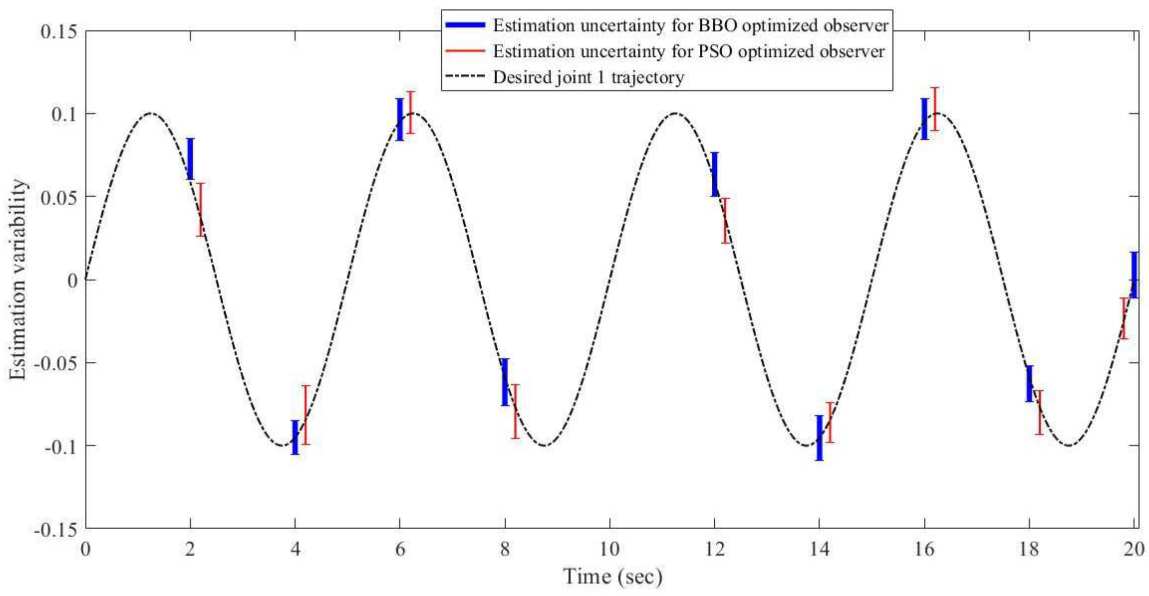


Figure 4.20: Error bar comparison of estimation variability of the first joint for $T_s = 0.01$ sec ($\theta_{PSO\,optimized} = 24.4772$, $\theta_{BBO\,optimized} = 21.7166$).

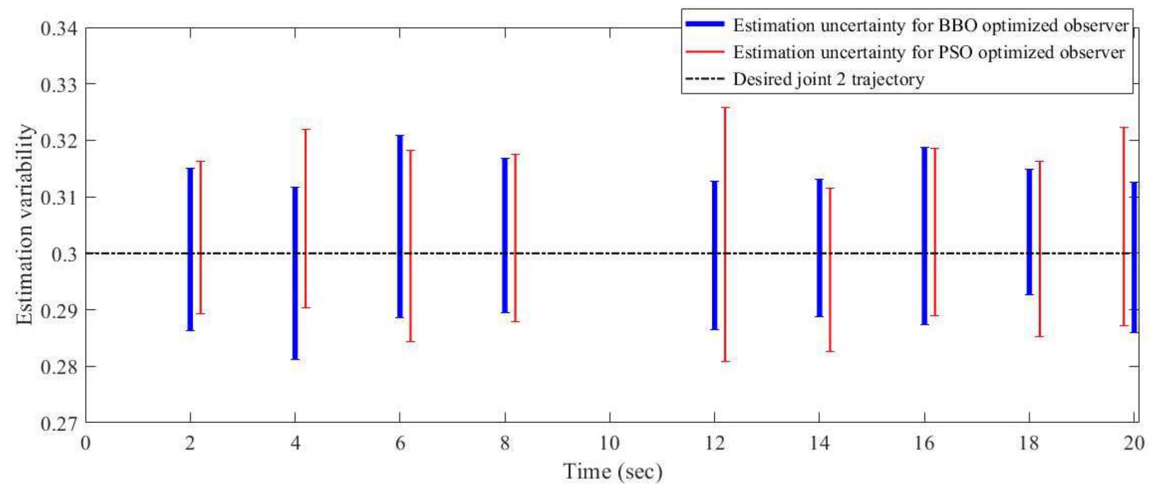


Figure 4.21: Error bar comparison of estimation variability of the second joint for $T_s = 0.01$ sec ($\theta_{PSO\,optimized} = 24.4772$, $\theta_{BBO\,optimized} = 21.7166$).

Bibliography

[1] Wang L.X. Stable adaptive fuzzy controllers with application to inverted pendulum tracking. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 26(5): 677-691, 1996. doi:10.1109/3477.537311

[2] Chafaa K., Saidi L., Ghanai M., and Benmahammed K. Direct adaptive type-2 fuzzy control for nonlinear systems. *International Journal of Computaional Intelligence and Applications*. 6(3), 389-411, 2006.

[3] Meng and Max. 1992. *Model based adaptive position and force control of robot manipulators*. Proquest Dissertations and Theses.

[4] Lyashevskiy S., and Chen Y. Designing of a class of nonlinear robotic manipulators: Contributions to control and stability. *Proceedings of 1995 34th IEEE Conference on Decision and Control*. 1995. doi:10.1109/cdc.1995.480219

CHAPTER 5

METAHEURISTIC OPTIMIZATION OF PD AND PID CONTROLLERS

Summary

5.1 Introduction

The most popular controllers used in industrial control processes are proportional-integral-derivative (PID) and proportional-derivative (PD) controllers because of their simple structures and robust performance. However, successful applications of PD and PID controllers require a satisfactory tuning of parameters according to the dynamics of the process.

This chapter presents the simulation results of optimization methods that are applied to a two link robot to estimate parameters of PD and PID controllers.

5.2 PD and PID controller

In the literature we can find different types of controllers like PD and PID where they were designed to stabilize dynamical systems. PD is one of the most important controllers and it is extensively used in different industry areas. PID is a combination of proportional, derivative and integral actions. It is an important element for distributed process control systems. Modern PID controllers are endowed with adaptive systems which can tune their free parameters. Note that PID controller acts in a very smooth and progressive manner, making sharp changes to consider the small deviations to correct rapid perturbations. Note also that PD and PID controllers are able to achieve the position control objective for robotic systems by calculating the error between the measured and the desired variables and minimizing the error by adjusting their parameters [1].

5.2.1 PD and PID controllers design

PD and PID parameters are chosen according to the system to be considered. Thus, their optimal values are very necessary to guarantee the desired performance.

A. Design of PD controller

PD controller framework is shown in Figure 5.1, where its control action is defined to be

$$\begin{aligned} \tau &= K_p\left(q_d - q\right) + K_d\left(\dot{q}_d - \dot{q}\right) + g(q) \\ &= K_p e + K_d \dot{e} + g(q) \end{aligned} \tag{5.1}$$

where q_d is the desired position vector; $\dot{q}_d$ the desired velocity vector; $g(q)$ the gravity forces $e = q_d - q$ the position error vector and $\dot{e} = \dot{q}_d - \dot{q}$ is velocity error vector. Note that q_d and $\dot{q}_d$ are compared to the actual position q and the actual velocity $\dot{q}$, respectively; and then the differences are multiplied by a position gain K_p and a velocity gain K_d to generate the control torque (5.1).

Note that an asymptotic tracking of the desired position is assured by law (5.1). Let the following Lyapunov function candidate [2]:

$$\nu = \frac{1}{2}\dot{q}^T H(q)\dot{q} + \frac{1}{2}e^T K_p e \tag{5.2}$$

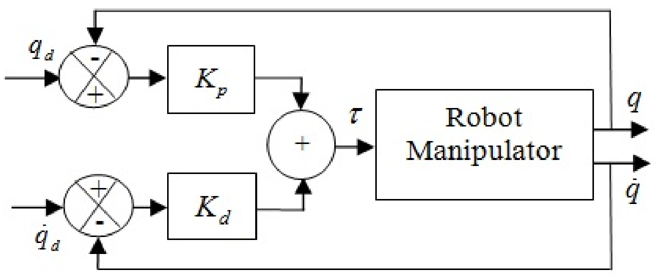


Figure 5.1: Structure of PD controller.

Lyapunov function (5.2) represents the total energy of the manipulator and it is always positive or equal to zero due to the positiveness of matrices $H(q)$ and K_p. The time derivative of v is

$$\dot{\nu} = \dot{q}^T H(q)\ddot{q} + \frac{1}{2}\dot{q}^T \dot{H}(q)\dot{q} - \dot{q}^T K_p e \tag{5.3}$$

Combining (4.5) in chapter 4 and (5.3) gives

$$\begin{aligned} \dot{\nu} &= \dot{q}^T\left(\tau - C\left(q,\dot{q}\right)\dot{q} - g(q)\right) + \tfrac{1}{2}\dot{q}^T \dot{H}(q)\dot{q} - \dot{q}^T K_p e \\ &= \dot{q}^T\left(\tau - g(q) - K_p e\right) + \tfrac{1}{2}\dot{q}^T\left(\dot{H}(q) - 2C\left(q,\dot{q}\right)\right)\dot{q} \\ &= \dot{q}^T\left(\tau - g(q) - K_p e\right) \end{aligned} \tag{5.4}$$

where we have used the fact that $\dot{H} - 2C$ is skew symmetric. Substituting PD control law (5.1) into (5.4) gives

$$\dot{\nu} = -\dot{q}^T K_d \dot{q} \leq 0 \tag{5.5}$$

The above analysis show that v decreases as long $\dot{q}$ is nonzero. In the case of $\dot{v} = 0$, (5.5) then implies that $\dot{q} \equiv 0$ and hence $\ddot{q} \equiv 0$. Using the dynamical equation (4.5) in chapter 4 and the PD control (5.1) we obtain:

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = K_p e - K_d\dot{q} + g(q) \tag{5.6}$$

then $K_p e \equiv 0$ and because K_p is nonsingular, we have $e \equiv 0$.

Therefore, control how (5.1) applied to the system (4.5) in chapter 4 achieves global asymptotic stability and the robot is therefore well-stabilized by the addition of PD-type control law.

B. Design of PID controller

A PID torque control for the robot manipulator (4.5) in chapter 4 is shown in Figure 5.2 and given by

$$\begin{aligned}\tau &= K_p\,(q_d - q) + K_i \int (q_d - q)\,dt \; + K_d\,(\dot{q}_d - \dot{q}) + g(q) \\ &= K_p e + K_i \int e\,dt - K_d\dot{q} + g(q)\end{aligned} \tag{5.7}$$

where K_p, K_i and K_d are the PID parameters to be tuned to achieve an accepted level of performance.

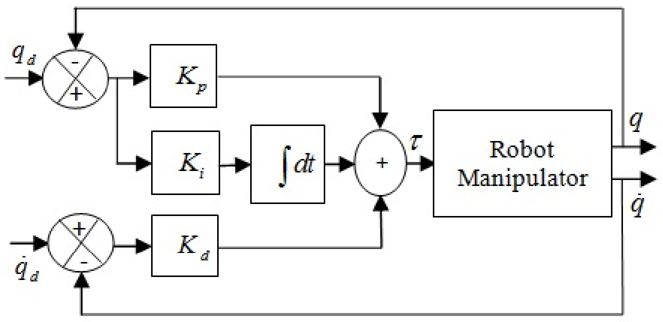


Figure 5.2: Structure of PID controller.

The kinetic energy of the manipulator is the scalar function which is represented in terms of the generalized coordinates and their derivatives as

$$K = \frac{1}{2}\dot{q}^T H(q)\dot{q} \tag{5.8}$$

The potential energy is expressed in terms of the generalized coordinates using the relationship

$$P = q^T r_c m \tag{5.9}$$

where $r_c \in \Re^n$; m denotes the mass. Defining $\dot{q} = \omega$ where $\omega \in \Re^n$; denotes the angular velocity vector.

Let rewrite (4.5) in chapter 4 as

$$\dot{\omega} = -H(q)^{-1}\left[C(q,\omega)\omega + g(q) - \tau\right] \tag{5.10}$$

The PID control function (5.7) becomes

$$\tau = K_p(q_q - q) + K_i \int (q_q - q)\,dt - K_d\omega + g(q) \tag{5.11}$$

Equation (5.1) imply that the resulting system is expressed as

$$\dot{\omega} = -H(q)^{-1}(C(q,\omega)\omega - K_p(q_q - q) \; - K_i \int (q_q - q)\, dt + K_d\omega + g(q)) \tag{5.12}$$

Let the following Lyapunov function candidate

$$\nu\,(q,\omega) = \frac{1}{2}q^T K_q q + q^T K_{q\omega}\omega \; + \frac{1}{2}\omega^T H(q)\omega + q^T r_c m \tag{5.13}$$

where $K_p \in \Re^{n\times n}$ and $K_{q\omega} \in \Re^{n\times n}$ are positive-definite matrices.
The time derivative of v is

$$\dot{\nu}\,(q,\omega) = \dot{q}^T K_q q + \dot{q}K_{q\omega}\omega + q^T K_{q\omega}\dot{\omega} \; + \dot{\omega}^T H(q)\omega + \frac{1}{2}\omega^T \dot{H}(q)\omega + r_c m \tag{5.14}$$

According to (5.12), (5.14) and because $H(q)$ is positive definite matrix, therefore it follows at once that $\dot{v}(q,\omega)$ is negative definite.

5.3 Proposed method

The PID or PD problematic is that its control is greatly affected by the parameters K_p, K_i and K_d. Bad choice for these parameters will make the result of tracking divergent or will give large errors. To surmount this difficulty and to obtain the best performances, K_p, K_i and K_d have to be considered as free parameters to be adapted. The tuning of K_p, K_i and K_d will affect both the transient time interval and steady-state operation of the response.

PID or PD parameters have to be optimized with a very high accuracy in order to obtain precise response. This task is very difficult due to the probable unknown system dynamics. To elucidate this problematic, controller parameter must be considered as free parameters to be adjusted. In the literature, the considered parameters were first tuned or adjusted by trial and error method which was a very hard task which takes long time. In order to surmount this difficulty and to avoid trial and error method, PSO with Variable Inertia Weight w, in which it will be decreased linearly with the iteration number (PSOVIW) technique was used to tune and optimize the controller parameters automatically.

