POLITECNICO DI MILANO

School of Industrial and Information Engineering

Master of Science in Automation and Control Engineering - Ingegneria dell'Automazione

Command Filtered Adaptive Backstepping Control with Disturbance Estimation by Least Squares Support Vector Regression

Supervisor:
Formentin Simone

Student:

Cuoghi Ludovico 913007

Academic Year: 2020/2021

Abstract

In questo lavoro, è stato sviluppato un controllo Adaptive Backstepping con Command Filter con compensazione di disturbi per controllare una classe di sistemi non lineari di ordine elevato con parametri sconosciuti e affetti da disturbi esterni. L'obiettivo del controllo e di garantire la stabilità globale del sistema e il corretto tracking tra l'uscita del sistema e un segnale di riferimento. Il sistema di controllo si basa sul controllo Backstepping. L idea del controllo Backstepping e di suddividere il sistema controllato in sottosistemi più piccoli che sono ricorsivamente stabilizzati da alcune cosiddette funzioni stabilizzanti. Il calcolo delle derivate delle funzioni stabilizzanti, necessarie nelle leggi di controllo, richiede grande computazione nel caso di sistemi di ordine elevato e per questo motivo in questo lavoro è stato implementato un filtro Command Filter nell'algoritmo di controllo standard del backstepping. Il filtro Command Filter ha permesso di ottenere nuove leggi di controllo senza la necessità di calcolare le derivate delle funzioni stabilizzanti. Il nuovo controllo Backstepping con Command Filter e stato successivamente esteso con leggi adattative che hanno permesso la stima dei parametri incogniti nel sistema. Tuttavia, una degenerazione delle prestazioni di tracking e stata mostrata per il controllo di una classe di sistemi non lineari affetti da disturbi esterni. Per migliorare le prestazioni di tracciamento, e stata implementata una rigerssione con Support Vector Regression per identificare i disturbi a partire da campioni rumorosi. Infine questi modelli dei disturbi sono stati inclusi nella precedentemente sviluppata legge di controllo, in modo tale da poter compensare i disturbi originali. Le prestazioni di tracciamento del nuovo sistema di controllo sviluppato su una classe di sistemi non lineari affetti da disturbi, hanno mostrato risultati comparabili al caso di sistema non affetti da disturbi.

Abstract (english)

In this work, a Command Filtered Adaptive Backstepping Control with disturbance compensation has been developed to control a class of high order nonlinear systems in a strict feedback form with unknown parameters and affected by external disturbances. The objective of the control system was to guarantee global system stability and the tracking between the output of the system and a reference signal. The control system is based on Backstepping control, which is a recursive approach to control nonlinear systems where the controlled system is divided into smaller subsystems which are recursively stabilized by some so-called stabilizing functions. The calculation of the derivatives of the stabilizing functions, necessary in the control laws, require heavy computation in the case of high order systems and for this reason, in this work a command filter has been implemented in the standard backstepping control algorithm. The developed command filtered backstepping control was extended with adaptive laws that allowed the estimations of the unknown parameters in the systems. However, when one tried to control a class of nonlinear systems affected by external disturbances, a degeneration in the performance was shown. To improve the tracking performance, function identification models by a Support Vector Machine variant known as Least Squares Support Vector Regression were used as regression models of the disturbances and then implemented in the control system to preform rejection of the original disturbances. To perform the regression, the regression models have been trained with a noisy data set of the disturbances. The trained models by Least Squares Support Vector Regression could approximate the unknown disturbances with satisfying accuracy. Finally, a novel Command Filtered Adaptive Backstepping control system was developed by using the regression models by LS-SVR to reject the external disturbances. The simulations showed that the tracking performance of the novel developed control system implemented to a nonlinear system affected by external disturbances are comparable to the case where no external disturbances were added to the system.

Contents

List of Figures

Chapter 1

Introduction

In the past decades, numerous approaches have been proposed for the design of nonlinear control systems. Among these, adaptive control has been widely used as a design methodology that can stabilize nonlinear systems affected by uncertainties or in presence of unknown parameters [1][2]. The main idea of Adaptive Control is to adjust on-line the control parameters in the case where the controlled system has parameters which are unknown or vary during the execution. A lot of researchers have presented several important results on Adaptive Control and among these, Model Reference Adaptive Control has been largely used when the aim of the control system was the tracking between the output of the system and the output of a reference model [3]. In order to achieve the tracking, Model Reference Adaptive Control systems have tuning parameters that are updated on-line based on the error between the output of the plant and the output of the reference model. When using a MRAC control system, some restrictions such as matching conditions are necessary in order guarantee the global stability [4]. To overcome these restrictions, backstepping control and adaptive backstepping control have been used to control nonlinear systems [2]. The main idea behind backstepping control is to recursively derive some so called stabilizing functions as pseudo inputs of lower dimension subsystems of the original system until all the subsystems, and so the original system, are stabilized. Backstepping control can guarantee tracking performance and global stability for different classes of strict-feedback system and has been largely used in the field of electronics engineering, aerospace engineering etc. [5][6][7]. Nevertheless, some drawbacks of the backstepping approaches also exist, such as the explosion of complexity during the calculation of the stabilizing functions in the case where the plant is a high order system ($n > 2$). In an attempt to solve the explosion of complexity, methods like the Command Filter or Sliding Mode filters have been implemented in backstepping and adaptive backstepping control [8][9][10]. However, in the presence of external disturbances, these must be exactly known or exactly estimated to guarantee good performances when using a backstepping approach [11]. In different control applications, Multilayer feedforward neural network and radial basis function (RBF) neural network have been used to approximate different classes of function with suitable accuracy [11][12]. In the case of RBF networks, one has a curse of dimensionality in the number of parameters that needs to be defined [13][14]. A promising technique for function approximation and pattern classification called Support Vector Machine has been recently used to overcome

the overfitting problem and the curse of dimensionality of the RBF in nonlinear control [15][16][17]. Support Vector Machine is a kernel approach derived from Neural Networks and originally used as a classifier that maps input data to a higher dimensional space where an optimal separating hyperplane can be derived, in contrast to a more complex and heavy computing nonlinear function in the original input space [18]. A regression variant of the Support Vector Machine called Support Vector Regression was proposed by Vladimir N. Vapnik and his colleagues [13][14]. The training process of a Support Vector Machine, and so Support Vector Regression, is characterized by the formulation of a convex optimization problem, which so also overcomes the problem of local minima suffered by classical neural network approaches and only few tuning parameters needs to be defined, in contrast to the curse of dimensionality of a standard RBF method.
This work aims to design an adaptive controller for a class of nonlinear systems of second order and higher ($n > 2$) that guarantees good tracking performances and global stability in presence of unknown parameters and external disturbances. To achieve this, a Command Filtered approach will be implemented for its simplicity in the formulation and effectiveness into the standard Backstepping Control to overcome the discussed explosion of complexity problem. Then, an adaptive control approach will be implemented to the controller in order to estimate unknown systems parameters. To estimate the external disturbances in the controlled system, a Least Squares variant of the Support Vector Regression known as Least Squares Support Vector Regression will be used for its efficiency in the computation and simplicity in the formulation compared to the traditional SVR. Finally, the LS-SVR regression models will be implemented in the control laws of a novel Command Filtered Adaptive Backstepping controller to compensate the external disturbances affecting the system.
The organization of the thesis is as follows.
Chapter 2 presents a brief introduction to Adaptive Control and backstepping control. The explosion of complexity problem is shown and a comparison bewteen the standard backstepping and the commanded filtered backstepping, which aims to solve the considered problem, is proposed. Finally an adaptive extension of the command filtered backstepping is developed. Chapter 3 considers the case where external disturbances are added to the system previously controlled by the developed adaptive command filtered backstepping control system. The degeneration of the performance is shown and so Neural Networks and Support Vector Machine are introduced. Finally a Support Vector Machine method to estimate nonlinear functions called Support Vector Regression is discussed. Chapter 4 presents a novel command filtered adaptive backstepping control system based on the control system developed in Chapter 3 where the trained SVRs models are included in the control laws to reject the external disturbances. Chapter 5 shows simulations of the developed command filtered adaptive backstepping controllers in the case of a second order and third order nonlinear systems. The Conclusion and Future Works chapter presents comments about the results of the simulations chapter and discuss about the drawbacks and future works of the developed control systems.

Chapter 2

Nonlinear Systems Control

2.1 Introduction Adaptive Control

Adaptive Control is a control methodology that covers techniques in which the control parameters are adjusted on-line in order to guarantee the desired performance in the case where the dynamic parameters of the plant, linear or nonlinear, change in time and/or are unknown. In the control theory, the knowledge of the plant model parameters is necessary for the tuning of the controller, which aims to guarantee the desired performances on the plant. In the case where the plant parameters are unknown or uncertain, Adaptive control techniques can provide a real time tuning procedure of the controller parameters based on some available data of the plant [1]. A controlled system can present unknown parameters due to lack of knowledge during the modeling and also cases of real systems where the parameters change in time are common in the control field [1]. An example could be the case where the environmental conditions change or because a nonlinear systems is linearized and a change in operation condition will lead to a different linearized model [1]. In the discussed cases, adaptive control has been proven to be effective to deal with the small or large unknown parameters and changes in the plant during its functioning [1][3][19]. Moreover, depending on the system, adaptive controllers operate for a short or long time until the desired performances are guaranteed. The on-line information about the model plant can be processed in different ways, and this distincts the existing adaptive control algorithms. It is important to known that all the adaptive controllers are modeled as nonlinear systems, since the control parameters are based on the measurements of system variables through the adaptation loop [1]. Among the adaptive controllers Model Reference Adaptive Control has proven to be effective in different cases when the desired performance of the system could be specified in a target, or reference, model [3]. In the next section Model Reference Adaptive Control will be introduced.

2.1.1 Model Reference Adaptive Control (MRAC)

As discussed in the previous section, the specification of the desired performance is essential in the first presented adaptive controller design approach.
In Model Reference Adaptive Control, the desired performance of the system can be seen as the output of a so called reference model, which has the desired location of the closed loop poles of the plant. The main goal of the control will be to match the output of the plant in a closed loop and the output of the derived reference model [1].
An example of a Model Reference Adaptive Control scheme is shown in the following figure:

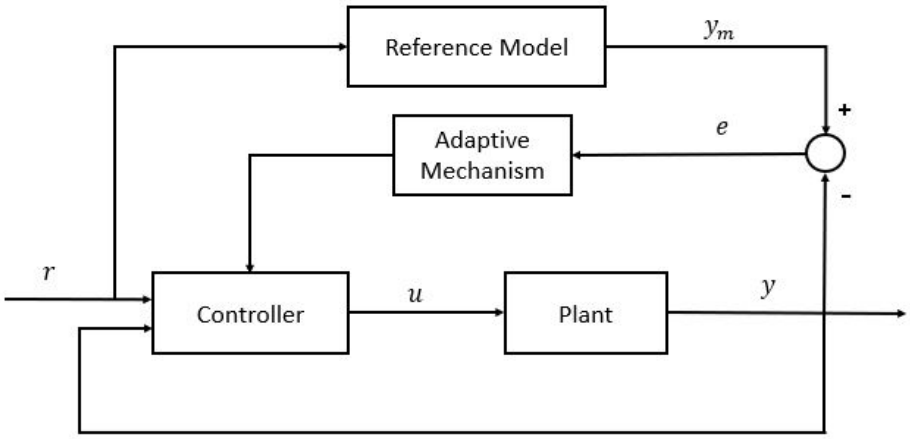

Figure 2.1: Model Reference Adaptive Control scheme

where y_m is the output of the reference model, y is the output of the plant, e is the error between the output of the plant and the output of the reference model, r is the reference signal and u is the control input.
The error e between the output of the reference model and the output of the plant is used as a real time information for the adaptation mechanism, which will adjust the control parameters in order to make the error between the two outputs asymptotically equal to zero. However, there exist restricting conditions that have be made on the controlled systems that limit the application of a Model Reference Adaptive Control approach [4]. An example is the assumption that the plant model has stable zeros in every situation, which in the discrete-time case is quite restrictive, and the problem becomes even more difficult in the multi-input multi-output case [1]. These conditions are known as matching conditions, and often limits the applications for Model Reference adaptive systems [1][4]. For this reason, another nonlinear control approach called Backstepping Control and its adaptive variant called Adaptive Backstepping Control have been widely used for nonlinear control systems [2][19][20][21]. Both of these methods are based on the stability theory known as "Lyapunov Stability Theory, which will be introduced in the following section.

2.2 Lyapunov stability theory

In the control theory, a rigorous stability theory that guarantees the stability of the system plays an important role, as an unstable control system is useless.
Lyapunov stability theory gives tools to analyze the stability of nonlinear systems and has been widely used in the control engineering field [22]. Lyapunov stability includes two different approaches: the first approach is known as Lyapunov direct method while the second, less used, is known as Lyapunov indirect method. The more used Lyapunov direct method can achieve global stability of a nonlinear system without the need to linearize the system. For this reason, the direct method is more used than the indirect method [22]. The main idea of the Lyapunov direct method is that if the total energy of a system is continuously dissipating, then the system will eventually reach an equilibrium point and so guarantee stability [22]. In order to apply the direct Lyapunov Stability method to a nonlinear system, a suitable scalar function, which it will be referred to as Lyapunov function, needs to be derived. Next the time derivative of the Lyapunov Function needs to be evaluated along the trajectory of the considered nonlinear system. The system will be considered stable if the derivative of the Lyapunov Function is negative along the trajectory.

2.2.1 Lyapunov Direct Method

Consider the following dynamical system:

$$\dot{x} = f(x,t) \qquad t \geq 0 \tag{2.1}$$
$$x(t_0) = x_0 \qquad t_0 \geq 0 \tag{2.2}$$

where $x \in \mathbf{R}^n$, f is a nonlinear continuous function in t where $t \in \mathbf{R}^+$.
It is assumed that $f(x,t)$ satisfies the existence condition and uniqueness of solutions. The nonlinear system is said to be autonomous, or time-invariant, if $f(x,t)$ does not depend explicitly on the time t. In this case, the system can be written as:

$$\dot{x} = f(x) \tag{2.3}$$

Let $x_{eq} = 0$ be an equilibrium point ($f(x_{eq},t) = 0$) where $f : \Omega \to \mathbf{R}^n$ is a locally Lipchitz and $\Omega \subset \mathbf{R}^n$ a domain that contains the origin.
The considered equilibrium point $x_{eq} = 0$ is said to be *locally stable* if all point in its neighborhood remain near x_{eq} for all time. The equilibrium point x_{eq} is said to be *locally asymptotically stable* if x_{eq} is locally stable and as $t \to \infty$ all solutions starting near x tend towards the equilibrium point x_{eq}.

Let a function $V(x_{eq}, t) : \Omega \rightarrow \mathbf{R}$ be a non negative and continuously differentiable function in Ω with derivative $\dot{V}(x_{eq}, t)$ along the trajectories of the considered system.

- If $-\dot{V}(x_{eq}, t)$ is negative semidefinite, then $x_{eq} = 0$ is a stable equilibrium point.
- If $-\dot{V}(x_{eq}, t)$ is negative definite, then $x_{eq} = 0$ is an asymptotically stable equilibrium point.

If such function $V(x)$ exists, it is called Lyapunov Function.
The existence of a Lyapunov function is sufficient to prove stability, in the sense of Lyapunov, in the region Ω [22]. The Lyapunov approach has been widely used since it allows to study and guarantee the stability of a given equilibrium point without solving the differential equations that describe the considered system. A disadvantage of a Lyapunov approach to study the stability of a system is that there is no general approach to derive a Lyapunov Function. In the literature, depending on the specific application, mathematical and physical insight are often used to derive a suitable Lyapunov Function. In the next section the Lyapunov stability theory will be used to guarantee the stability of the system.

2.3 Backstepping Control

Backstepping control is a recursive approach to control nonlinear systems developed by V. Kokotovic, Krstic and Kanellakopoulos [2]. The main goal of the backstepping control is to guarantee the tracking between the output of the considered controlled system and a reference signal while guaranteeing global stability. In order to achieve the tracking, the considered nonlinear system is first transformed into a strict feedback form and then divided into smaller subsystems which will be gradually stabilized. The procedure starts by the stabilizing the first subsystem, which guarantees the main tracking objective by using a stabilizing function which will act as virtual control. The defined stabilized function will generate a new subsystem. The control action of the newly generated subsystem will be 'stepped back' to a new virtual control, which will be generate a new subsytem and so a new virtual control will be defined. The procedure ends when the real control input u, which acts as the final virtual control, is reached. At each step, for every subsystem, an augmented Lyapunov Function of the Lyapunov Function defined in the previous step is derived and then then a direct Lyapunov stability method is applied to guarantee the asymptotic stability of the considered subsystem. Because the control action is *back stepped* after each subsystem is stabilized, the process is called "backstepping".
An example of a strict-feedback systems is:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 \\ \vdots \\ \dot{x}_i = f_i(x_1, x_2, \dots x_i) + g_i(x_1, x_2, \dots x_i)x_{i+1} \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \dots x_n) + g_n(x_1, x_2, \dots x_n)u \end{cases}$$

where $x_1, \dots, x_n \in \mathbf{R}$ are the state variables of the system, $u \in \mathbf{R}$ is the control input and f_i, g_i, for $i = 1, \dots, n$ are known functions.

The basic recursive design of a backstepping control is known as the *integrator backstepping*, which will be shown in the next section. Based on the integrator backstepping, the control design will be expanded to the class strict-feedback systems case previously mentioned.

2.3.1 Integrator Backstepping Control

Consider the following system:

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \tag{2.4}$$
$$\dot{x}_2 = u \tag{2.5}$$

where $x_1, x_2 \in \mathbf{R}$ are the state variables of the system, $u \in \mathbf{R}$ is the control input and f_1, g_1 are known functions.
The objective of the control system is to design a state feedback control law such that $x_1, x_2 \to 0$ as $t \to \infty$. This system can be viewed as a cascade connection of two subsystems. The first subsystem is (2.4) with x_2 as input and the second subsystem is the integrator (2.5).
The second system variable x_2 will be considered a virtual control input for the stabilization of the first system variable x_1. Assume that there exists a smooth state feedback control law $x_2 = \alpha(x_1)$, with $\alpha(0) = 0$, such that the origin of:

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)\alpha(x_1) \tag{2.6}$$

is asymptotically stable. Consider that for the choice of $\alpha(x_1)$, a Lyapunov function $V(x_1)$ that guarantees the following condition is known:

$$\frac{\partial V}{\partial x_1}[f_1(x_1) + g_1(x_1)\alpha(x_1)] \leq -W(x_1) \tag{2.7}$$

where $W(x_1)$ is positive definite. By adding and subtracting $g(x_1)\alpha(x_1)$ on the right hand side of (2.4), one has:

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)\alpha(x_1) + g_1(x_1)[x_2 - \alpha(x_1)] \tag{2.8}$$
$$\dot{x}_2 = u \tag{2.9}$$