In this section, we propose a new alternative for the adaptation and optimization of K_p, K_i and K_d based on the PSOVIW algorithm in order to eliminate the steady-state error, reduce the overshoot amplitude and decrease the rise time. For this purpose, we suggest to combine PSOVIW optimization with PID or PD in order to design an efficient PID (PD) for tow link robot manipulator. The framework of the proposed method is made of two steps. In the first step represented in Figure 5.3, we present a PSOVIW-PID (PD) combination working in an offline way to optimize the optimal values of K_p, K_i and K_d. In the second step, we take the optimized quantities from step one and insert them into the online PID (PD) controller of tow link robot manipulator parameters.

The structure of the PSOVIW-PID (PD) parameter optimization system is illustrated in Figure 5.3. We consider that the input of the system is the vector $r(t) = \begin{bmatrix} \theta_{d1} & \theta_{d2} \end{bmatrix}^T$ and the measured response is $y = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T$ obtained by an angle sensor (encoder). Note that the error between r and y is set to be an input for the PID (PD) as well as the optimized parameters K_p, K_i and K_d. Actual tracking errors are used by the performance evaluator. The performance evaluator estimates the objective function which is a Mean Absolute Error (MAE) criterion between the actual output and the desired reference input defined in what follows

$$MAE = \frac{1}{N}\left(\sum_{k=1}^{N}\sum_{i=1}^{2}|e_{i,k}|\right). \tag{5.15}$$

where $i = 1, 2$ is the number of robot articulations and N is the number of data samples, such that:

$e_1 = \{\theta_{1d}(k) - \theta_1(k)\}$ is the output error of the first articulation;

$e_2 = \{\theta_{2d}(k) - \theta_2(k)\}$ is the output error of the second articulation.

Based on MAE values, PSOVIW optimizer will estimate the unknown PID (PD) free parameters by updating the solutions according to PSO algorithm.

The framework of Figure 5.3 will be repeated until a preset number of iterations will be accomplished and then optimal values of PID parameters are obtained. Note that the first step in the proposed algorithm is carried out in an offline manner. This is caused due the fact that PSOVIW requires several repetitions to obtain the optimal solutions. For each iteration, the whole framework of Figure 5.3 is executed one time on the entire time interval; consequently, this structure has to be executed several times which will allow PID free parameters to be adjusted in each iteration.

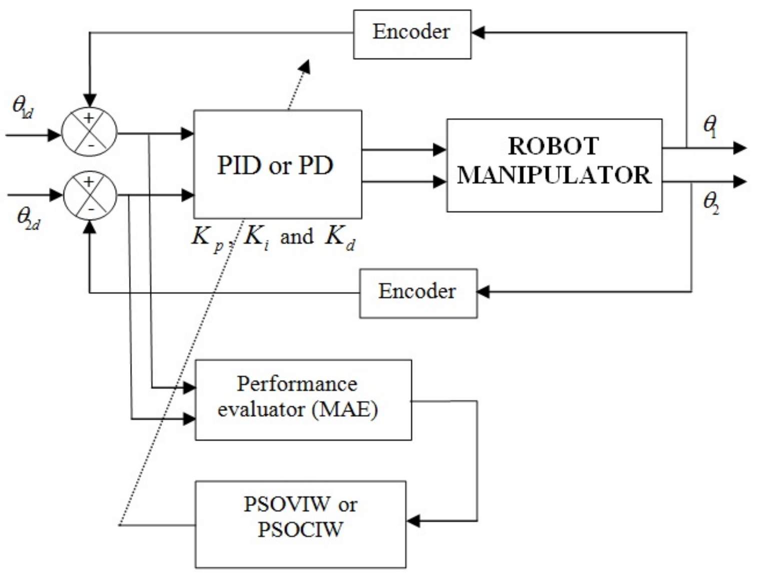


Figure 5.3: Block diagram of PSO-PID.

5.3.1 Simulation example

The Particle Swarm Optimization Algorithm (PSO) is combined with Proportional-Derivative (PD) and Proportional-Derivative-Integral (PID) to design more efficient PD and PID controllers for robotic manipulators. PSO is used to optimize the controller parameters K_p (proportional gain), K_i (integral gain) and K_d (derivative gain) to achieve better performances. The proposed algorithm is performed in two steps: (1) First, PD and PID parameters will be optimized in an offline manner by the PSO algorithm. (2) Second, the optimal parameters values are injected in the online control loop.

To verify the effectiveness of the proposed method, a two link robot manipulator with two Revolute joints (RR) shown in Figure 4.2 in chapter 4 will be considered. The block diagram of the control loop under optimization is shown in Figure 5.3.

For simulation purposes we take $m_1 = m_2 = 1\,kg$ and $a_1 = a_2 = 1\,m$. Since the dynamic of the considered system is two dimensional, therefore the joint variable and the generalized force vector will be defined by $q = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T$ and $\tau = \begin{bmatrix} \tau_1 & \tau_2 \end{bmatrix}^T$, respectively, with τ_1 and τ_2 are the torques supplied by the actuators. Note that all of our codes are written in Matlab language in M-files with sampling period $10^{-3}\,s$. For comparison purposes, the optimization performances are evaluated using the MAE criterion.

In fact, it is not simple to deduce exact values for K_p, K_i, and K_d giving the best performances. This will be solved in what follows by using our PSO-PD and PSO-PID which will allow us to obtain better results with higher precision than the classical trial and error method. It should be noted that the convergence of the PSO method to the optimal solution depends on the parameters c_1, c_2 and w. According to our tests, c_1 and c_2 best values lie in the interval $[0.5, 1.05]$; and $w \in [0.3, 1]$.

In this investigation two strategies are used for the computation of the inertia weight w to evaluate the performance of parameters. The inertia weight is introduced into the equation to balance between the capacities of the global search and the local search, as it is one of the important factors for the PSO's convergence which directly affects the percentage of previous velocities on the current velocity at the current time step for both strategies: (1) PSO with constant inertia weight w (PSOCIW), in which it will be fixed at 0.9, this high value will force the particles to fly with a significant influence of the previous velocity. Note that this method is characterized by an increase in the convergence speed of PSO algorithm and a large inertia weight factor provides PSO a global optimum. (2) PSO with variable inertia weight (PSOVIW) was introduced in PSO's equations in order to improve the performance of PSO (according to equation 5.16), in which it will be decreased linearly with the iteration number to a small value. With this low value of w, current velocity will contribute more to the particle's trajectory and provides PSO a local optimum, in contrast to the first strategy. For high values of inertia weight, the global search capability is powerful but the local search capability is powerless. Likewise, when inertia weight is lower, the local search capability is powerful, and the global search capability is powerless. This balancing improves the performance of PSO.

$$w_k = w_{\max} - \frac{w_{\max} - w_{\min}}{N} k \tag{5.16}$$

where $w_{\max} = 1$ and $w_{\min} = 0.3$ are the initial and final values of the inertia weight, respectively, and N is the maximum number of iterations used in PSO.

Note that, in this case excellent results will be obtained as will be shown later.

The PSO Parameters of the two strategies PSOCIW and PSOVIW are summarized in Table 5.1 and Table 5.2.

The best fitness functions (MAEs) and their corresponding optimized controllers gain parameters ($K_p\,, K_i\, and\, K_d$) obtained by our proposed approaches (combination

Table 5.1: Parameters of the PSOCIW algorithm.

Designation	Variable	Value
Number of particles in a group	N	20
Number of iterations	I	$40, 50, 60, 100$
Inertia weight factor	w	0.9
Acceleration constants	c_1, c_2	0.5

Table 5.2: Parameters of the PSOVIW algorithm.

Designation	Variable	Value
Number of particles in a group	N	20
Number of iterations	I	$40, 50, 60, 100$
Minimum inertia weight factor	w_{min}	0.3
Maximum inertia weight factor	w_{max}	1
Acceleration constants	c_1, c_2	1.05

PSO-PD and PSO-PID) (see Figure 5.3) are reported in Table 5.3 and Table 5.4, respectively.

Table 5.3: Optimized parameters and their performances for PD controller using PSO.

Optimization method	Case	I	K_{p1}	K_{d1}	K_{p2}	K_{d2}	MAE
PSOCIW	1	40	553.5999	88.3948	376.4420	37.3761	1.0737e-05
	2	50	556.1911	87.0808	369.5691	31.2962	9.7914e-06
	3	60	558.1911	87.0808	369.5691	31.2962	1.3111e-06
	4	100	558.1911	87.0808	369.5691	31.2962	1.3111e-06
PSOVIW	5	40	521.1328	51.3526	544.2311	75.9123	6.6234e-06
	6	50	597.0536	62.0977	560.3582	67.0380	1.5341e-06
	7	60	816.1067	89.9357	418.7053	43.7568	6.0632e-10
	8	100	663.2064	113.7360	695.1595	105.5423	1.5916e-14

Optimization results show that the method is able to find the optimal solution and reduce the error efficiently within 100 iterations. We note that the best value of MAE which corresponds to the best estimate of PD and PID gain parameters are given in case 8 of Table 5.3 and case 7 of Table 5.4, respectively.

The convergence of the fitness functions are shown in Figure 5.4 and Figure 5.5 for PD and PID controllers; respectively, where we notice that the MAE is decreased at most after 10 iterations, which confirms the convergence and the stability of optimization process.

Table 5.4: Optimized parameters and their performances for PID controller using PSO.

Optimization method	Case	I	K_{p1}	K_{i1}	K_{d1}	K_{p2}	K_{i1}	K_{d2}	MAE
PSOCIW	1	40	575.8960	392.3983	112.5328	392.4810	462.4569	9.5001	1.8169e-04
	2	50	575.89601	392.3983	112.5328	392.4810	462.4569	9.5001	1.8169e-04
	3	60	575.89601	392.3983	112.5328	392.4810	462.4569	9.5001	1.8169e-04
	4	100	577.0524	356.7020	105.4120	344.7770	432.3530	10.1167	1.3856e-05
PSOVIW	5	40	417.6209	123.9084	71.2909	644.9327	625.4555	79.8841	3.7550e-18
	6	50	509.3163	181.1770	65.0094	775.7143	661.6406	73.8510	3.0524e-18
	7	60	435.5346	100.8813	56.4018	621.7791	517.8345	59.1068	3.7466e-19
	8	100	426.8821	124.1054	76.5055	641.5470	546.7384	83.0836	1.1656e-18

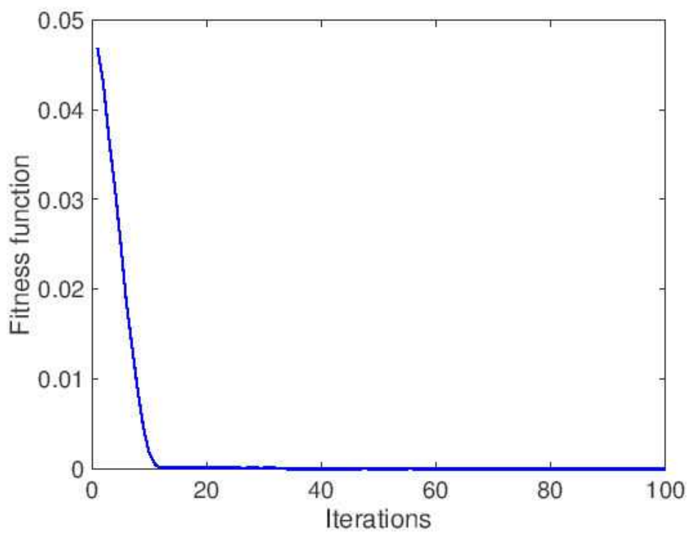


Figure 5.4: Evolution of the fitness function relative to PD (case of 8 iterations in Table 5.3).

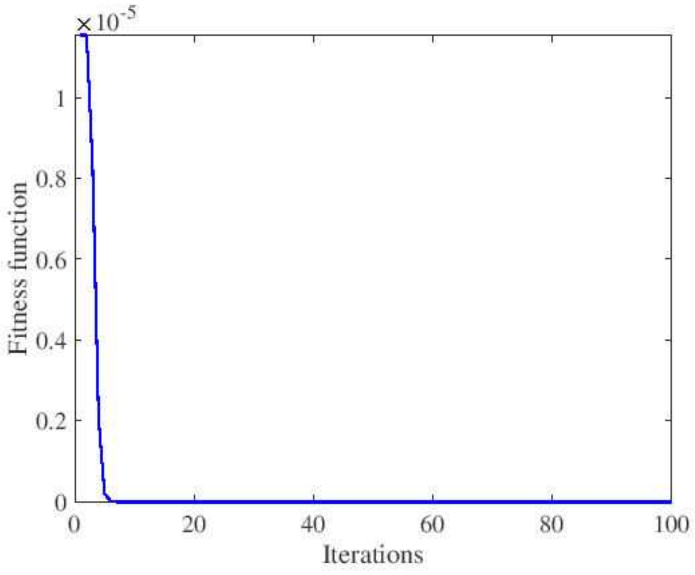
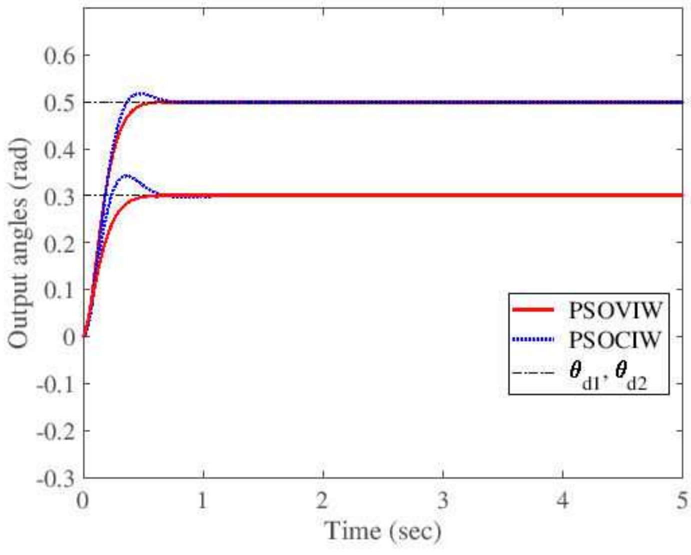


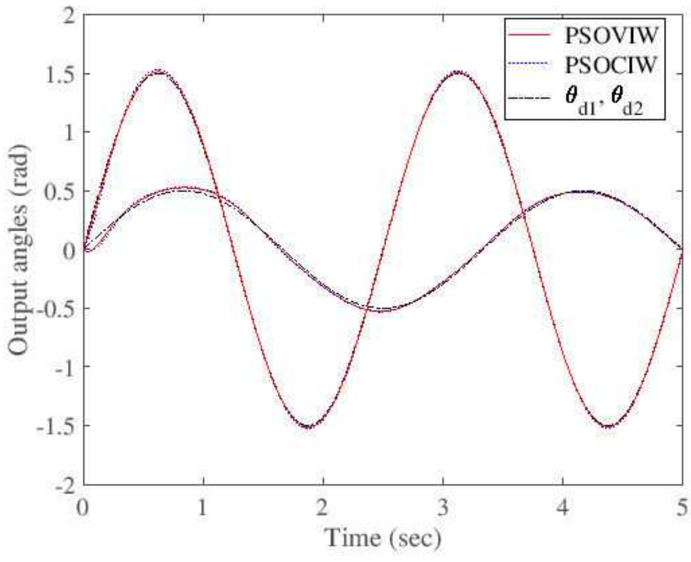
Figure 5.5: Evolution of the fitness function relative to PID (case of 8 iterations in Table 5.4).