The error e between the state x_2 and the pseudo control $\alpha(x_1)$ is defined as:

$$e = x_2 - \alpha(x_1) \tag{2.10}$$

By writing the initial system in the x_1 and e coordinates, one has:

$$\dot{x}_1 = [f_1(x_1) + g_1(x_1)\alpha(x_1)] + g_1(x_1)e \tag{2.11}$$
$$\dot{e} = u - \dot{\alpha}(x_1) \tag{2.12}$$

Since f_1, g_1 and $\alpha(x_1)$ are known functions, the derivative $\dot{\alpha}(x_1)$ can be computed by using the expression:

$$\dot{\alpha} = \frac{\partial \alpha}{\partial x_1}[f_1(x_1) + g_1(x_1)e] \tag{2.13}$$

The following subsystem with state variables x_1 and e can be defined:

$$\dot{x}_1 = [f_1(x_1) + g_1(x_1)\alpha(x_1)] + g_1(x_1)e \tag{2.14}$$
$$\dot{e} = u - \dot{\alpha}(x_1) \tag{2.15}$$

In this newly obtained subsystem, the origin of the first component (2.14) is asymptotically stable when the virtual input e is zero because the condition in (2.7). By applying the backstepping procedure, the stabilizing function $\alpha(x_1)$ acting as a pseudo has been "back stepped" through the integrator from $u = e + \dot{\alpha}(x_1)$. An augmented Lyapunov Function V_1, which will include the newly defined e variable, will be defined as follows:

$$V_1(x_1, e) = V(x_1) + \frac{1}{2}e^2 \tag{2.16}$$

by taking the derivative of the augmented Lyapunov Function $\dot{V}_1(x_1, e)$, one obtains:

$$\dot{V}_1(x_1, e) = \dot{V}(x_1) + e\dot{e} \tag{2.17}$$
$$\dot{V}_1(x_1, e) = \frac{\partial V}{\partial x_1}[f_1 + g_1\alpha(x_1) + g_1 e] + e(u - \dot{\alpha}(x_1)) \tag{2.18}$$
$$\dot{V}_1(x_1, e) = \frac{\partial V}{\partial x_1}[f_1 + g_1\alpha(x_1)] + e(u - \frac{\partial \alpha}{\partial x_1}[f_1 + g_1 e] + \frac{\partial V}{\partial x_1}g_1) \tag{2.19}$$
$$< W_1 + e(u - \frac{\partial \alpha}{\partial x_1}[f_1 + g_1 e] + \frac{\partial V}{\partial x_1}g_1) \tag{2.20}$$

The real control input u is available in the derivative of the augmented Lyapunov Function.
If one chooses the following value for the real control input u:

$$u = -k_1 e + -\frac{\partial \alpha}{\partial x_1}[f_1 + g_1 e] - \frac{\partial V}{\partial x_1}g_1, \qquad k_1 > 0 \tag{2.21}$$

then

$$\dot{V}_1(x_1, e) < -W(x_1) - k_1 e^2 \tag{2.22}$$

which shows that the origin ($x_1 = 0, e = 0$) is asymptotically stable for the Lyapunov direct method. Since $\alpha(0) = 0$, and $e \to 0$ as $t \to \infty$; then, the, origin of the initial system $[x_1(0), x_2(0)]^T = [0, 0]^T$ is asymptotically stable as well. The backstepping algorithm stopped after two iterations as the real control input u was available in the derivative of the last defined Lyapunov Function $\dot{V}_1(x_1, e)$. In the next section, a backstepping control scheme for a second order system in a strict feedback form will be derived based on the presented integrator backstepping approach.

2.3.2 Backstepping System Design

This section illustrates the implementation of the backstepping methodology to a second order system in a strict feedback form. The explosion of complexity caused by the calculation of the derivative of the stabilizing function will be shown. In previous section, the Integrator backstepping has been used to stabilize the origin of the system. In this case the backstepping algorithm is used to guarantee the tracking between the output of the system $y = x_1$ and the desired trajectory y_d. First, the case where all the parameters of the systems are known will be considered. Later an adaptive approach will be implemented in order to estimate the unknown parameters while guaranteeing the tracking between the output of the system $y = x_1$ and the desired trajectory y_d.
Consider the following second order system:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + (1 + 0.1x_1^2)x_2 \\ \dot{x}_2 = x_1x_2 + (2 + cos(x_1))u \\ y = x_1 \end{cases} \tag{2.23}$$

where x_1,x_2 are the state variables and y is the output of the system. The initial condition is $x_0 = [x_{10}, x_{20}]^T = [0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$.

STEP 1

The tracking error between the output $y = x_1$ and the reference signal y_d is defined as:

$$\tilde{x}_1 = x_1 - y_d \tag{2.24}$$

To derive the stabilizing function α_1 that will act as a virtual input to stabilize the first system in order to guarantee $x_1 = y_d$ asymptotically, the Lyapunov Function $V_1(\tilde{x}_1)$ needs to be derived and its derivative will be calculated.
The derivative of the tracking error (2.24) is evaluated as follows:

$$\dot{\tilde{x}}_1 = \dot{x}_1 - \dot{y}_d = 0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d \tag{2.25}$$

The following Lyapunov Function $V_1(\tilde{x}_1)$ is considered:

$$V_1(\tilde{x}_1) = \frac{1}{2}\tilde{x}_1^2 \tag{2.26}$$

Then, the derivative $\dot{V}_1(\tilde{x}_1)$ is evaluated:

$$\dot{V}_1(\tilde{x}_1) = \tilde{x}_1\dot{\tilde{x}}_1 = \tilde{x}_1(0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d) \tag{2.27}$$

In order to achieve $\dot{V}_1(\tilde{x}_1) < 0$ and so $\tilde{x}_1 \to 0$ asymptotically, the stabilizing function $x_2 = \alpha_1$ is chosen as:

$$\alpha_1 = -\frac{1}{(1 + 0.1x_1^2)}(0.5x_1 - \dot{y}_d + k_1\tilde{x}_1) \tag{2.28}$$

where $k_1 > 0$ is a tuning parameter.

By substituting $x_2 = \alpha_1$ in (2.27):

$$\dot{V}_1(\tilde{x}_1) = -k_1\tilde{x}_1^2 < 0 \tag{2.29}$$

This guarantees $\tilde{x}_1 \to 0$ asymptotically.

STEP 2

The error variable $\tilde{x}_2$ will be defined as:

$$\tilde{x}_2 = x_2 - \alpha_1 \tag{2.30}$$

The objective is to make $\tilde{x}_2 \to 0$ asymptotically in order to force the state variable x_2 to follow α_1. An augmented Lyapunov Function $V_2(\tilde{x}_1, \tilde{x}_2)$ will be defined and then the stabilizing function α_2 that will guarantee $\dot{V}_2(\tilde{x}_1, \tilde{x}_2) < 0$ will be calculated.
The derivative of $\tilde{x}_2$ is evaluated:

$$\begin{aligned}\dot{\tilde{x}}_2 &= \dot{x}_2 - \dot{\alpha}_1 \\ &= x_1x_2 + (2 + cos(x_1))u + \frac{1}{(1+0.1x_1^2)^2}\Big[(0.5\dot{x}_1 - \ddot{y}_d + k_1\dot{\tilde{x}}_1)(1+0.1x_1^2) - \\ &+ 0.5x_1\dot{x}_1(0.5x_1 - \dot{y}_d + k_1\tilde{x}_1)\Big]\end{aligned} \tag{2.31}$$

where:

$$\dot{\tilde{x}}_1 = \dot{x}_1 - \dot{y}_d \tag{2.32}$$

$$\dot{x}_1 = 0.5x_1 + (1 + 0.1x_1^2)\tilde{x}_2 \tag{2.33}$$

The candidate Lyapunov function $V_2(\tilde{x}_1, \tilde{x}_2)$ is chosen as follows:

$$V_2(\tilde{x}_1, \tilde{x}_2) = \frac{1}{2}\tilde{x}_1^2 + \frac{1}{2}\tilde{x}_2^2 \tag{2.34}$$

And so its derivative is considered:

$$\dot{V}_1(\tilde{x}_1, \tilde{x}_2) = \tilde{x}_1\dot{\tilde{x}}_1 + \tilde{x}_2\dot{\tilde{x}}_2 = \tilde{x}_1(0.5x_1 + (1 + 0.1x_1^2)(\tilde{x}_2 + \alpha_1) - \dot{y}_d) + \tilde{x}_2\dot{\tilde{x}}_2 \tag{2.35}$$

By substituting α_1 with (2.28) and $\dot{\tilde{x}}_2$ with (2.31), the derivative $\dot{V}_2(\tilde{x}_1, \tilde{x}_2)$ becomes:

$$\begin{aligned}\dot{V}_2(\tilde{x}_1, \tilde{x}_2) &= \tilde{x}_1(-k_1\tilde{x}_1 + (1 + 0.1x_1^2)\tilde{x}_2) + \tilde{x}_2(x_1x_2 + (2 + cos(x_1))u + \Theta) \\ &= -k_1\tilde{x}_1^2 + \tilde{x}_2((1 + 0.1x_1^2)\tilde{x}_1 + x_1x_2 + (2 + cos(x_1))u + \Theta)\end{aligned} \tag{2.36}$$

where:

$$\Theta = \frac{1}{(1+0.1x_1^2)^2}\Big[(0.5\dot{x}_1 - \ddot{y}_d + k_1\dot{\tilde{x}}_1)(1+0.1x_1^2) - 0.5x_1\dot{x}_1(0.5x_1 - \dot{y}_d + k_1\tilde{x}_1)\Big] \tag{2.37}$$

The real control input variable u is available in the derivative of the augmented Lyapunov Function (equation 2.36). For this reason there is no need to define a second stabilizing function α_2. The control law u that allows $\dot{V}_2(\tilde{x}_1, \tilde{x}_2) < 0$ is defined as follows:

$$u = -\frac{1}{2 + cos(x_1)}(\Theta + x_1 x_2 + (1 + 0.1x_1^2)\tilde{x}_1 + k_2\tilde{x}_2) \tag{2.38}$$

where $k_2 > 0$ is a tuning parameter.
The derivative of the Lyapunov Function $\dot{V}_2(\tilde{x}_1, \tilde{x}_2)$ becomes:

$$\dot{V}_1(\tilde{x}_1, \tilde{x}_2) = -k_1\tilde{x}_1^2 - k_2\tilde{x}_2^2 < 0 \tag{2.39}$$

This guarantees $\tilde{x}_1 = x_1 - y_d \to 0$ and $\tilde{x}_2 = x_2 - \alpha_1 \to 0$ asymptotically. Summarizing the process, $x_1 - y_d \to 0$ was guaranteed by using a virtual input α_1. Later the real available signal x_2 was forced to follow α_1 by choosing a proper value of the control input u.

2.4 Command Filtered Backstepping Control

In the traditional backstepping algorithm presented in the previous chapter, at each step of the procedure, the calculation of the virtual control signals α_i and their derivatives is required. Theoretically, the calculation of the virtual control signal derivatives is simple, but it can be quite complicated and tedious in applications when the order of the controlled system increases [23]. This happens because at the end of the procedure, the real control signal u will include the derivative of α_n, which requires the second derivative of α_{n-1}, which requires the third derivative of α_{n-2}, and so on. In the previously presented example, in the case of a second order system, equation (2.36) and (2.37) clearly show the problem known as "explosion of complexity" [8]. In certain applications, such as induction motors, the number of backstepping iterations is small and the computation is achievable [24]. In other applications, such as a helicopter application or high-order nonlinear multiagent systems [7][19][25], the analytic derivation of the stabilizing functions α_i is tedious and heavy computing. This problem known as "explosion of complexity" has been addressed by a variety of methods [8][10][23]. In this work the explosion of complexity problem will be avoided by using a command filter approach. The basic concept of a command filter appraoch is to filter at each step the stabilizing function α_i in order to obtain two new signals, $x_{i+1,c}$ and its derivative $\dot{x}_{i+1,c}$. These new signals will be then used instead of α_i and its derivative $\dot{\alpha}_i$. To compensate the filtering error, compensation error variables will be defined. The command filter approach decouples of the design of the controllers for the backstepping iterations, it avoids the tedious algebra related to computing the command signal derivatives and it only requires the reference signals y_d and its $(n-1)$ derivatives to be available as inputs to the system [8]. In the following section, a command filtered backstepping control will be developed for the second order system (2.23) previously controlled by a traditional backstepping approach.

2.4.1 Command filter design

In this section an implementation of the command filter to the Backstepping approach is presented. The objective is eliminate the analytic computation of the derivatives $\dot{\alpha}_1$ for $i = 1, \ldots, (n-1)$, while guaranteeing a rigorous stability analysis.
Consider the following strict-feedback nonlinear of order n:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 \\ \vdots \\ \dot{x}_i = f_i(x_1, x_2, \ldots, x_i) + g_i(x_1, x_2, \ldots, x_i)x_{i+1} \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \ldots, x_n) + g_n(x_1, x_2, \ldots, x_n)u \\ y = x_1 \end{cases} \tag{2.40}$$

In the traditional backstepping design, the stabilizing functions α_i are defined as:

$$\alpha_1(x_1, y_d) = \frac{1}{g_1}(-k_1\tilde{x}_1 + \dot{y}_d - f_1) \tag{2.41}$$

$$\alpha_i(x_1, \ldots, x_i, y_d) = \frac{1}{g_i}(-k_i\tilde{x}_i + \dot{\alpha}_{i-1} - g_{i-1}x_{i_1}) \tag{2.42}$$

for $i = 2, \ldots, n$ and $k_i > 0$ for $i = 1, \ldots, n$ and the control variable is assigned the value $u(t) = \alpha_n(x_1, \ldots, x_n, y_d)$. On the other hand, in the command filtered approach, the definition of the virtual control signals α_i of the backstepping procedure is as follows:

$$\alpha_1(x_1, x_{1,c}) = \frac{1}{g_1}(-k_1\tilde{x}_1 + \dot{x}_{1,c} - f_1) \tag{2.43}$$

$$\alpha_i(x_1, \ldots, x_i, x_{i,c}, v_{i-1}) = \frac{1}{g_i}(-k_i\tilde{x}_i + \dot{x}_{i,c} - f_i - g_{i-1}v_{i-1}) \tag{2.44}$$

for $i = 2, \ldots, n$ and $k_i > 0$ for $i = 1, \ldots, n$, $x_{1,c} = y_d$ and the control variable is assigned the value $u(t) = \alpha_n(x_1, \ldots, x_n, x_{n,c}, v_{n-1})$.
The following errors are defined:

- The tracking error signals $\tilde{x}_i$:

$$\tilde{x}_i = x_i - x_{i,c} \qquad i = 1, \ldots, n. \tag{2.45}$$

  where $x_{1.c} = y_d$. It resembles the tracking error in the traditional backstepping control (see equation 2.24).

- The signals ξ_i, for $i = 1, \ldots, (n-1)$ produced by filtering, $(x_{i+1,c} - \alpha_i)$ describe the unachieved portion of α_i and are define as follows:

$$\dot{\xi}_i = -k_i\xi_i + g_i(x_{x+1,c} - \alpha_i) \tag{2.46}$$

  where $\xi_i(0) = 0$ and $\xi_n = 0$.

- The compensated tracking errors v_i are obtained by removing the filtered unachieved portion of α_i, as represented by ξ_i, from the tracking error and are defined as:

$$v_i = \tilde{x}_i - \xi_i \qquad i = 1, \ldots, n. \tag{2.47}$$

The signals $x_{i,c}$ and $\dot{x}_{i,c}$ for $i = 1, 2, \ldots, n$ are the output of the following command filter

$$\begin{cases} \dot{q}_{i,1} = \omega_n q_{i,2} \\ \dot{q}_{i,2} = -2\zeta\omega_n q_{i,2} - \omega_n(q_{i,1} - \alpha_i) \end{cases} \tag{2.48}$$

where $x_{i+1,c} = q_{i,1}$ and $\dot{x}_{i+1,c} = \omega_n q_{i,2}$ are the outputs of each command filter implemented in the system. For $i = 1$, $x_{1,c} = y_d = \alpha_0$ and $\dot{x}_{1,c} = \dot{y}_d = \dot{\alpha}_0$.

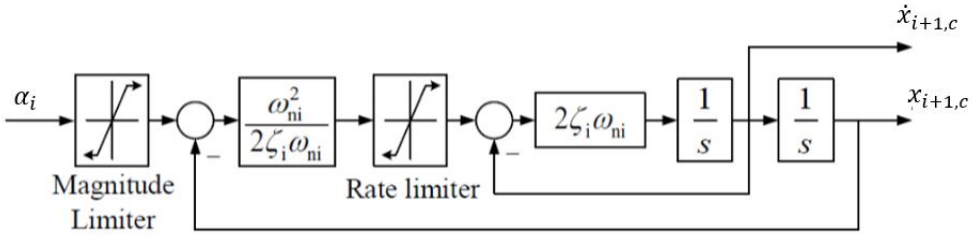

Figure 2.2: The structure of a command filter

The filter initial conditions are:

$$q_{i,1}(0) = \alpha_i(x_1(0, \ldots, x_i(0), x_{i,x}(0), v_{i-1}(0)))$$
$$q_{i,2}(0) = 0$$

The filter design parameters are $\omega_n > 0$ and $\zeta \in (0, 1]$. Each command filter is designed to compute $x_{i+1,c}$ and $\dot{x}_{i+1,c}$ without differentiation.
To achieve good tracking performance between $x_{i+1,c}$, $\dot{x}_{i+1,c}$ and α_i, $\dot{\alpha}_i$ respectively, the designer would choose $\omega_n > k_{i+1}$ for $i = 1, \ldots, n$ [8].
In the next section, a command filtered backstepping control will be developed and implemented to a second order nonlinear system.

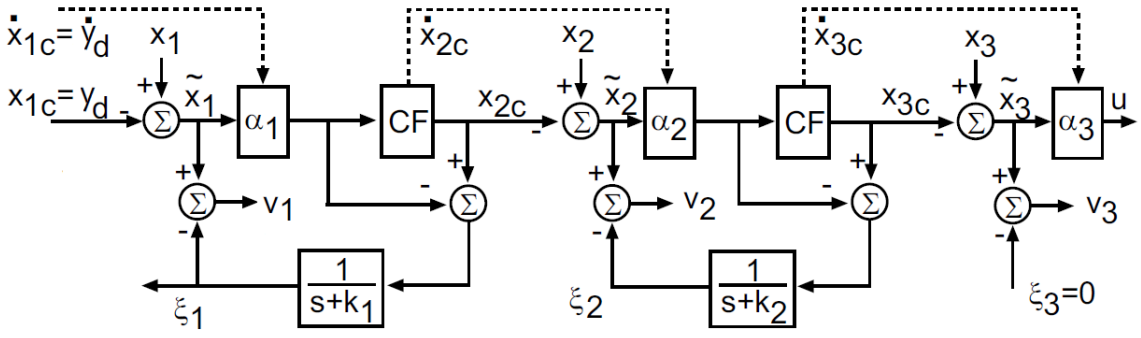

Figure 2.3: Block diagram of a command filtered backstepping controller for a 3rd order system

2.4.2 Command Filtered Backstepping System Design

In this section, a command filtered backstepping control will be implemented to the second order system (equation 2.23) previously controlled by a traditional backstepping control, in order to show the simplification in the design process.
Consider the following second order system:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + (1 + 0.1x_1^2)x_2 \\ \dot{x}_2 = x_1x_2 + (2 + cos(x_1))u \\ y = x_1 \end{cases} \tag{2.49}$$

where x_1,x_2 are the state variables and y is the output of the system.
The initial condition is $x_0 = [x_{10}, x_{20}]^T = [0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$. The implemented command filtered backstepping approach resembles the traditional backstepping approach until the first stabilizing function α_1 is obtained. The stabilizing function α_1 will be then filtered and the new control law will be obtained, without using its derivative $\dot{\alpha}_1$.