Simulation results for PD controller are shown in Figure 5.6 where we can see clearly that the performances of PSOVIW are better than those of PSOCIW for both cases constant desired trajectory (Figure 5.6(a)) and sinusoidal desired trajectory (Figure 5.6(b)). We also present in Figure 5.7 and Figure 5.8 the corresponding tracking error and control action relative to the first articulation. Remark that the tracking error converges exponentially to zero.

Same interpretation for PID controller (see Figure 5.9) in which we visualize also an excellent perturbation rejection performance. Note that the perturbation was applied at $t = 5\,s$ with amplitude $5N$. The corresponding tracking error and control action relative to the first articulation of the robot are presented in Figure 5.10 and Figure 5.11 where we denote perfect performances.



(a)



(b)

Figure 5.6: PD Positions tracking with the best optimized parameters (a) Constant desired trajectory (b) Sinusoidal desired trajectory.

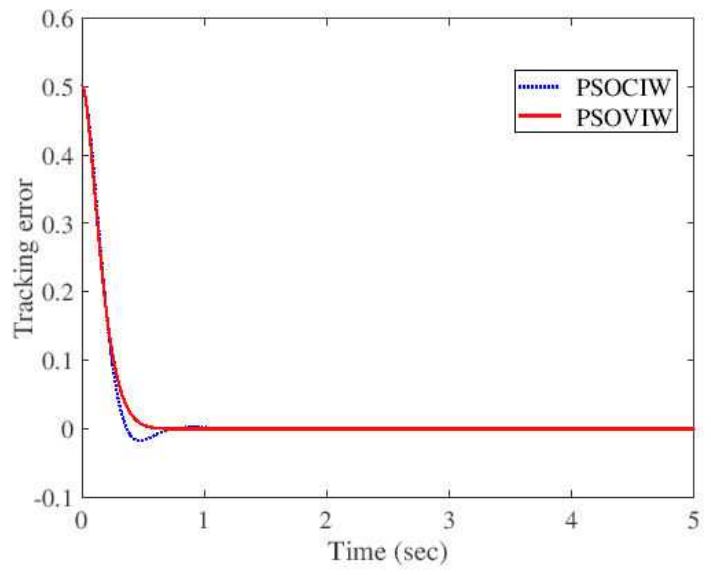


Figure 5.7: Tracking position error e_1 of the first articulation.

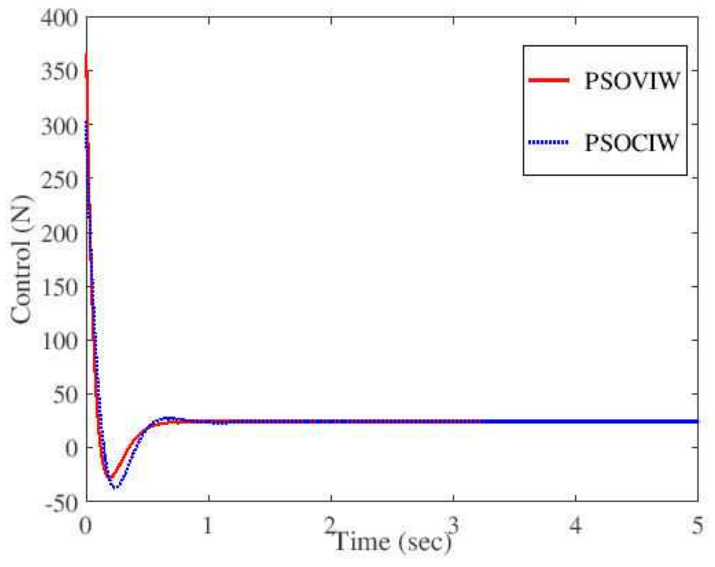
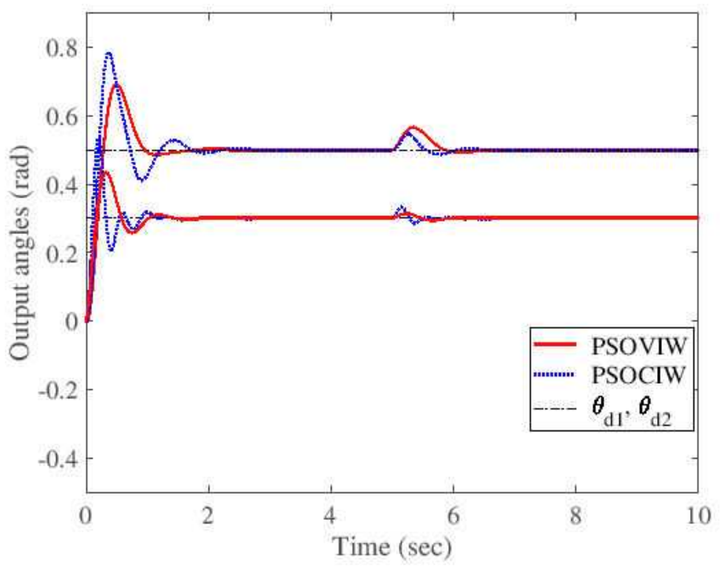
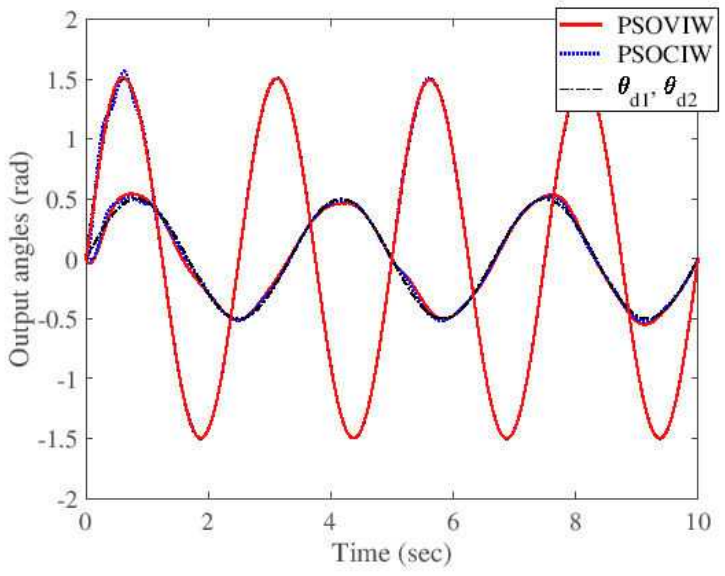


Figure 5.8: Input control action u_1 of the first articulation.

To validate the proposed approach, we will present in what follows a short comparative study in which we compare the introduced method with Genetic Algorithm. Genetic Algorithm will play now the same role as PSO, which means that we replace in Figure 5.3 the PSO block by a GA block. For this purpose, GA will estimate and optimize the controller parameters in two steps as in the case of PSO.



(a)



(b)

Figure 5.9: PID Positions tracking and perturbation rejection with the best optimized parameters: Constant desired trajectory (a); Sinusoidal desired trajectory (b).

The parameters of GA are chosen as shown in Table 5.5 where we have selected different generations $I = 40, 50, 60, 100$ with the following parameters: population size $N = 20$, mutation probability $M = 0.2$ and crossover probability $C = 0.5$.

GA fitness functions evolution are presented in Figure 5.12 and Figure 5.13 for PD and PID controllers; respectively, where we notice that the MAE is converged in 20 iterations max, contrary to PSO case which converges in 10 iterations max (see Figure 5.4,

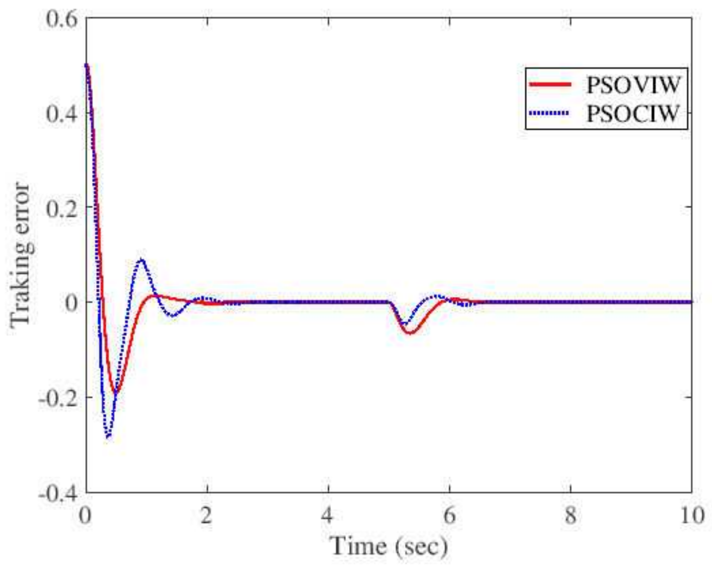


Figure 5.10: Tracking position error e_1 of the first articulation.

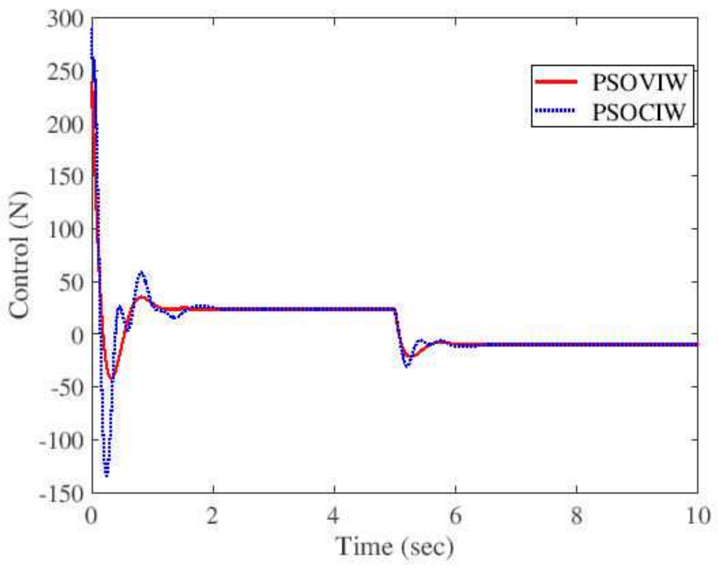


Figure 5.11: Input control action u_1 of the first articulation.

Figure 5.5, Figure 5.12 and Figure 5.13) which confirm that PSO is faster than GA. Note also that both PSO methods (PSOCIW and PSOVIW) give more accurate results compared to GA method (see Table 5.3, Table 5.6, Table 5.4 and Table 5.7).

By this short comparative study, we confirm the effectiveness of the proposed method and its superiority in speed convergence high resolution (see Figure 5.14, Figure 5.15, Figure 5.16, Figure 5.17, Figure 5.18 and Figure 5.19).

Table 5.5: Parameters of the genetic algorithm.

Designation	Variable	Value
Population size	N	20
Generations	I	$40, 50, 60, 100$
Mutation probability	M	0.2
Crossover probability	C	0.5

Table 5.6: Optimized parameters and their performances for PD controller using GA.

Optimization method	Case	I	K_{p1}	K_{d1}	K_{p2}	K_{d2}	MAE
GA	1	40	64.6040	28.6951	46.9204	10.1212	1.8757e-04
	2	50	76.2287	30.2603	74.9499	13.9796	6.3897e-05
	3	60	52.9399	23.6698	79.5247	14.4101	9.3264e-05
	4	100	89.0021	30.7783	49.2828	1.3190	9.9626e-05

Table 5.7: Optimized parameters and their performances for PID controller using GA.

Optimization method	Case	I	K_{p1}	K_{i1}	K_{d1}	K_{p2}	K_{i1}	K_{d2}	MAE
GA	1	40	45.9883	3.7957	23.2161	44.7578	10.5608	2.2023	1.0940e-03
	2	50	77.2967	7.7502	30.5331	64.3610	49.7162	5.1843	3.3240e-03
	3	60	77.6165	7.5343	32.2968	87.5616	19.1396	3.1810	8.7338e-04
	4	100	110.8116	15.3839	33.3837	75.5687	22.9882	3.9823	1.5665e-04

To confirm the efficiency of the proposed control, we present in the following a statistical comparison between the best obtained results of PSOVIW and GA for PID case. As a statistical analysis tool, we are going to use the error bars for the optimized parameters to evaluate the accuracy of the control quality. This technique is a graphical representation of the variability of the optimized parameters (on graphs) to indicate the uncertainty and provides a general idea of the tracking precision. If the bars are large, this means we have bad optimization (high variability or high uncertainty), contrary, if the bars are narrow, then the optimization quality is better (less uncertainty).