STEP 1

The backstepping algorithm starts by defining the tracking error $\tilde{x}_1$ between the output $y = x_1$ and the reference signal y_d as:

$$\tilde{x}_1 = x_1 - y_d \tag{2.50}$$

To derive the stabilizing function α_1, which will act as a virtual input to stabilize the first system in order to guarantee $x_1 = y_d$ asymptotically, the Lyapunov Function $V_1(\tilde{x}_1)$ and its derivative will be defined.
First, the derivative of the tracking error $\tilde{x}_1$ is calculated:

$$\dot{\tilde{x}}_1 = \dot{x}_1 - \dot{y}_d = 0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d \tag{2.51}$$

The following Lyapunov Function $V_1(\tilde{x}_1)$ is considered:

$$V_1(\tilde{x}_1) = \frac{1}{2}\tilde{x}_1^2 \tag{2.52}$$

Then its derivative $\dot{V}_1(\tilde{x}_1)$ is calculated:

$$\dot{V}_1(\tilde{x}_1) = \tilde{x}_1\dot{\tilde{x}}_1 = \tilde{x}_1(0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d) \tag{2.53}$$

To achieve $\dot{V}_1(\tilde{x}_1) < 0$ and so $\tilde{x}_1 \to 0$ asymptotically, the following stabilizing function $x_2 = \alpha_1$ is chosen:

$$\alpha_1 = -\frac{1}{(1 + 0.1x_1^2)}(0.5x_1 - \dot{y}_d + k_1\tilde{x}_1) \tag{2.54}$$

where $k_1 > 0$ is a tuning parameter.
By substituting $x_2 = \alpha_1$ in (2.53):

$$\dot{V}_1(\tilde{x}_1) = -k_1\tilde{x}_1^2 < 0 \tag{2.55}$$

This guarantees $\tilde{x}_1 \to 0$ asymptotically.

STEP 2

The next step is to pass α_1 to a command filter to obtain $x_{2,c}$ and $\dot{x}_{2,c}$. Recalling equation (2.48), the command filter can be described by the following space state equations:

$$\begin{cases} \dot{q}_{1,1} = \omega_n q_{1,2} \\ \dot{q}_{1,2} = -2\zeta\omega_n q_{1,2} - \omega_n(z_{1,1} - \alpha_2) \end{cases} \tag{2.56}$$

where $x_{2,c} = q_{1,1}$ and $\dot{x}_{2,c} = \omega_n q_{1,2}$ are the outputs of the command filter.
The second tracking error signal $\tilde{x}_2$ is defined as follows:

$$\tilde{x}_2 = x_2 - x_{2,c} \tag{2.57}$$

and its derivative $\dot{\tilde{x}}_2$ is calculated:

$$\begin{aligned} \dot{\tilde{x}}_2 &= \dot{x}_2 - \dot{x}_{2,c} \\ &= x_1 x_2 + (2 + cos(x_1))u - \dot{x}_{2,c} \end{aligned} \tag{2.58}$$

The command filter will produce a filtering error that may increase the difficulty in getting the tiny tracking error. For this reason a compensation tracking error v_1 of $\tilde{x}_1$ is defined as:

$$v_1 = \tilde{x}_1 - \xi_1 \tag{2.59}$$

Where the $\dot{\xi}_1$ signal is defined as:

$$\dot{\xi}_1 = -k_1\xi_1 + (1 + 0.1x_1^2)(x_{2,c} - \alpha_1) \quad , \quad \xi_1(0) = 0 \tag{2.60}$$

The derivative $\dot{v}_1$ of v_1 is now calculated, as it will be necessary in the calculation of the derivative of the next Lyapunov Function.

$$\begin{aligned} \dot{v}_1 &= \dot{\tilde{x}}_1 - \dot{\xi}_1 \\ &= (0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d) - (-k_1\xi_1 + (1 + 0.1x_1^2)(x_{2,c} - \alpha_1)) \\ &= (0.5x_1 + (1 + 0.1x_1^2)x_2 - \dot{y}_d + k_1\xi_1 - (1 + 0.1x_1^2)x_{2,c} + (1 + 0.1x_1^2)\alpha_1 \\ &= (0.5x_1 + (1 + 0.1x_1^2)\tilde{x}_2 - \dot{y}_d + k_1\xi_1 + (1 + 0.1x_1^2)\alpha_1 \\ &= (1 + 0.1x_1^2)\tilde{x}_2 + k_1\xi_1 + (1 + 0.1x_1^2)\tilde{x}_1 \\ &= -k_1(\tilde{x}_1 - \xi_1) + (1 + 0.1x_1^2)\tilde{x}_2 \\ &= -k_1 v_1 + (1 + 0.1x_1^2)\tilde{x}_2 \end{aligned} \tag{2.61}$$

The following Lyapunov Function $V_2(v_1, \tilde{x}_2)$ is considered:

$$V_2(v_1, \tilde{x}_2) = \frac{1}{2}v_1^2 + \frac{1}{2}\tilde{x}_2^2 \tag{2.62}$$

its derivative is calculated as follows:

$$\begin{aligned} \dot{V}_2(v_1, \tilde{x}_2) &= v_1\dot{v}_1 + \tilde{x}_2\dot{\tilde{x}}_2 \\ &= v_1(-k_1 v_1 + (1 + 0.1x_1^2)\tilde{x}_2) + \tilde{x}_2\dot{\tilde{x}}_2 \\ &= v_1(-k_1 v_1 + (1 + 0.1x_1^2)\tilde{x}_2) + \tilde{x}_2(x_1 x_2 + (2 + cos(x_1))u - \dot{x}_{2,c}) \\ &= -k_1 v_1^2 + \tilde{x}_2((1 + 0.1x_1^2)v_1 + x_1 x_2 + (2 + cos(x_1))u - \dot{x}_{2,c}) \end{aligned} \tag{2.63}$$

The real control input variable u is available in the derivative of the second Lyapunov function(equation 2.63). For this reason another virtual control α_2 is not necessary.
The value of the control signal u that allows $\dot{V}_1(v_1, \tilde{x}_2) < 0$ is defined as follows:

$$u = -\frac{1}{2 + cos(x_1)}(-\dot{x}_{2,c} + x_1 x_2 + (1 + 0.1x_1^2)v_1 + k_2\tilde{x}_2) \tag{2.64}$$

where $k_2 > 0$ is a tuning parameter.
The Lyapunov Function $\dot{V}_1(v_1, \tilde{x}_2)$ becomes:

$$\dot{V}_1(v_1, \tilde{x}_2) = -k_1 v_1^2 - k_2 \tilde{x}_2^2 < 0 \tag{2.65}$$

This guarantees $\tilde{x}_2 \to 0$ asymptotically, $v_1 \to 0 \implies \tilde{x}_1 \to 0$ asymptotically, with the consequence that $x_1 = y_d$ asymptotically. It can be noticed that in the command filtered approach here implemented, the derivative $\dot{\alpha}_1$ was not used to determine the control input u, avoiding so the explosion of complexity encountered when using the traditional backstepping control method.
In the next section, the developed command filtered backstepping control will be extended to the case where the controlled system presents unknown parameters, which will be estimated by using an adaptive approach.

2.5 Adaptive Command Filtered Backstepping Control

In this section, an adaptive control law will be implemented in the command filtered backstepping control previously developed, in order to estimate unknown parameters in the system.
The following class of nonlinear systems is considered:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \phi_1^T(x_1)\theta_1 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 + \phi_2^T(x_1, x_2)\theta_2 \\ \vdots \\ \dot{x}_i = f_i(x_1, x_2, \dots x_i) + g_i(x_1, x_2, \dots x_i)x_{i+1} + \phi_i^T(x_1, x_2, \dots x_i)\theta_i \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \dots x_n) + g_n(x_1, x_2, \dots x_n)u + \phi_n^T(x_1, x_2, \dots x_n)\theta_n \\ y = x_1 \end{cases} \tag{2.66}$$

where $x_1, \dots, x_n \in \mathbf{R}$ are the state variables of the system, $u \in \mathbf{R}$ is the control input and $f_1, f_2, \dots, f_i, \dots, f_n$, $g_1, g_2, \dots, g_i, \dots, g_{\in}\mathbf{R}$, $\phi_1, \phi_2, \dots \phi_i \dots, \phi_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are known functions and $\theta_1, \theta_2, \dots, \theta_i, \dots, \theta_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are unknown parameters. The objective is to globally stabilize the system and achieve the asymptotic tracking between the system output $y = x_1$ and the reference signal y_d. In section 2.3.2 an implementation of the traditional backstepping control based on the integrator backstepping was considered to control a second order nonlinear system to show the explosion of complexity problem. In section 2.4.2 a command filtered backstepping control was implemented to the same second order nonlinear system to avoid the explosion of complexity problem by using a command filter. In the following section, a parametric nonlinear system of order three with unknown parameters will be considered and the previously implemented command filtered backstepping approach will be extended to the adaptive case. Being the backstepping control a recursive approach, the extension to higher order systems results simple, and it will be discussed later in the work.