We present in Figure 5.20 and Figure 5.21 error bars comparison between the PSOVIW and GA of the first and the second joints of the robot using the best parameters given in the last line in Table 5.4 for PSOVIW case and the best parameters given in the last line in Table 5.7 for GA case. For this purpose, we introduce state random

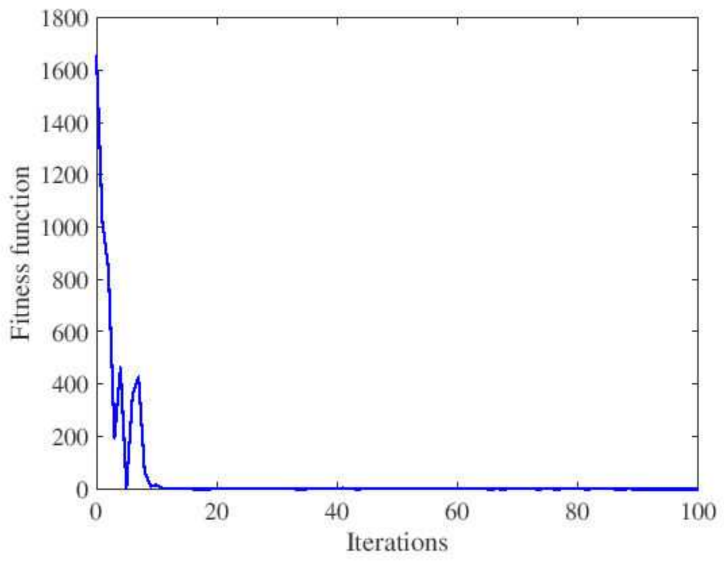


Figure 5.12: Evolution of the fitness relative to PD (case 4 iterations in Table 5.6).

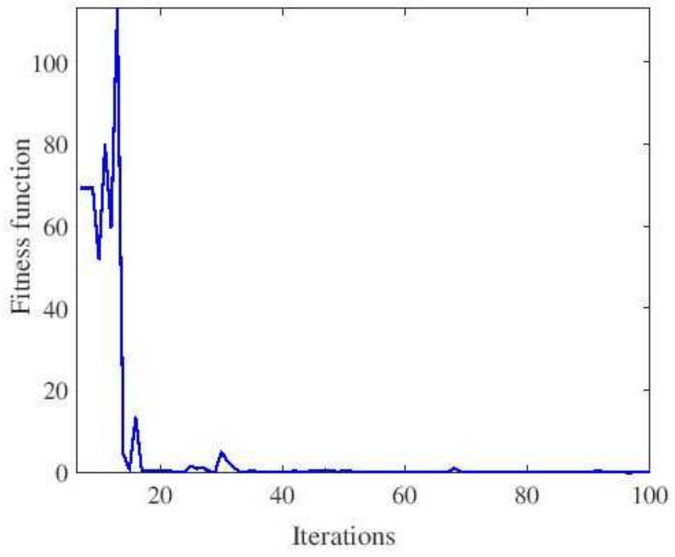
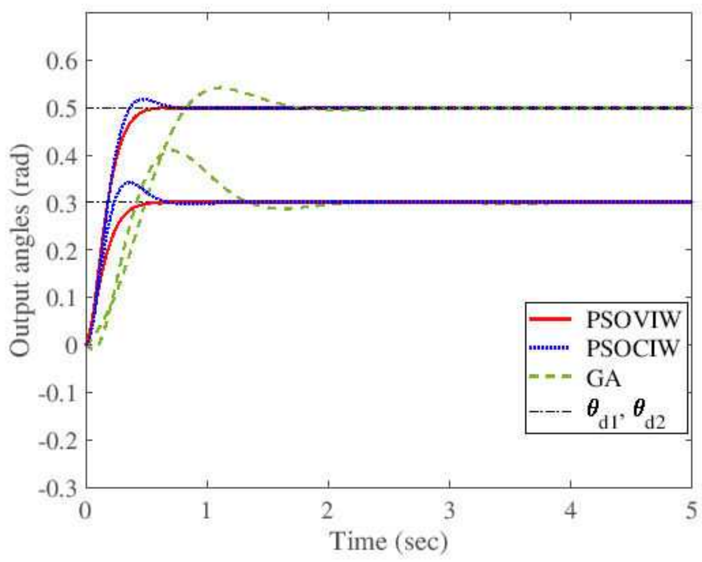
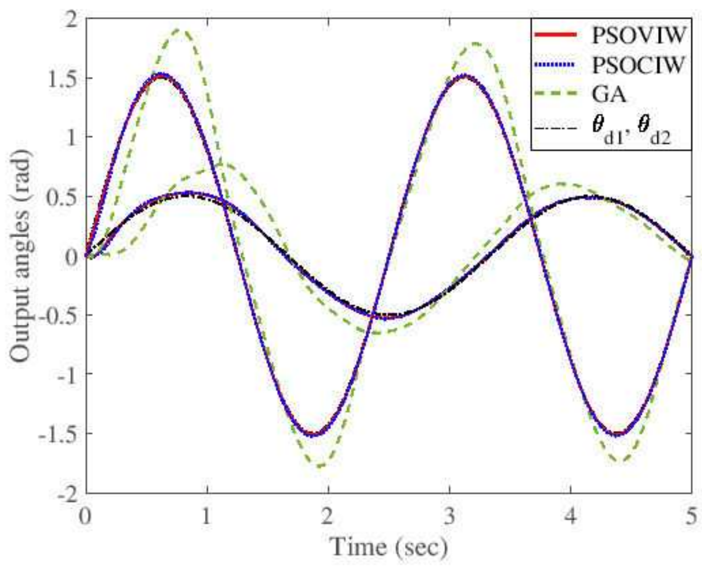


Figure 5.13: Evolution of the fitness relative to PID (case 4 iterations in Table 5.7).

noise $n(t)$ with variance one and mean value zero i.e., $n(t) \sim N(0,1)$. Figure 5.20 and Figure 5.21 suggest clearly that widths of the error bars for the PSOVIW method are the narrowest and more centered compared to the GA case. Error bars in Figure 5.20 and Figure 5.21 confirm the efficiency of PSOVIW method compared to GA method where we notice that PSOVIW tracking variance is tighter and well centered.



(a)



(b)

Figure 5.14: PD Positions tracking with the best optimized parameters: Constant desired trajectory (a); Sinusoidal desired trajectory (b).

5.4 Conclusion

In this chapter, PD and PID gains have been adjusted by particle swarm optimization. The introduced algorithm was tested on the control of a two link robot manipulator. Simulation results show that the method gives an excellent performance. Also, the effectiveness of the approach was confirmed by a short comparative study in which we found that it outperforms the genetic algorithm technique for this type of applications, where the resulting estimates are more precise and the optimization is faster than GA. Furthermore, we concluded that the PSOVIW approach, which used variable inertia weight, performed better results than GA and PSOCIW.

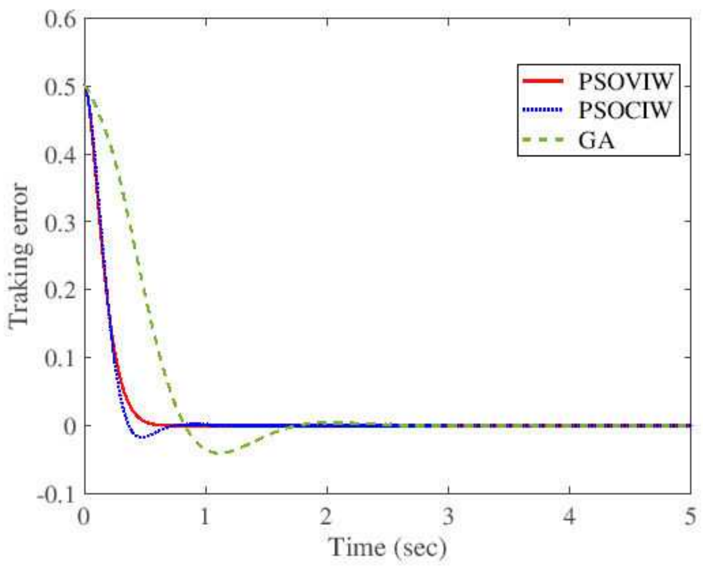


Figure 5.15: Tracking position error e_1 of the first articulation.

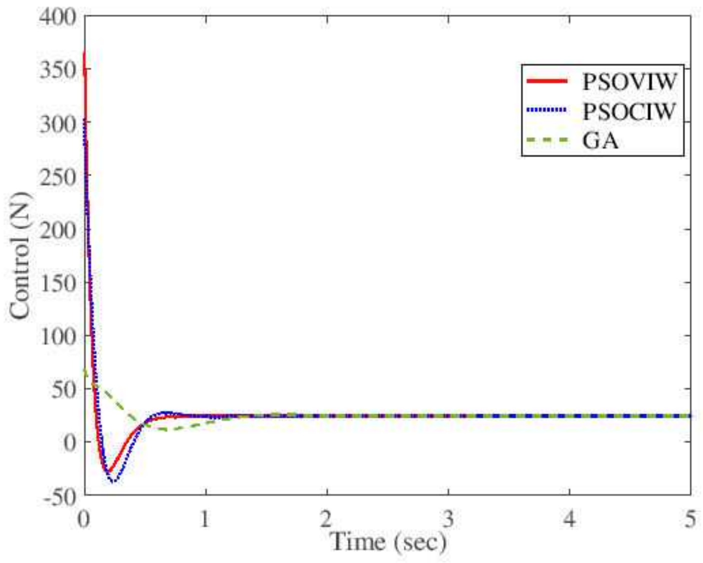
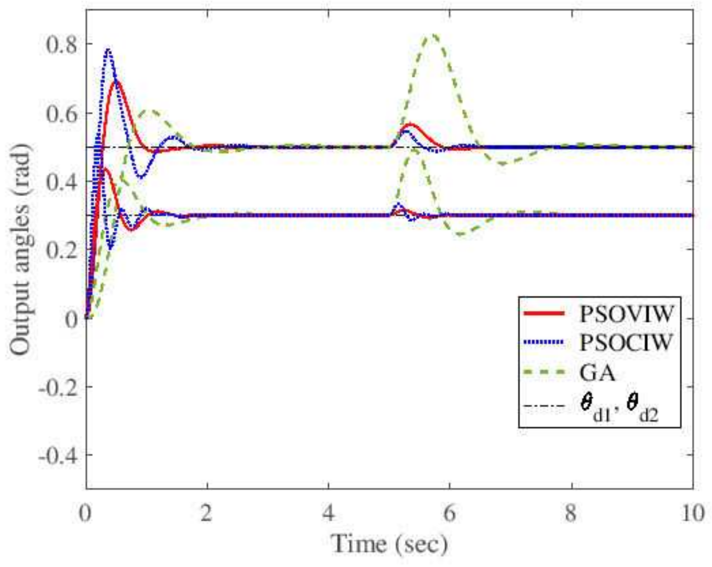
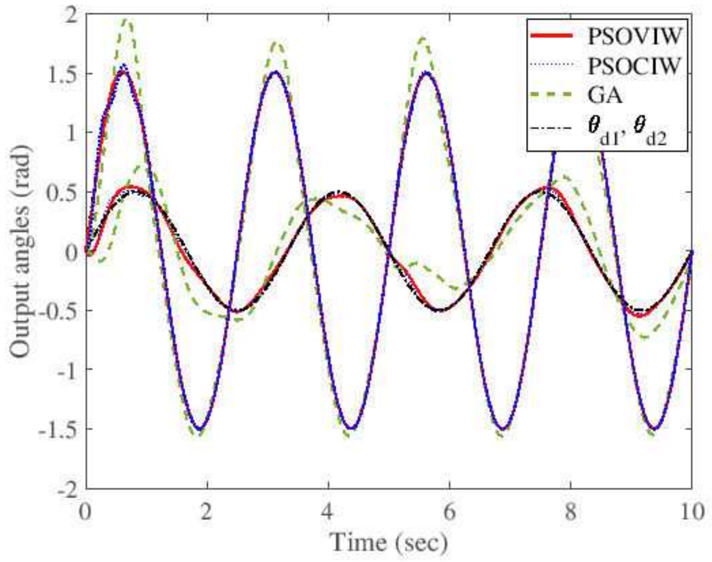


Figure 5.16: Input control action u_1 of the first articulation.



(a)



(b)

Figure 5.17: PID Positions tracking and perturbation rejection with the best optimized parameters: Constant desired trajectory (a); Sinusoidal desired trajectory (b).

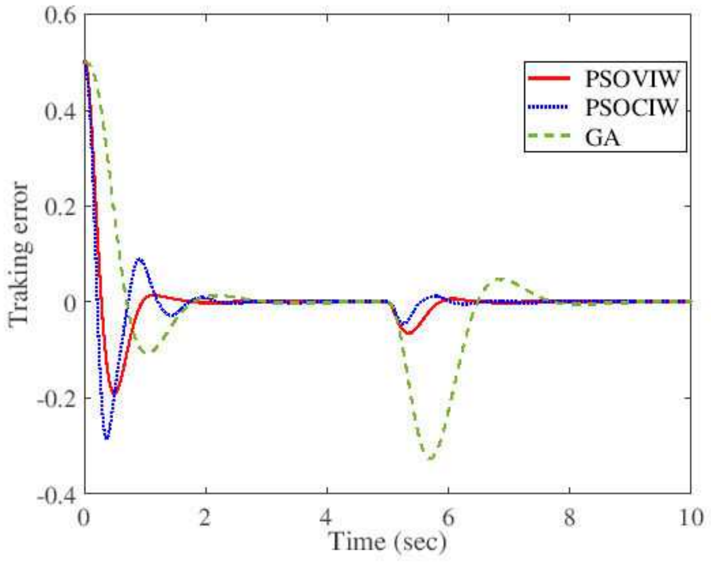


Figure 5.18: Tracking position error e_1 of the first articulation.

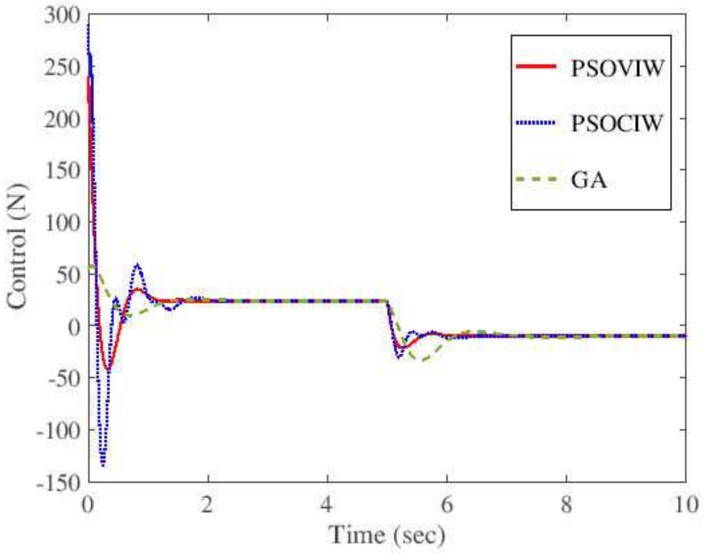


Figure 5.19: Input control action u_1 of the first articulation.

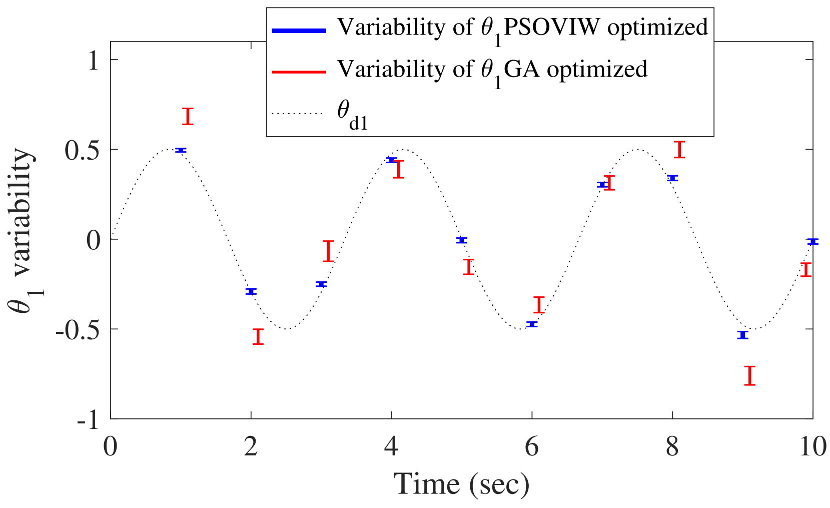


Figure 5.20: Error bar comparison of tracking variability of the first joint.

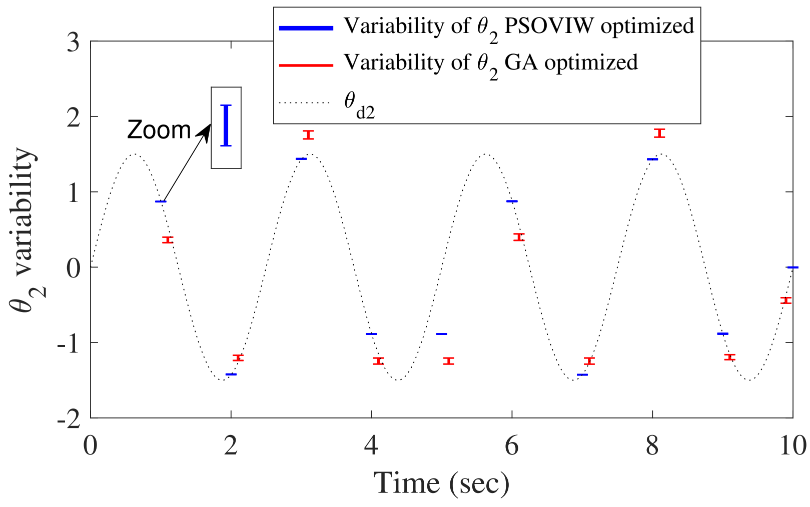


Figure 5.21: Error bar comparison of tracking variability of the second joint.

Bibliography

[1] Jaleel J.A., and Thanvy N. A comparative study between PI, PD, PID and lead-lag controllers for power system stabilizer. *IEEE Trans. Int Conf. Circuits, Power and Computing Technologies [ICCPCT]*. 456-460. 2013.

[2] Meng and Max. 1992. *Model based adaptive position and force control of robot manipulators*. Proquest Dissertations and Theses.

CHAPTER 6

INDIRECT ADAPTIVE CONTROL

Summary

6.1 Introduction

In this chapter, we develop an indirect adaptive control approach based on type-2 fuzzy logic, to control the nonlinear dynamic system defined in chapter 1. In this chapter, we have presented an indirect adaptive control structure, where the fuzzy system was used as a fuzzy model to identify the system to be controlled. Another supervisor regulator is added to stabilize the closed loop system once it tends to destabilize [1].

6.2 Induction motor and vector control

The dynamic behaviour of an induction machine in dq synchronous reference is described by:

$$\frac{di_{ds}}{dt} = \frac{1}{\sigma L_s}(-(R_s + (\frac{L_m}{L_r})^2 R_r)i_{ds} + \sigma L_s \omega_s i_{qs} + \frac{L_m R_r}{L_r^2}\phi_{dr} + \frac{L_m}{L_r}\phi_{qr}\omega_r + u_{ds}) \tag{6.1}$$

$$\frac{di_{qs}}{dt} = \frac{1}{\sigma L_s}(-\sigma L_s \omega_s i_{ds} - (R_s + (\frac{L_m}{L_r})^2 R_r)i_{qs} - \frac{L_m}{L_r}\phi_{dr}\omega_r + \frac{L_m R_r}{L_r^2}\phi_{qr} + u_{qs}) \tag{6.2}$$

$$\frac{d\phi_{dr}}{dt} = \frac{L_m}{T_r}i_{ds} - \frac{\phi_{dr}}{T_r} + (\omega_s - \omega_r)\phi_{qr} \tag{6.3}$$

$$\frac{d\phi_{qr}}{dt} = \frac{L_m}{T_r}i_{qs} - \frac{\phi_{qr}}{T_r} - (\omega_s - \omega_r)\phi_{dr} \tag{6.4}$$

$$\frac{d\omega_r}{dt} = \frac{p}{J}T_e - \frac{B}{J}\omega_r - \frac{p}{J}T_L \tag{6.5}$$

where u_{ds}, u_{qs} are $d-$ and $q-$axis stator voltages; i_{ds}, i_{qs} are $d-$ and $q-$axis stator currents; ϕ_{dr}, ϕ_{qr} are $d-$ and $q-$axis rotor flux linkages; ω_s, ω_r are stator angular frequency and rotor electrical angular speed; R_s, R_r are stator and rotor resistances; L_s, L_r and L_m are stator inductance, rotor inductance and mutual inductance; $T_r = L_r/R_r$ is the rotor time constant; σ is the total leakage factor $\sigma = 1 - L_m^2/(L_s L_r)$; T_e, T_L are electromagnetic torque and load torque; p is the number of poles; J is the inertial moment of the motor; and B is viscous friction coefficient. The produced electromagnetic torque can be written in terms of stator currents and rotor fluxes as:

$$T_e = \frac{pL_m}{L_r}(i_{qs}\phi_{dr} - i_{ds}\phi_{qr}) \tag{6.6}$$

The decoupling control of torque and rotor flux can be obtained using the vector control technique [2, 3]. In the rotor flux oriented vector, the flux is oriented to the $d-$axis, so that $\phi_{qr} = 0$, and kept at a constant rated value $\phi_{dr} = \phi_r$. At steady-state, slip angular frequency can be expressed as:

$$\omega_{sl} = \omega_s - \omega_r = \frac{L_m R_r i_{qs}}{L_r \phi_r} \tag{6.7}$$

the generated motor torque T_e (6.6) is reduced to a linear function of the torque current component i_{qs}:

$$T_e = \frac{pL_m\phi_r}{L_r}i_{qs} \tag{6.8}$$

the application of vector control in the current model of IM leads to the following equations:

$$\frac{d\phi_r}{dt} = -\frac{\phi_r}{T_r} + \frac{L_m}{T_r} i_{ds} \tag{6.9}$$

$$\frac{dN}{dt} = -\frac{B}{J} N + \frac{30pL_m\phi_r}{\pi J L_r} i_{qs} - \frac{30}{\pi J} T_L \tag{6.10}$$

where ϕ_r is the rotor flux, N is the motor speed expressed in revolution per minute and (i_{ds}, i_{qs}) are the components of stator current.

6.3 Problem formulation

Consider the first order nonlinear dynamical system of the form:

$$\begin{cases} \dot{x} = f(x) + g(x)u + d \\ y = x \end{cases} \tag{6.11}$$

The control objective is to find a feedback control law u such that to make the state $x\,(t)$ track a given desired bounded reference trajectory $y_m\,(t)$. It is known that if the plant model is not known, it is intuitively reasonable to replace it by an estimated model and use this model for designing the controller. This is the basic idea of an indirect adaptive controller, in which the controller is designed based on an estimated model of the plant assuming this model is the true model of the plant, and the estimated model parameters are updated by an on-line algorithm. If the plant dynamics of (6.11) is known, i.e., f and g are known and the system is free of external disturbance d, we can solve the control problem stated above by the so-called feedback linearization method. In this method, f and g are used to construct the following feedback controller:

$$u = \frac{1}{g(x)} \left[-f(x) + \dot{y}_m(t) + ke \right] \tag{6.12}$$

where $e = e(t) = y_m(t) - y(t)$ is the tracking error, and k is chosen such that the root of the polynomial $h(s) = s + k$ is in the open left-half of the complex plane. Applying the control law given in (6.12) to the system given in (6.11) we obtain the following error dynamics:

$$\dot{e} + ke = 0 \tag{6.13}$$

where the main objective of the control is $\lim\limits_{t\to\infty} e(t) = 0$. However, since f and g are unknown, we cannot use them for constructing the control law given by (6.12).

Therefore, in the following, we replace them by their estimates $\hat{f}$ and $\hat{g}$ to construct an adaptive controller:

$$u_c = u_c(x/\underline{\theta}_f, \underline{\theta}_g) = \frac{1}{\hat{g}(x/\underline{\theta}_g)}\left[-\hat{f}(x/\underline{\theta}_f) + \dot{y}_m(t) + ke\right] \tag{6.14}$$

where θ_f and θ_g are parameters of the approximating systems $\hat{f}$ and $\hat{g}$, respectively.

Figure 6.3 shows a bloc diagram of control structure based on IFOC induction motor fed by a current-controlled PWM voltage-source inverter. The procedure of hysteresis current control used here consists of a comparison between the current errors against a fixed hysteresis band. The system uses two control loops: flux control and speed control to yield I_{ds} and I_{qs} which represent the controlled flux and torque components respectively. In order to maintain the stator current in acceptable range, the current inputs (I_{ds}, I_{qs}) are transformed into limited inputs (I^*_{ds}, I^*_{qs}). The instantaneous three-phase reference current $(i^*_{as}, i^*_{bs}, i^*_{cs})$ is obtained from the dq stator current (I^*_{ds}, I^*_{qs}) by applying the inverse Park transform.

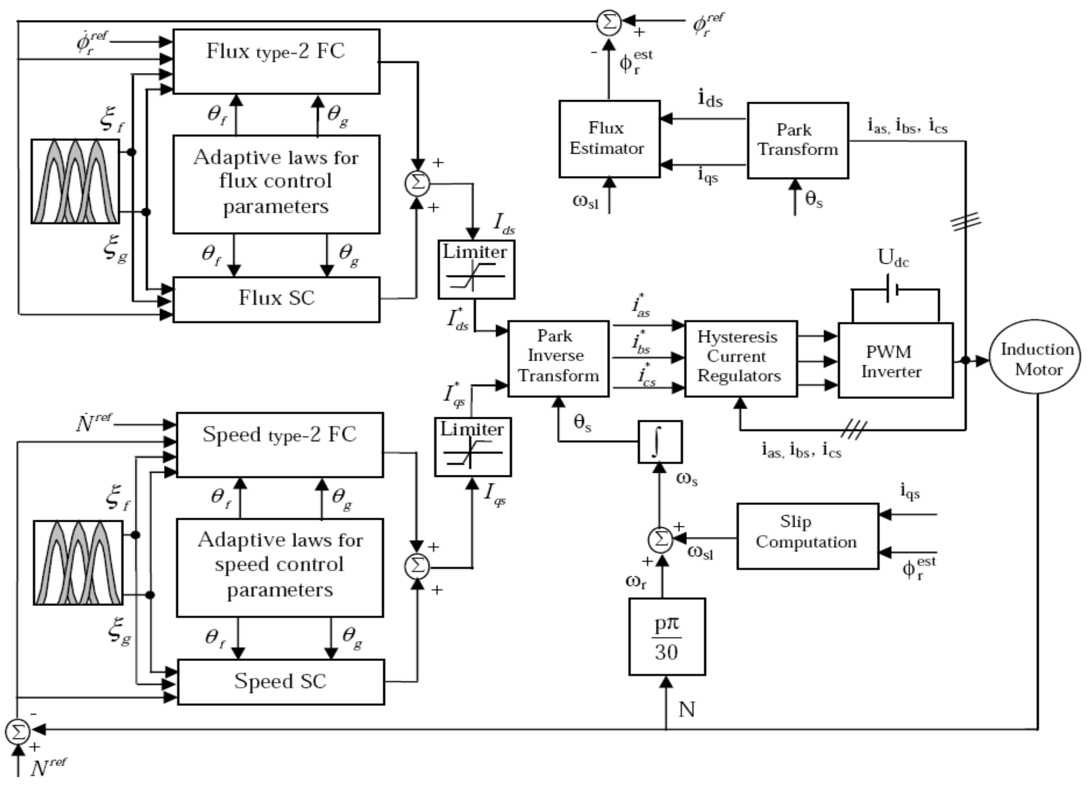


Figure 6.1: Input interval valued MF.

6.4 Type-2 fuzzy logic system design

Figure 6.1 depicts the structure of a type-2 FLS; it is quite similar to a type-1 FLS, the only difference being that the antecedent and/or consequent sets in a type-2 FLS are type-2. There are five principal parts in a type-2 FLS: fuzzifier, rule base, inference engine, type-reducer and defuzzifier. The type-2 fuzzy rule base consists of a collection of $IF - THEN$ rules in the following form:

$$R^i \ : \ IF \ x_1 \ is \ \tilde{F}_1^i \ and ... and \ x_n \ is \ \tilde{F}_n^i, \ \ THEN \ y \ is \ \tilde{G}^i \tag{6.15}$$

where $\tilde{F}_j^i$ are antecedent type-2 sets ($j = 1, 2, ..., n$), $y \in Y$ is the output, G^i are consequent type-2 sets, and $i = 1, 2, ..., M$, and M is the total number of rules.