2.5.1 Third order system control

The following parametric nonlinear system of order three is considered:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \phi_1^T(x_1)\theta_1 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 + \phi_2^T(x_1, x_2)\theta_2 \\ \dot{x}_3 = f_3(x_1, x_2, x_3) + g_3(x_1, x_2, x_3)u + \phi_3^T(x_1, x_2, x_3)\theta_3 \\ y = x_1 \end{cases} \tag{2.67}$$

where $x_1, x_2, x_3 \in \mathbf{R}$ are the state variables of the system, $u \in \mathbf{R}$ is the control input and $f_1, f_2, f_3 \in \mathbf{R}$, $g_1, g_2, g_3 \in \mathbf{R}$ and $\phi_1, \phi_2, \phi_3 \in \mathbf{R}^T$ are known nonlinear functions and $\theta_1, \theta_2, \theta_3 \in \mathbf{R}^T$ are unknown parameters.
The objective is to globally stabilize the system and achieve the asymptotic tracking between the output of the system $y = x_1$ and the reference signal y_d.

STEP 1

The backstepping algorithm starts by defining the tracking error between the output $y = x_1$ and the reference signal y_d as:

$$\tilde{x}_1 = x_1 - y_d \tag{2.68}$$

To derive the stabilizing function α_1, which will act as a virtual input to stabilize the first system and guarantee $x_1 = y_d$ asymptotically, a Lyapunov Function and its derivative will be defined.
First, the derivative of the tracking error $\tilde{x}_1$ is considered:

$$\dot{\tilde{x}}_1 = \dot{x}_1 - \dot{y}_d = f_1 + g_1 x_2 + \phi_1^T \theta_1 - \dot{y}_d \tag{2.69}$$

Since θ_1 is unknown, an estimate value $\hat{\theta}_1$ of θ_1 is considered and then an estimation law for $\hat{\theta}_1$ will be determined. Moreover, an estimation error $\tilde{\theta}$ between the real value of θ_1 and its estimation $\hat{\theta}_1$ will be defined as follows:

$$\tilde{\theta}_1 = \theta_1 - \hat{\theta}_1 \tag{2.70}$$

The following Lyapunov Function $V_1(\tilde{x}_1, \tilde{\theta}_1)$, which also includes the estimation error $\tilde{\theta}_1$ is chosen:

$$V_1(\tilde{x}_1, \tilde{\theta}_1) = \frac{1}{2}\tilde{x}_1^2 + \frac{1}{2}\tilde{\theta}_1^T \Gamma_1^{-1} \tilde{\theta}_1 \tag{2.71}$$

where Γ_1 is a diagonal positive definite matrix.
The derivative of the Lyapunov Function $V_1(\tilde{x}_1, \tilde{\theta}_1)$ will be now calculated:

$$\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) = \tilde{x}_1 \dot{\tilde{x}}_1 + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 = \tilde{x}_1 (f_1 + g_1 x_2 + \phi_1^T \theta_1 - \dot{y}_d) + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 \tag{2.72}$$

In order to achieve $\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) < 0$ and so $\tilde{x}_1 \to 0$ asymptotically, one would choose the stabilizing function $x_2 = \alpha_1$ as:

$$\alpha_1 = \alpha_1^* = -\frac{1}{g_1}(f_1 + \phi_1^T \theta_1 - \dot{y}_d - k_1 \tilde{x}_1) \tag{2.73}$$

where $k_1 > 0$ is a tuning parameter.
Because θ_1 is not available, the following stabilizing function α_1 is chosen, where instead of θ_1, its estimate value $\hat{\theta}_1$ is included:

$$\alpha_1 = -\frac{1}{g_1}(f_1 + \phi_1^T \hat{\theta}_1 - \dot{y}_d - k_1 \tilde{x}_1) \tag{2.74}$$

By substituting $x_2 = \alpha_1$ in equation 2.72:

$$\begin{aligned} \dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) &= \tilde{x}_1(-k_1 \tilde{x}_1 + \phi_1^T \theta_1 - \phi_1^T \hat{\theta}_1) + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 \\ &= \tilde{x}_1(-k_1 \tilde{x}_1 + \phi_1^T \tilde{\theta}_1) + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 \\ &= -k_1 \tilde{x}_1^2 + \tilde{\theta}_1^T (\Gamma_1^{-1} \dot{\tilde{\theta}}_1 + \tilde{x}_1 \phi_1) \end{aligned} \tag{2.75}$$

If one chooses the following value for $\dot{\tilde{\theta}}_1$:

$$\dot{\tilde{\theta}}_1 = -\Gamma_1 \tilde{x}_1 \phi_1 \tag{2.76}$$

the derivative of the Lyapunov Function $\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1)$ becomes:

$$\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) = -k_1 \tilde{x}_1^2 < 0 \tag{2.77}$$

This guarantees that $\tilde{x}_1 \to 0$ asymptotically.
Based on equation 2.76, the update law for the estimated parameter $\hat{\theta}_1$ can be derived:

$$\dot{\hat{\theta}}_1 = -\dot{\tilde{\theta}}_1 = \Gamma_1 \tilde{x}_1 \phi_1 \tag{2.78}$$

$$\hat{\theta}_1 = \Gamma_1 \int \tilde{x}_1 \phi_1 \tag{2.79}$$

STEP 2

The next step is to pass α_1 to a command filter to obtain $x_{2,c}$ and $\dot{x}_{2,c}$.
The following command filter in the form of a second order system is considered:

$$\begin{cases} \dot{q}_{1,1} = \omega_n q_{1,2} \\ \dot{q}_{1,2} = -2\zeta\omega_n q_{1,2} - \omega_n (q_{1,1} - \alpha_1) \end{cases} \tag{2.80}$$

where $x_{2,c} = q_{1,1}$ and $\dot{x}_{2,c} = \omega_n q_{1,2}$ are the outputs the command filter.
The tracking error $\tilde{x}_2$ between x_2 and $x_{2,c}$ is defined as follows:

$$\tilde{x}_2 = x_2 - x_{2,c} \tag{2.81}$$

and its derivative $\dot{\tilde{x}}_2$:

$$\dot{\tilde{x}}_2 = \dot{x}_2 - \dot{x}_{2,c} \tag{2.82}$$

Because a command filter was used to filter α_1, a compensation tracking error v_1 of $\tilde{x}_1$ is derived, and it will be used instead of $\tilde{x}_1$ in the next calculations:

$$v_1 = \tilde{x}_1 - \xi_1 \tag{2.83}$$

The $\dot{\xi}_1$ signal is defined as:

$$\dot{\xi}_1 = -k_1 \xi_1 + g_1 (x_{2,c} - \alpha_1) \quad , \quad \xi_1(0) = 0 \tag{2.84}$$

The derivative of v_1 is derived, as it will be used in the calculation of the next Lyapunov function:

$$\begin{aligned} \dot{v}_1 &= \dot{\tilde{x}}_1 - \dot{\xi}_1 \\ &= [f_1 + g_1 x_2 + \phi_1^T \theta_1 - \dot{y}_d] - [-k_1 \xi_1 + g_1 (x_{2,c} - \alpha_1)] \\ &= f_1 + g_1 x_2 + \phi_1^T \theta_1 - \dot{y}_d + k_1 \xi_1 - g_1 x_{2,c} + g_1 \alpha_1 \end{aligned}$$

by substituting $\tilde{x}_2 = x_2 - x_{2,c}$ and α_1 with equation 2.74, one obtains:

$$\begin{aligned} \dot{v}_1 &= g_1 \tilde{x}_2 + \phi_1^T \theta_1 + k_1 \xi_1 - k_1 \tilde{x}_1 - \phi_1^T \hat{\theta}_1 \\ &= -k_1 (\tilde{x}_1 - \xi_1) + g_1 \tilde{x}_2 + \phi_1^T \tilde{\theta}_1 \\ &= -k_1 v_1 + g_1 \tilde{x}_2 + \phi_1^T \tilde{\theta}_1 \end{aligned} \tag{2.85}$$

The following Lyapunov Function $V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ will be chosen, where instead of $\tilde{x}_1$, its compensated signal v_1 is included:

$$V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = \frac{1}{2}\tilde{x}_1^2 + \frac{1}{2}\tilde{x}_2^2 + \frac{1}{2}\tilde{\theta}_1^T\Gamma_1^{-1}\tilde{\theta}_1 + \frac{1}{2}\tilde{\theta}_2^T\Gamma_2^{-1}\tilde{\theta}_2 \tag{2.86}$$

where Γ_1 and Γ_2 are diagonal positive definite matrices.
The derivative of $V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ will be now calculated:

$$\begin{aligned}
&\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = v_1\dot{v}_1 + \tilde{x}_2\dot{\tilde{x}}_2 + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 \\
&= v_1(-k_1v_1 + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1) + \tilde{x}_2(f_2 + g_2x_3 + \phi_2^T\theta_2 - \dot{x}_{2,c}) + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 \\
&= -k_1v_1^2 + \tilde{x}_2(g_1v_1 + f_2 + g_2x_3 + \phi_2^T\theta_2 - \dot{x}_{2,c}) + \tilde{\theta}_1^T(\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + v_1\phi) + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2
\end{aligned} \tag{2.87}$$

The following values for the stabilizing function $x_3 = \alpha_2$ and $\dot{\tilde{\theta}}_1$ are chosen:

$$\alpha_2 = -\frac{1}{g_2}(f_2 + \phi_2^T\hat{\theta}_2 - \dot{x}_{2c} - k_2\tilde{x}_2) \tag{2.88}$$

$$\dot{\tilde{\theta}}_1 = -\Gamma_1 v_1\phi_1 \tag{2.89}$$

where $k_2 > 0$ and Γ_1 are tuning parameters.
By substituting $x_3 = \alpha_2$ and $\dot{\tilde{\theta}}_1$ into equation 2.87:

$$\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = -k_1\tilde{x}_1^2 - k_2\tilde{x}_2^2 + \tilde{\theta}_2^T(\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{x}_2\phi) \tag{2.90}$$