In an interval type-2 FLS with meet under minimum or product $t - norm$, the firing interval $W^i = [\underline{w}^i, \bar{w}^i]$ of the rule is an interval type-1 set, which is determined by its left-most and right-most points $\underline{w}^i$ and $\bar{w}^i$ such that:

$$\underline{w}^i = \underline{u}_{\tilde{F}_1^i}(x_1) * \cdots * \underline{u}_{\tilde{F}_n^i}(x_n) \tag{6.16}$$

$$\bar{w}^i = \bar{\mu}_{\tilde{F}_1^i}(x_1) * \cdots * \bar{\mu}_{\tilde{F}_n^i}(x_n) \tag{6.17}$$

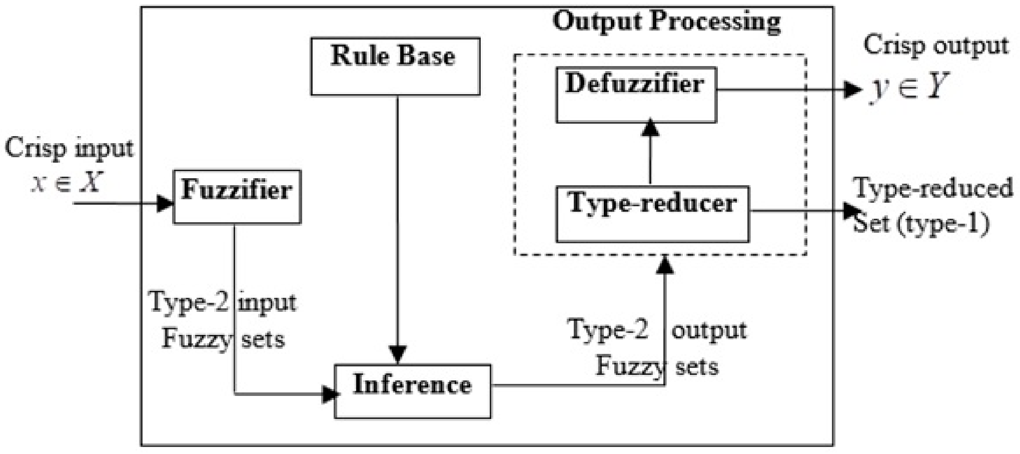


Figure 6.2: The structure of a type-2 FLS, with its two outputs: type reduced set and crisp defuzzified value.

Type reduction was proposed by *Karnik* and *Mendel* [4]. It is an extension of type-1 defuzzification method. There exist many kinds of type-reduction [5, 6, 7, 8], and in our work we propose the introduction of MEKM type-reduction method proposed in [4] in the control framework. Let's call centre of sets (cos) the output result of the type reduction process. In this paper, we propose to apply type-2 FLSs to obtain the estimates $\hat{f}$ and $\hat{g}$ for each of flux and speed controllers. The antecedent type-2 membership functions $\mu_{\tilde{F}_j^i}$ will be fixed as shown in Figure 6.2, and the consequent sets which are the adjustable parameters will be considered as type-1 centroids. Since $\hat{f}$ and $\hat{g}$ are type-2 fuzzy logic systems, their output sets (type-reduced sets) $\tilde{F}_{\cos}$ and $\tilde{G}$cos calculated by the centre of sets method will be given as follows:

$$\hat{F}_{\cos}(\theta_f^1,...,\theta_f^M,W_f^1,...,W_f^M)=\int\limits_{\theta_f^1}...\int\limits_{\theta_f^M}\int\limits_{w_f^1}...\int\limits_{w_f^M}1/\frac{\sum\limits_{i=1}^{M}w_f^i\theta_f^i}{\sum\limits_{i=1}^{M}w_f^i} \tag{6.18}$$

$$\hat{G}_{\cos}(\theta_g^1,...,\theta_g^M,W_g^1,...,W_g^M)=\int\limits_{\theta_g^1}...\int\limits_{\theta_g^M}\int\limits_{w_g^1}...\int\limits_{w_g^M}1/\frac{\sum\limits_{i=1}^{M}w_g^i\theta_g^i}{\sum\limits_{i=1}^{M}w_g^i} \tag{6.19}$$

where w_f^i and w_g^i are the firing intervals corresponding to the i^{th} rule of the type-2 FSs $\hat{f}$ and $\hat{g}$, respectively, θ_f^i and θ_g^i are the free parameters of the type-2 FSs θ_f^i and θ_g^i, respectively. Since each set on the right-hand side of (6.18) and (6.19) is an interval type-1 set, hence $\tilde{F}_{\cos}$ and $\tilde{G}_{\cos}$ are also an interval type-1 sets. So, to find $\tilde{F}_{\cos}$ and $\tilde{G}_{\cos}$, we just need to compute the two end points of these intervals.

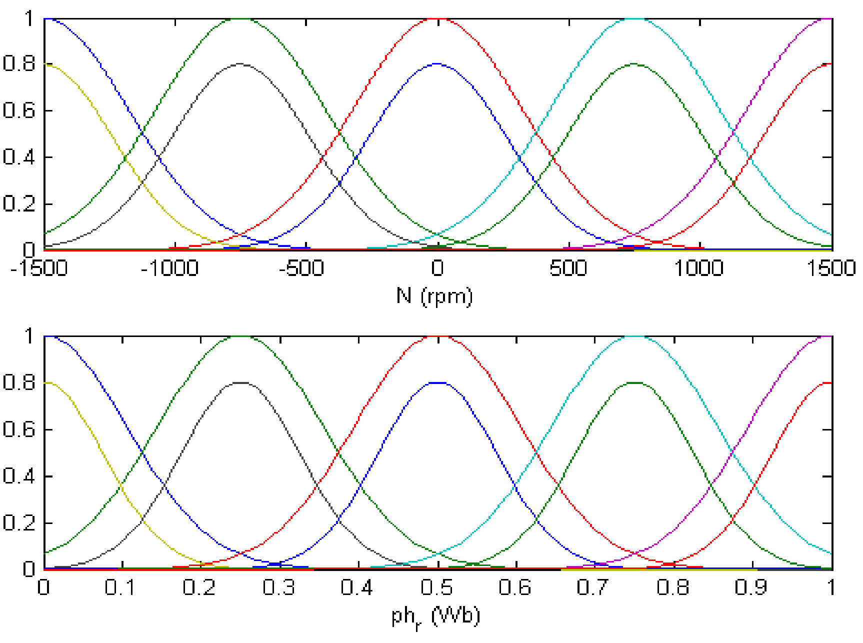


Figure 6.3: Antecedent type-2 membership functions.

In [9, 10] only one type-1 vector of fuzzy basis functions $\underline{\xi}\left(\underline{x}\right)$ was used to obtain $\hat{f}$ and $\hat{g}$ simultaneously. Unfortunately, this does not carry over to type-2 FLSs, due to that type-reduction will give for each $\hat{f}$ and $\hat{g}$ two different vectors of fuzzy basis functions. Next we will show how type-2 fuzzy logic will associate with every $\hat{f}$ and $\hat{g}$ a self vector of fuzzy basis functions $\xi_f\left(x\right)$ and $\xi_g\left(x\right)$, respectively. For any value of $\hat{f} \in \hat{F}_{\cos}$ and for any value of $\hat{g} \in \hat{G}_{\cos}$, $\hat{f}$ and $\hat{g}$ can be represented as:

$$\hat{f} = \frac{\sum\limits_{i=1}^{M} w_f^i \theta_f^i}{\sum\limits_{i=1}^{M} w_f^i}, \quad \hat{g} = \frac{\sum\limits_{i=1}^{M} w_g^i \theta_g^i}{\sum\limits_{i=1}^{M} w_g^i} \tag{6.20}$$

The maximum values of $\hat{f}$ and $\hat{g}$ are $\hat{f}_r$ and $\hat{g}_r$ respectively, and the minimum values of $\hat{f}$ and $\hat{g}$ are $\hat{f}_l$ and $\hat{g}_l$ respectively. In the centre of sets (cos)-type reduction method *Karnik* and *Mendel* [7] have shown that the two end points of $\tilde{F}_{\cos}$, $\hat{f}_l$ and $\hat{f}_r$ depend only on a mixture of $\underline{w}_f^i$ or $\bar{w}_f^i$ values, since $w_f^i \in [\underline{w}_f^i, \bar{w}_f^i]$. In the same manner, the two end points of $\tilde{G}_{\cos}$, $\hat{g}_l$ and $\hat{g}_r$ depend only on a mixture of $\underline{w}_g^i$ or $\bar{w}_g^i$ values, since $w_g^i \in [\underline{w}_g^i, \bar{w}_g^i]$. In this case, $\hat{f}_l$, $\hat{f}_r$, $\hat{g}_l$ and $\hat{g}_r$ can each be represented as a vector of fuzzy basis functions (FBF) expansion, i.e.,

$$\hat{f}_l = \frac{\sum\limits_{i=1}^{M} w_{fl}^i \theta_f^i}{\sum\limits_{i=1}^{M} w_{fl}^i} = \sum_{i=1}^{M} \theta_f^i \xi_{fl}^i = \underline{\theta}_f^T \underline{\xi}_{fl}(\underline{x}) \tag{6.21}$$

where w_{fl}^i is the firing strength membership (either $\underline{w}_f^i$ or $\bar{w}_f^i$) contributing to the left-most point $\hat{f}_l$, s.t:

$$\xi_{fl}^i = \frac{w_{fl}^i}{\sum\limits_{i=1}^{M} w_{fl}^i} \tag{6.22}$$

are the components of the first FBF vector $\underline{\xi}_{fl}(\underline{x})$ of $\hat{f}$, i.e., $\underline{\xi}_{fl}^T(\underline{x}) = [\xi_{fl}^1, ..., \xi_{fl}^M]$ and $\underline{\theta}_f^T = [\theta_f^1, ..., \theta_f^M]$ is the parameter vector of the type-2 FLS $\hat{f}$. Similarly,

$$\hat{f}_r = \frac{\sum\limits_{i=1}^{M} w_{fr}^i \theta_f^i}{\sum\limits_{i=1}^{M} w_{fr}^i} = \sum_{i=1}^{M} \theta_f^i \xi_{fr}^i = \underline{\theta}_f^T \underline{\xi}_{fr}(x) \tag{6.23}$$

where w_{fr}^i denotes the firing strength membership grad contributing to the right-most point $\hat{f}_r$ and:

$$\xi_{fr}^i = \frac{w_{fr}^i}{\sum\limits_{i=1}^{M} w_{fr}^i} \tag{6.24}$$

are the components of the second FBF vector $\underline{\xi}_{fr}(\underline{x})$ of $\hat{f}$, i.e., $\underline{\xi}_{fr}^T(\underline{x}) = [\xi_{fr}^1, ..., \xi_{fr}^M]$. Similarly, we have for $\hat{g}$ two FBF vectors $\underline{\xi}_{gl}^T(\underline{x}) = [\xi_{gl}^1, ..., \xi_{gl}^M]$ and $\underline{\xi}_{gr}^T(\underline{x}) = [\xi_{gr}^1, ..., \xi_{gr}^M]$ such that:

$$\hat{g}_l = \frac{\sum_{i=1}^{M} w_{gl}^i \theta_g^i}{\sum_{i=1}^{M} w_{gl}^i} = \sum_{i=1}^{M} \theta_g^i \xi_{gl}^i = \underline{\theta}_g^T \underline{\xi}_{gl}(\underline{x}) \tag{6.25}$$

$$\hat{g}_r = \frac{\sum_{i=1}^{M} w_{gr}^i \theta_g^i}{\sum_{i=1}^{M} w_{gr}^i} = \sum_{i=1}^{M} \theta_g^i \xi_{gr}^i = \underline{\theta}_g^T \underline{\xi}_{gr}(x) \tag{6.26}$$

where $\underline{\theta}_g^T = [\theta_g^1, ..., \theta_g^M]$ is the parameter vector of the type-2 FLS $\hat{g}$,

$$\xi_{gl}^i = \frac{w_{gl}^i}{\sum_{i=1}^{M} w_{gl}^i}, \quad \xi_{gr}^i = \frac{w_{gr}^i}{\sum_{i=1}^{M} w_{gr}^i} \tag{6.27}$$

To obtain a crisp outputs from the type-2 FLSs $\hat{f}$ and $\hat{g}$, we must defuzzify the type-reduced sets $\tilde{F}_{\cos}$ and $\tilde{G}_{\cos}$. Since these type reduced sets are interval sets, therefore, the defuzzified output of $\hat{f}$ will be the average of $\hat{f}_l$ and $\hat{f}_r$, and the defuzzified output of $\hat{g}$ will be the average of $\hat{g}_l$ and $\hat{g}_r$ i.e.,

$$\hat{f} = \frac{\hat{f}_l + \hat{f}_r}{2}, \quad \hat{g} = \frac{\hat{g}_l + \hat{g}_r}{2} \tag{6.28}$$

Replacing (6.21), (6.23), (6.25), and (6.26) into (6.28) we obtain:

$$\hat{f} = \frac{\underline{\theta}_f^T \underline{\xi}_{fl} + \underline{\theta}_f^T \underline{\xi}_{fr}}{2} = \underline{\theta}_f^T \left[\frac{\underline{\xi}_{fl} + \underline{\xi}_{fr}}{2}\right] = \underline{\theta}_f^T \underline{\xi}_f() \tag{6.29}$$

$$\hat{g} = \frac{\underline{\theta}_g^T \underline{\xi}_{gl} + \underline{\theta}_g^T \underline{\xi}_{gr}}{2} = \underline{\theta}_g^T \left[\frac{\underline{\xi}_{gl} + \underline{\xi}_{gr}}{2}\right] = \underline{\theta}_g^T \underline{\xi}_g() \tag{6.30}$$

where $\underline{\xi}_f = \left(\underline{\xi}_{fl} + \underline{\xi}_{fr}\right)/2$ is the average FBF vector of $\hat{f}$ and $\underline{\xi}_g = \left(\underline{\xi}_{gl} + \underline{\xi}_{gr}\right)/2$ is the average FBF vector of $\hat{g}$. In order to compute $\underline{\xi}_{fl}$ and $\underline{\xi}_{fr}$ ($\underline{\xi}_{gl}$ and $\underline{\xi}_{gr}$), we need to compute w_{fl}^i and w_{fr}^i,$i = 1, 2, ..., M$ (w_{gl}^i and w_{gr}^i,$i = 1, 2, ..., M$). This can be done using the computational method given in [11, 12]. The crisp values of $\hat{f}$ and $\hat{g}$ can be obtained either by (6.28), or by using the FBFs $\underline{\xi}_f(x)$ and $\underline{\xi}_g(x)$, respectively, as shown in (6.29) and (6.30). Recall that the above method is applied independently to estimate the functions $\hat{f}$ and $\hat{g}$ for both flux and speed models.