Finally, the following value for $\dot{\tilde{\theta}}_2$ is chosen:

$$\dot{\tilde{\theta}}_1 = -\Gamma_2\tilde{x}_2\phi_2 \tag{2.91}$$

The derivative of the Lyapunov Function $\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ becomes:

$$\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = -k_1\tilde{x}_1^2 - k_2\tilde{x}_2^2 < 0 \tag{2.92}$$

This guarantees:

- $v_1 = \tilde{x}_1 - \xi_1 \to 0$ asymptotically
- $\tilde{x}_2 = x_2 - \dot{x}_{2,c} \to 0$ asymptotically

The new Adaptive Laws to estimate $\hat{\theta}_1$ and $\hat{\theta}_2$ can be derived:

$$\dot{\hat{\theta}}_1 = -\dot{\tilde{\theta}}_1 = \Gamma_1 v_1\phi_1 \tag{2.93}$$

$$\hat{\theta}_1 = \Gamma_1 \int v_1\phi_1 \tag{2.94}$$

$$\dot{\hat{\theta}}_2 = -\dot{\tilde{\theta}}_2 = \Gamma_2\tilde{x}_2\phi_2 \tag{2.95}$$

$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2\phi_2 \tag{2.96}$$

Based on this intermediate result, it's already possible to determine a control for second order system nonlinear system. The control can be achieve considering $\alpha_2 = u$.

STEP 3

The backstepping procedure will be iterated for the third and last time because the controlled system is a third order system ($n = 3$). The stabilizing function α_2 will be passed to a command filter to obtain $x_{3,c}$ and $\dot{x}_{3,c}$.
The following command filter will be considered:

$$\begin{cases} \dot{q}_{2,1} = \omega_n q_{2,2} \\ \dot{q}_{2,2} = -2\zeta\omega_n q_{2,2} - \omega_n(q_{2,1} - \alpha_2) \end{cases} \tag{2.97}$$

where $x_{3,c} = q_{2,1}$ and $\dot{x}_{3,c} = \omega_n q_{2,2}$ are the outputs of the command filter.
The tracking error $\tilde{x}_3$ between x_3 and $x_{3,c}$ is defined as follows:

$$\tilde{x}_3 = x_3 - x_{3,c} \tag{2.98}$$

and its derivative

$$\dot{\tilde{x}}_3 = \dot{x}_3 - \dot{x}_{3,c} \tag{2.99}$$

Because a command filter has been used to filter α_2, a compensation tracking error v_2 of $\tilde{x}_2$ is defined and it will be used instead of $\tilde{x}_2$ in the next calculations:

$$v_2 = \tilde{x}_3 - \xi_2 \tag{2.100}$$

The $\dot{\xi}_2$ signal is defined as:

$$\dot{\xi}_2 = -k_2\xi_2 + g_2(x_{3,c} - \alpha_2), \qquad \xi_2(0) = 0 \tag{2.101}$$

The derivative of v_2 will be considered as it will be necessary in the calculation of the derivative of the next Lyapunov function.

$$\begin{aligned} \dot{v}_2 &= \dot{\tilde{x}}_2 - \dot{\xi}_2 \\ &= [f_2 + g_2 x_3 + \phi_2^T\theta_2 - \dot{x}_{2,c}] - [-k_2\xi_2 + g_2(x_{3,c} - \alpha_2)] \\ &= f_2 + g_2 x_3 + \phi_2^T\theta_2 - \dot{x}_{2,c} + k_2\xi_2 - g_2 x_{3,c} + g_2\alpha_2 \end{aligned} \tag{2.102}$$

by substituting $\dot{\tilde{x}}_2 = \dot{x}_2 - \dot{x}_{2,c}$ and α_2 with equation 2.88, one obtains:

$$\begin{aligned} \dot{v}_2 &= g_2\tilde{x}_3 + \phi_2^T\theta_2 + k_2\xi_2 - k_2\tilde{x}_3 - \phi_2^T\hat{\theta}_2 \\ &= -k_2(\tilde{x}_3 - \xi_2) + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2 \\ &= -k_2 v_2 + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2 \end{aligned} \tag{2.103}$$

The third Lyapunov Function $V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ is defined as follows:

$$V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = \frac{1}{2}\tilde{v}_1^2 + \frac{1}{2}\tilde{v}_2^2 + \frac{1}{2}\tilde{x}_3^2 + \frac{1}{2}\tilde{\theta}_1^T\Gamma_1^{-1}\tilde{\theta}_1 + \frac{1}{2}\tilde{\theta}_2^T\Gamma_2^{-1}\tilde{\theta}_2 + \frac{1}{2}\tilde{\theta}_3^T\Gamma_3^{-1}\tilde{\theta}_3 \tag{2.104}$$

Where Γ_1, Γ_2 and Γ_3 are diagonal positive definite matrices.
The derivative of $V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ will be now calculated:

$$\begin{aligned} &\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = v_1\dot{v}_1 + v_2\dot{v}_2 + \tilde{x}_3\dot{\tilde{x}}_3 + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \\ &= v_1(-k_1 v_1 + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1) + v_2(-k_2 v_1 + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2) + \\ &+ \tilde{x}_3(f_2 + g_3 u + \phi_3^T\theta_3 - \dot{x}_{3,c}) + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \\ &= -k_1 v_1^2 + -k_2 v_1^2 + \tilde{x}_3(g_2 v_1 + f_3 + g_3 u + \phi_3^T\theta_3 - \dot{x}_{3,c}) + \\ &+ \tilde{\theta}_1^T(\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + v_1\phi) + \tilde{\theta}_2^T(\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + v_2\phi) + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \end{aligned} \tag{2.105}$$

One chooses the following values for the input signal u, $\dot{\hat{\theta}}_1$ and $\dot{\hat{\theta}}_2$:

$$u = -\frac{1}{g_3}(f_3 + \phi_3^T\hat{\theta}_3 - \dot{x}_{3,c} - k_3\tilde{x}_3) \tag{2.106}$$

$$\dot{\hat{\theta}}_1 = -\Gamma_1 v_1 \phi_1 \tag{2.107}$$

$$\dot{\hat{\theta}}_2 = -\Gamma_2 v_2 \phi_2 \tag{2.108}$$

where $k_3 > 0$, Γ_1 and Γ_2 are tuning parameters.
By substituting u, $\dot{\hat{\theta}}_1$ and $\dot{\hat{\theta}}_2$ in 2.105, one obtains:

$$\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = -k_1 v_1^2 - k_2 v_2^2 - k_3\tilde{x}_3 + \tilde{\theta}_3^T(\Gamma_3^{-1}\dot{\tilde{\theta}}_3 + \tilde{x}_3\phi) \tag{2.109}$$

Finally, the following value for $\dot{\tilde{\theta}}_3$ is chosen:

$$\dot{\tilde{\theta}}_3 = -\Gamma_3\tilde{x}_3\phi_3 \tag{2.110}$$

The Lyapunov Function $\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ becomes:

$$\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = -k_1\tilde{x}_2^2 - k_2\tilde{x}_2^2 - k_3\tilde{x}_3^2 < 0 \tag{2.111}$$

This guarantees:

- $v_1 = \tilde{x}_1 - \xi_1 \to 0$ asymptotically
- $v_2 = x_2 - \xi_2 \to$ asymptotically
- $\tilde{x}_3 = x_3 - \dot{x}_{3,c} \to 0$ asymptotically

The new Adaptive Laws to estimate $\hat{\theta}_2$ and $\hat{\theta}_3$ can be derived:

$$\dot{\hat{\theta}}_2 = -\dot{\tilde{\theta}}_2 = \Gamma_1 v_2 \phi_2 \tag{2.112}$$

$$\hat{\theta}_2 = \Gamma_2 \int v_2\phi_2 \tag{2.113}$$

$$\dot{\hat{\theta}}_3 = -\dot{\tilde{\theta}}_3 = \Gamma_3\tilde{x}_3\phi_3 \tag{2.114}$$

$$\hat{\theta}_3 = \Gamma_3 \int \tilde{x}_3\phi_3 \tag{2.115}$$

The important results will be summarized.
Stabilizing functions α_1, α_2 and the real control input u values:

$$\alpha_1 = -\frac{1}{g_1}(f_1 + \phi_1^T\hat{\theta}_1 - y_d - k_1\tilde{x}_1)$$

$$\alpha_2 = -\frac{1}{g_2}(f_2 + \phi_2^T\hat{\theta}_2 - \dot{x}_{2,c} - k_2\tilde{x}_2)$$

$$u = -\frac{1}{g_3}(f_3 + \phi_3^T\hat{\theta}_3 - \dot{x}_{3,c} - k_3\tilde{x}_3)$$