6.5 Adaptive control structure

In this section, we will develop the IAC-based type-2 fuzzy controller with supervisory control scheme. Applying (6.11) to (6.14) and after straightforward manipulation, we obtain the error equation:

$$e = -ke + \left(\hat{f}(x/\underline{\theta}_f) - f(x)\right) + \left(\hat{g}(x/\underline{\theta}_g) - g(x)\right) u_c - d \tag{6.31}$$

we know that there exists a unique positive constant p which satisfies the Lyapunov equation:

$$-kp - pk = -q \tag{6.32}$$

where q is an arbitrary positive constant. Let $V_e = \frac{1}{2}pe^2$, then using (6.31) and (6.32) we have:

$$\dot{V}_e = -\frac{1}{2}qe^2 + pe[\left(\hat{f}(x/\underline{\theta}_f) - f(x)\right) + \left(\hat{g}(x/\underline{\theta}_g) - g(x)\right) u_c - d] \tag{6.33}$$

then, we must have $\dot{V}_e \leq 0$ when V_e is greater than a large constant $\tilde{V}$, however, from (6.33) we see that it is very difficult to design the u_c such that the last term of (6.33) is less than zero. We solve this problem by appending another control term (supervisory control) u_s to the u_c. So, the final control becomes

$$u = u_c + u_s \tag{6.34}$$

this additional control term is called a supervisory control. The purpose of this supervisory control u_s is to force $\dot{V}_e \leq 0$ when $V_e \geq \tilde{V}$. Substituting (6.34) into (6.11) and after some manipulations to force $\dot{V}_e$ to be negative, we obtain the following supervisory control:

$$u_s = \begin{cases} sgn(\underline{e}^T P \underline{b}_c) \frac{1}{g_L(\underline{x})} \left[\left|\hat{f}\right| + f^U + |\hat{g}u_c| + \left|g^U u_c\right| + d_m\right] \\ if \quad V_e \geq \tilde{V} \\ 0 \\ if \quad V_e < \tilde{V} \end{cases} \tag{6.35}$$

where $f^U(\underline{x})$ an upper bound of f, $g^U(\underline{x})$ and $g_L(\underline{x})$ are an upper and an lower bounds of g, respectively and d_m is the upper bound of a perturbation d. Next, we replace $\hat{f}$ and $\hat{g}$ by the type-2 fuzzy logic systems given in (6.29) and (6.30). In order to adjust the parameters $\underline{\theta}_f$ and $\underline{\theta}_g$ in the type-2 fuzzy logic systems, we derive the following adaptive laws:

$$\dot{\underline{\theta}}_f = -\gamma_1 ep\underline{\xi}_f(x) \tag{6.36}$$

$$\dot{\underline{\theta}}_g = -\gamma_2 ep\underline{\xi}_g(x)u_c \tag{6.37}$$

6.6 Simulation results

To prove the effectiveness of the developed controller, simulations on an IM have been carried out. The overall control system which has been simulated is shown in Figure 6.3. The three-phase induction motor is characterized by the parameters shown in Table 6.1. The current-controlled inverter is fed by 514 V DC voltages, and the hysteresis bandwidth of stator current is fixed to 0.1A. We admit that the influence of the flux on the dynamic of the speed is neglected. In this case, the bounds f^U, g^U and g_L for each controller are chosen in according with (6.9) and (6.10) as follows:

- Speed controller bounds:

$$f^u = \frac{f}{J}N, \;\; g^u = 1.1g', \;\; g_L = 0.9g' \;\; with \;\; g' = \frac{30pL_m\phi_r^{ref}}{\pi J L_r} \tag{6.38}$$

- Flux controller bounds:

$$f^u = 1.1\frac{\phi_r^{est}}{T_r}, \;\; g^u = 1.1g'', \;\; g_L = 0.9g'' \;\; with \;\; g'' = \frac{L_m}{T_r} \tag{6.39}$$

The desired flux and speed tracking are involved with regulator coefficients tuned by trial and error to the values given in Table 6.2. The speed response and the speed reference ($1000rpm$) are depicted in Figure 6.4(a), which shows good performances in tracking and an excellent load charge rejection caused by the applied load torque shown in Figure 6.4(b). The corresponding electromagnetic torque response is generated, as shown in Figure 6.4(c), to compensate the load charge and to keep speed regulation, see also current of phase an in Figure 6.4(d). Figure 6.4(e)-(f), it can be seen that the flux is well oriented along the $d-$axis of the synchronous frame and controlled to have a constant value.

In spite of these sudden changes, the controllers continue to work very well, where we see in Figure 6.5(a)-(b) the speed and flux tracking errors converge to zero in the steady

Table 6.1: Parameters of the used machine.

Induction Motor 1.5 kW, 220/380 V, 50 HZ	
R_r	4.750 Ω
R_s	4.750 Ω
L_r	0.374 H
L_s	0.374 H
L_m	0.158 H
B	0.070 $N\ m\ s$
J	0.021 $Kg\ m^2$
P	2

Table 6.2: Regulator parameters.

	q	k	p	M_f	M_g	γ_f	γ_g	V_t
Flux controller	10	80	0.00625	15	4	3.75	0.4	0.0001
Speed controller	10	50	0.1	400	640	4	6.4	0.1

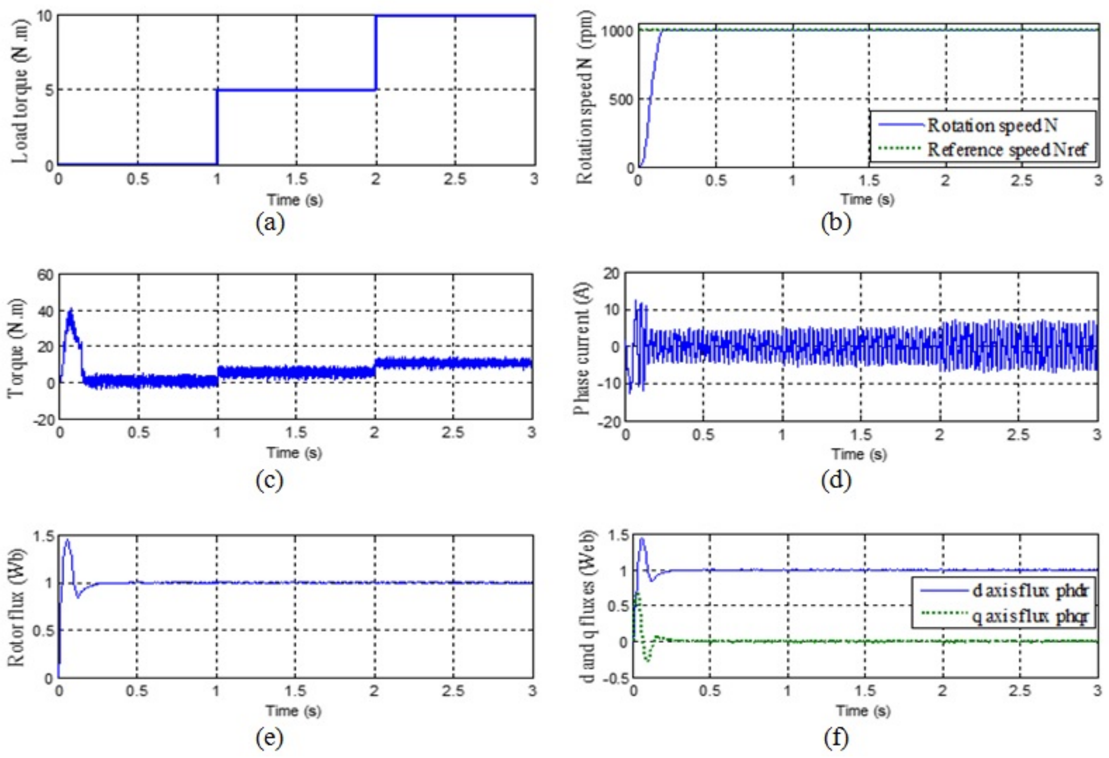


Figure 6.4: Speed responses and flux responses of IM, (a) Motor speed, (b) Applied load torque, (c) Electromagnetic torque, (d) Phase current, (e) Rotor flux, (f) d and q fluxes.

states. It's clear from Figure 6.5(c)-(d) that the supervisory control actions of the two controllers were activated many times in order to stabilize the closed-loop system.

Simulation results show that the speed and flux type-2 AFC yield excellent dynamic performances for an induction motor drive and they assure the insensitivity to the working conditions. In order to evaluate the insensitivity to the parameters variations of the proposed controller, we keep the same load torque shown in Figure 6.4(b) and at the same time.

We increase the rotor resistance values with 50% at time 1 sec and 100% at time 2 sec as shown in Figure 6.6(a). It is clearly shown from Figure 6.6(b)-(c) that the speed and its corresponding tracking errors converge to zero and stay small. Note also that fluxes responses are very satisfactory as shown in Figure 6.6(d)-(f). To evaluate the performances of our approach, we will compare it with two other techniques: an optimized PID controller and a type-1 version of the proposed method. The parameters of the PID controller (K_p, K_d and K_i) are optimized by particle swarm optimization (PSO) technique as explained in [13]. During optimization simulations, swarm populations are set to 20 particles and its coefficients w, c_1 and c_2 are set to 0.8, 1 and 1.5, respectively [13].

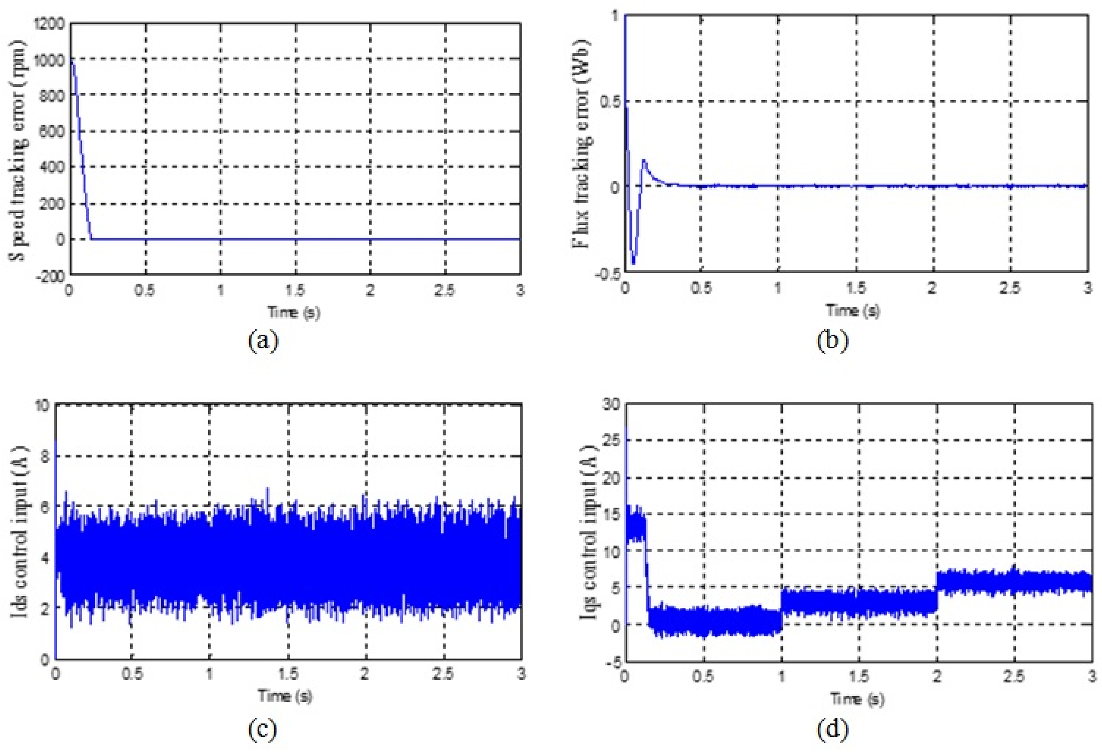


Figure 6.5: Errors of tracking of speed and flux with their corresponding command laws, (a) Error of speed (b) Error of flux (c) Stator q current (d) Stator d current.

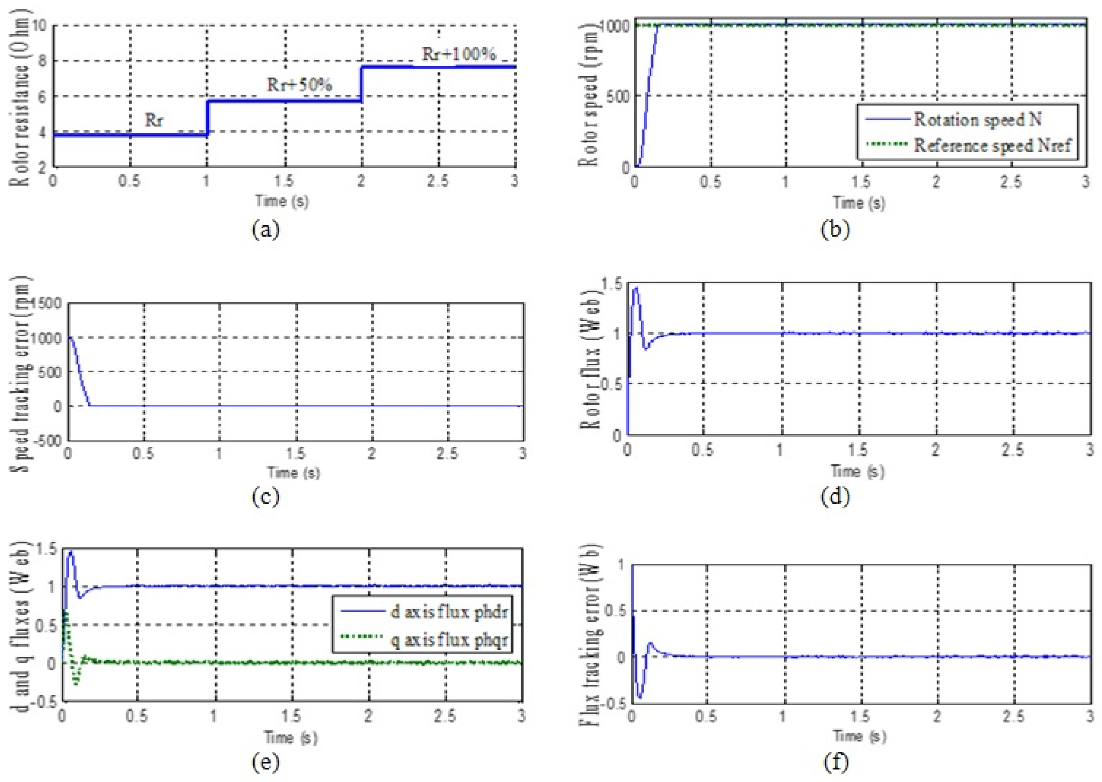


Figure 6.6: Tests of robustness under parametric variations, (a) Rotor resistance variation, (b) Speed of the rotor, (c) Error of the speed, (d) Rotor flux, (e) *d* and *q* fluxes, (f) Flux tracking error.

The type-1 version of our type-2 adaptive fuzzy controller is obtained just by eliminating the lower membership functions form the type-2 membership functions represented in Figure 6.4 which gives us a type-1 membership functions, and then the type reducer bloc is eliminated as shown in Figure 6.2. Note that the presented type-2 fuzzy adaptive controller gives more accurate rotor speed compared to the PID and the type-1 fuzzy controllers as shown in Figure 6.7. For quantitative comparison purposes, the performance of the proposed method is evaluated by using the MSE (mean square error) criterion between the reference speed N^{ref} and the actual rotor speed N as follows:

$$MSE(speed) = \frac{1}{K}\sum_{i=1}^{K}\left(N - N^{ref}\right)^2 \tag{6.40}$$

We show in Table 6.3 the corresponding MSEs (speed) for the three controllers where we confirm the superiority with respect to precision of the proposed controller over the PID and type-1 fuzzy controller. To compare the amount of energy needed by the three controllers, let's define the MSE of the torque with respect to zero torque as follows:

$$MSE(torque) = \frac{1}{K}\sum_{i=1}^{K}(T_e - 0)^2 = \frac{1}{K}\sum_{i=1}^{K}{T_e}^2 \tag{6.41}$$

The formula (6.41) can be used as a measure of the control effort. According to (6.41), we see in Table 6.3 that the motor torque reach its minimum values with our type-2 fuzzy controller, which is a proof that our proposed controller can achieve better performances (MSE speed) with minimum energy (MSE torque). To confirm more the efficiency of the proposed method, let's check it with a very big challenge which is the zero speed (low speed) tracking. In this case, a reference speed of 20 rpm is applied, and the obtained results are depicted in Figure 6.8, where we clearly see the high performances of the type-2 fuzzy adaptive controller over its type-1 counterpart and the PID controller (very big ripples in the case of PID and type-1 fuzzy controllers but small ripples with the type-2 fuzzy controller). This fact is confirmed numerically in Table 6.4.

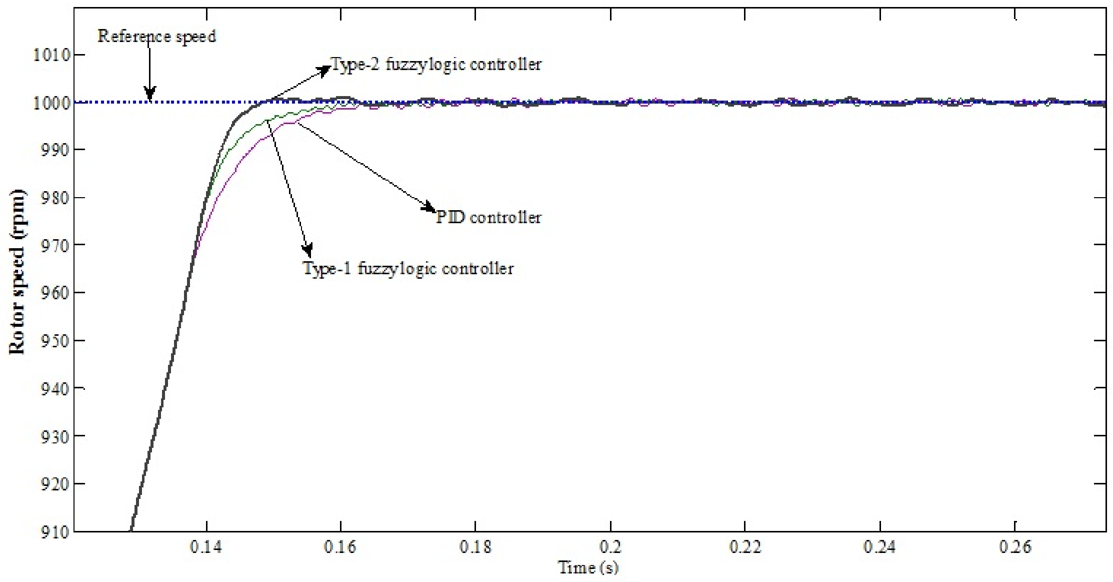


Figure 6.7: Superposition of the speed responses of PID, Ordinary FLC and IVFLC controllers.

Table 6.3: Comparison between PID, ordinary FLC and IVFLC.

	PID	Type-1	Typ-2
MSE (Speed)	6.7135e+06	2.6960e+06	8.3731e+05
MSE (Torque)	8.8842e+06	8.1005e+06	8.0135e+05

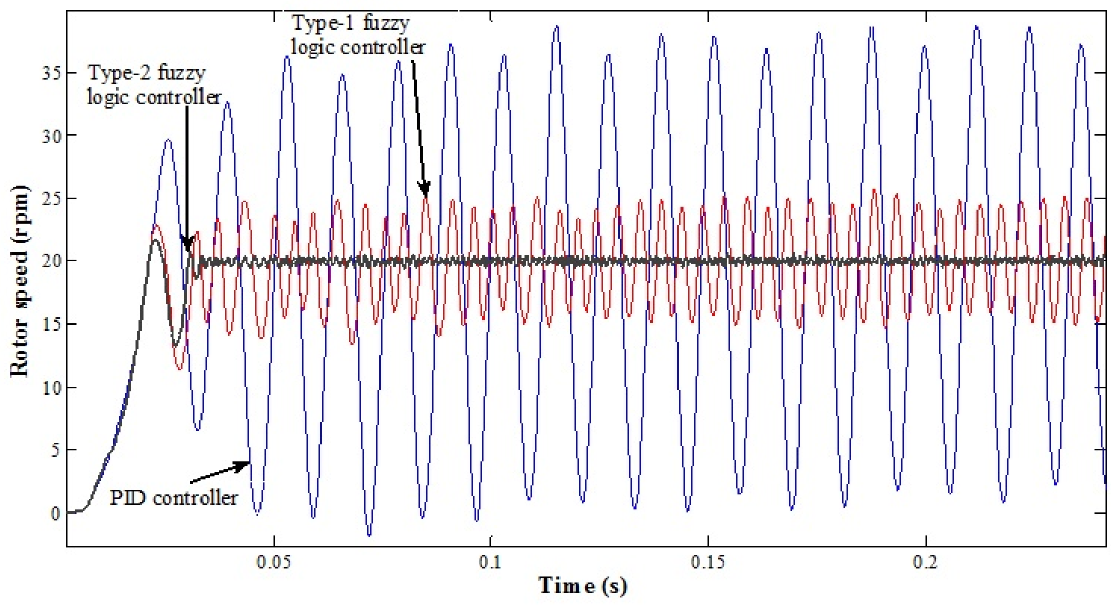


Figure 6.8: Superposition of the speed responses of PID, ordinary FLC and IVFLC in low speed case.

Table 6.4: Comparative table in low speed.

	PID	Type-1	Typ-2
MSE (low Speed)	3.6215e+04	1.7925e+04	1.0466e+03

6.7 Conclusion

In this chapter, an indirect adaptive fuzzy controller based on type-2 fuzzy systems with a supervisory controller has been designed and applied to the control of an induction motor drive. Based on Lyapunov synthesis approach, the free parameters of the type-2 fuzzy adaptive controller can be tuned on-line by an adaptive law. It has been shown that the proposed controller can provide the properties of insensitivity to uncertainties and external disturbances. Simulation results showed that our proposed approach is very effective to control an induction motor. The superiority of our algorithm over other techniques like PID control and the type-1 fuzzy adaptive controller was confirmed by a short comparative study.

Bibliography

[1] Bounouara N., Ghanai M., Medjghou A., and Chafaa K. Stable and robust control strategy using interval-valued fuzzy systems. *International Journal of Applied Power Engineering (IJAPE).* 9(3): 205-217, 2020. doi: 10.11591/ijape.v9.i3.pp205-217

[2] Nguyen T.T. The neural network-combined optimal control system of induction motor. *International Journal of Electrical and Computer Engineering.* 9(4): 2513-2522, 2019.

[3] Kim Y.C., Song H.B., Cho M.T., and Moon S.H. A study on direct vector control system for induction motor speed control. *Embedded and Multimedia Computing Technology and Service* 181: 599-612, 2012.

[4] Wu D., and Mendel J.M. Enhenced karnik-mendel algorithms. *IEEE Transaction on Fuzzy Systems.* 17(4): 923-934, 2009.

[5] Salaken S.M., Khosravi A., and Nahavandi S. Modification on enhanced Karnik-Mendel algorithm. *Expert Systems with Applications.* 65: 283-291, 2016.

[6] TaiK A., El-Sayed R., Biglarbegian M., Gonzalez C.I., Castillo O., and Mahmud S. Review of recent type-2 fuzzy controller applications. *Algorithms.* 9(2): 1-19, 2016.

[7] Benbrahim M., Essounbouli N., Hamzaoui A., and Betta A. Adaptive Type-2 Fuzzy Sliding Mode Controller for SISO Nonlinear Systems Subject to Actuator Faults. *International Journal of Automation and Computing.* 10(4): 335-342, 2013.

[8] Liu X., Mendel J.M., and Dongrui W. Study on enhanced Karnik-mendel algorithms: Initialization explanations and computation improvements. *Information sciences.* 184(1): 75-91, 2012.

[9] Wang L.X. (1994).*Adaptive Fuzzy Systems and control: Design and stability Analysis.* Englewood Cliffs, NJ: Prentice-Hall.

[10] Hamzaoui A., Essoumbouli N., Benmahammed K., and Zaytoon J. State observer based robust adaptive fuzzy controller for nonlinear uncertain and perturbed systems. *IEEE transaction on fuzzy systems.* 34(2): 942-950, 2004.

[11] Chafaa K., Saidi L., Ghanai M., and Benmahammed K. Direct adaptive type-2 fuzzy control for nonlinear systems. *International Journal of Computaional Intelligence and Applications.* 6(3), 389-411, 2006.

[12] Chafaa K., Saidi L., Ghanai M., and Benmahammed K. Indirect adaptive interval type-2 fuzzy control for unknown nonlinear systems. *International Journal of Modelling, Identification and Control.* 2(2): 106-119, 2007.

[13] Laamari Y., Chafaa K., and Athamena B. Particle swarm optimization of an extended Kalman filter for speed and rotor flux estimation of an induction motor drive. *Electrical engineering.* 97: 129-138, 2015.

GENERAL CONCLUSION

The work presented in this dissertation included three major contributions:
1) The synthesis of observers for a class of uniformly observable nonlinear systems with sampled output. We have shown that the state of the system can be estimated with a continuous discrete-time high gain observer whereas the observer gain has been optimized by metaheuristic optimization. We have proved the convergence of this observer in the disturbance and the uncertain environments.
2) A metaheuristic optimization method using particle swarm optimization for the adjustment of PD and PID gains. Two PSO approaches were used: PSOCIW and PSOVIW. In this technique, the algorithms were combined with PID and PD controllers for the purpose of improving controller effectiveness. The approaches were designed to optimize offline controller parameters. After that, optimal optimized parameters were injected into the online control loop. The introduced algorithms were validated on the control of a two link robot manipulator.
3) Apply an indirect adaptive fuzzy controller based on type-2 fuzzy systems with a supervisory controller to the control of the nonlinear system.

In **chapter I**, we have described the nonlinear systems and have presented the basic concepts and theorems related to Lyapunov which garantee systems stability.

In **chapter II** we have described the continuous-time high gain and the continuous discrete-time high gain observers for a class of uniformly observable MIMO nonlinear systems with the presence of disturbances and uncertainties.

In **chapter III** we have briefly described the principle of genetic algorithms, particle swarm optimization algorithm, and biogeography-based optimization algorithm, which allow providing a sufficiently good solution to an optimization problem.

In **chapter IV** the observer gain has been optimized by biogeography-based optimization for the estimation of the states used in feedback linearization controller for the control of two link robot manipulator. Simulation results showed that the optimal estimations obtained by BBO are much better than the best solutions obtained by PSO and GA algorithms.

In **chapter V** two strategies of PSO algorithm has been used for the adaptation of PI and PID free parameters: PSOVIW and PSOCIW. The obtained results with PSOVIW strategy show its superiority and efficiency over PSOCIW.

In **chapter VI**, an indirect adaptive fuzzy controller based on type-2 fuzzy systems with a supervisory controller has been presented in order to control the dynamical nonlinear system induction motor drive. The indirect fuzzy adaptive controller was constructed from a collection of fuzzy IF-THEN rules whose parameters have been adjusted on-line by an appropriate laws adaptation. It has made according to the synthesis of Lyapunov to guarantee the stability of closed-loop system. Simulation results of an induction motor have been performed to study the efficiency of the proposed method and the superiority of our algorithm. Comparisons with PID control and the type-1 adaptive controller show that the results are very efficient and achieve good performances.

PERSPECTIVE

Our future perspectives are to extend this work for unknown nonlinear systems when f and g are not known.

1) Use fuzzy estimation for the unknown functions f and g.

2) Apply machine learning to enhance the obtained performances.