Adaptive laws for $\hat{\theta}_1, \hat{\theta}_2$ and $\hat{\theta}_3$

$$\hat{\theta}_1 = \Gamma_2 \int v_1\phi_1$$

$$\hat{\theta}_2 = \Gamma_2 \int v_2\phi_2$$

$$\hat{\theta}_3 = \Gamma_3 \int \tilde{x}_3\phi_3$$

Chapter 3

Disturbance Identification

3.1 System control in presence of disturbances

In the previous chapter, a control law for nonlinear system in the strict form with unknown parameters was obtained by using a Command Filtered Adaptive Backstepping approach. In the case where external disturbances $\tilde{\Delta} = [\Delta_1, \dots \Delta_n]$ enter the system, the previously developed algorithm cannot guarantee asymptotical stability or good tracking performance between the system output and the desired reference signal.
The following class of nonlinear system affected by external disturbances will be considered:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \phi_1^T(x_1)\theta_1 + \Delta_1 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 + \phi_2^T(x_1, x_2)\theta_2 + \Delta_2 \\ \vdots \\ \dot{x}_i = f_i(x_1, x_2, \dots x_i) + g_i(x_1, x_2, \dots x_i)x_{i+1} + \phi_i^T(x_1, x_2, \dots x_i)\theta_i + \Delta_i \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \dots x_n) + g_n(x_1, x_2, \dots x_n)u + \phi_n^T(x_1, x_2, \dots x_n)\theta_n + \Delta_n \\ y = x_1 \end{cases} \tag{3.1}$$

where Δ_i for $i = 1, \dots, n$ are external disturbances, $x_1, \dots, x_n \in \mathbf{R}$ represent the state variables of the system, $u \in \mathbf{R}$ is the control input and $f_1, f_2, \dots, f_i, \dots, f_n$, $g_1, g_2, \dots, g_i, \dots, g_n \in \mathbf{R}$, $\phi_1, \phi_2, \dots \phi_i \dots, \phi_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are known functions and $\theta_1, \theta_2, \dots, \theta_i, \dots, \theta_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are unknown parameters .
The objective is to globally stabilize the system and achieve the asymptotic tracking between the output of the system $y = x_1$ and the reference signal y_d. The control systems developed in the previous chapter cannot guarantee stability as they were developed considering a non-disturbance case. The following examples will show the degrade of the tracking performance caused by the introduction into the system of the external disturbances, which cannot be rejected by using the previously developed command filtered control.

The following second order nonlinear system is considered:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + \theta_1 x_1^2 + (1 + 0.1x_1^2)x_2 + \Delta_1 \\ \dot{x}_2 = x_1 x_2 + \theta_2(0.8\sin(x_1) + \sqrt{x_2}) + (2 + cos(x_1))u + \Delta_2 \\ y = x_1 \end{cases} \tag{3.2}$$

where Δ_1 and Δ_2 are external disturbances, θ_1 and θ_2 are unknown parameters, x_1,x_2 are the state variables and y is the output of the system.
The initial condition is $x_0 = [x_{10}, x_{20}]^T = [0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$. In the following, some plots of the tracking error $\tilde{x}_1 = x_1 - y_d$ will show the degradation in the tracking performance caused by the introduction of external disturbances in the controlled system without any disturbance rejection in the control system.

The following external disturbances signals Δ_1 and Δ_2 will be considered:

$$\Delta_1 = \delta_1[(cos(0.4t) + [0.5 + 0.5sin(0.8t)]) \tag{3.3}$$

$$\Delta_2 = \delta_2[0.7sin(t + \frac{\pi}{4}) + 0.5sin(t)] \tag{3.4}$$

where δ_1 ranges from 1 to 1.3 and δ_2 ranges from 1 to 1.5.

No disturbance case

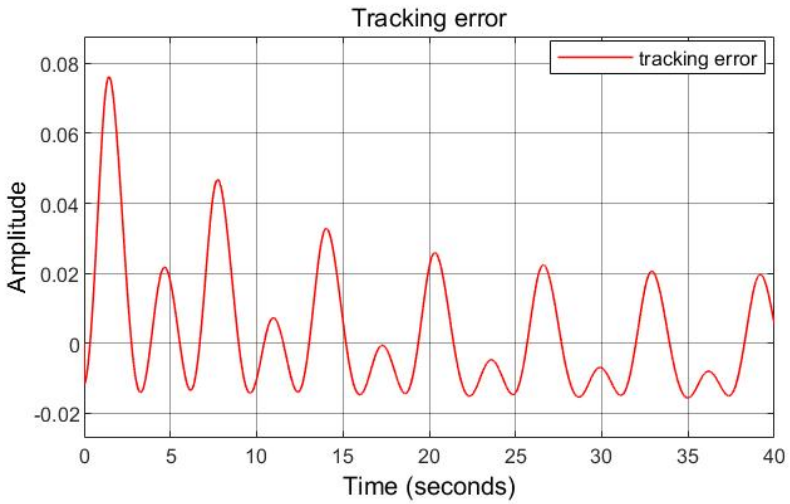

Figure 3.1: Error between reference signal y_d and the output $y = x_1$ in case where no external disturbances area added to the system

In the case where no disturbances are present in the system $(\Delta_1 = \Delta_2 = 0)$, the error $\tilde{x}_1$ has peak values at $t = 1s$ and $t = 7s$ caused by the unknown parameters θ_1 and θ_2 that makes the tracking more difficult until the Adaptive law estimates proper estimation values $\hat{\theta}_1$ and $\hat{\theta}_2$.
After $t = 10s$, the adaptive algorithm determines good estimations of θ_1 and θ_2 and so the error stays in the range of $\tilde{x}_1 = \pm 0.02$ when $t > 15s$.

Disturbance case

In the case where the disturbances are added to the system, the error $\tilde{x}_1$ has a peak value of $\tilde{x}_1 = 0.1$ at $t = 1.5s$ and for $t > 5$ it stays in the range of ± 0.06 because the adaptive algorithm finds satisfying estimated values $\hat{\theta}_1$ and $\hat{\theta}_2$.

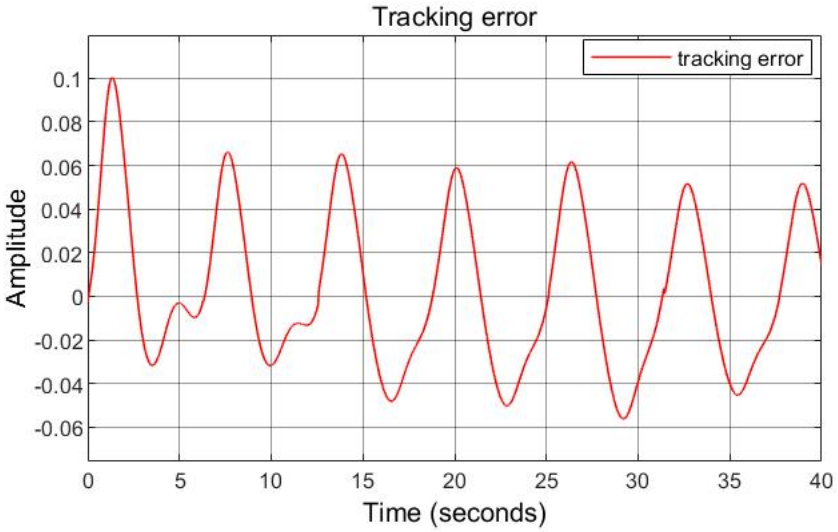

Figure 3.2: Error between reference signal y_d and the output $y = x_1$ in case where an external disturbances are added to the system

As shown in the previous plots, the developed Command Filtered Adaptive Backstepping algorithm guaranteed satisfying tracking performance in the "non disturbances case" but in the case where external disturbances were added to the system, the performance was deteriorated.
In this work, the case where a plain data set of the unknown disturbances is known is considered. The considered available data is affected by noise and a learning algorithm will be implemented to estimate the original signal based on the noisy data. Several Neural network based approaches have been implemented in backstepping control in some researches [11][12]. A known drawback of traditional Neural network approaches is in the curse of dimensionality that one has in the number of parameters, which recently lead the researchers to use a Neural network derived method known as Support vector Machine [14][16][26]. In this work, a Support Vector Machine based method called Support Vector Regression will be used to compute a regression of the disturbances based on the available noisy data. Then a disturbance rejection system will be implemented in a novel command filtered adaptive backstepping control to reject the original disturbances Δ_1 and Δ_2 by using the regression models by SVRs. In the following section, a brief introduction to Neural networks function estimation approaches will be presented.

3.2 Introduction to Neural Networks

An Artificial Neural Network (ANN) is a computational model that is inspired by the way biological Neural networks in the human brain process information. Artificial Neural networks have generated a lot of excitement in Machine Learning research and industry, thanks to many breakthrough results in speech recognition, computer vision and text processing [14][26]. A particular type of Artificial Neural network called Multi Layer Perceptron provides a nonlinear map between the input vector and the output vector and has been largely used in different engineering field to learn functions $f() : \mathbf{R}^n \rightarrow \mathbf{R}^m$ by a given dataset, where n is the number of dimensions of the input and m is the number of dimensions of the output. The basic unit of computation in a Neural network is the neuron, often called perceptron or node. It receives input from some other nodes, or from an external source and computes an output. Each input has an associated weight w_i, which is assigned on the basis of its relative importance to other inputs. The node applies a function f, known as Activation function, to the weighted sum of its inputs as shown below:

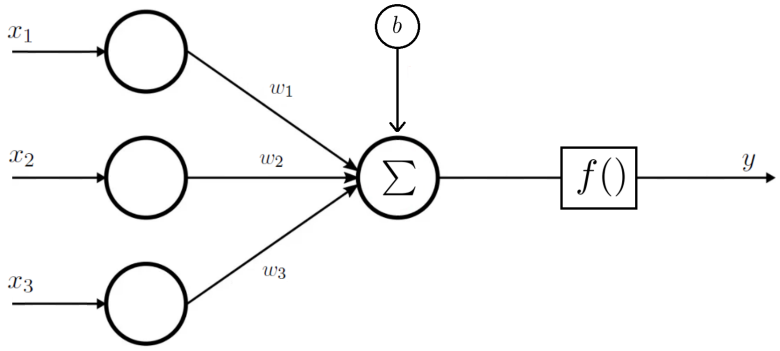

Figure 3.3: An example of Perceptron

The above network takes numerical inputs x_1, x_2 and x_3 with the associated weights w_1, w_2 and w_3. Additionally, there is another input with value 1 and weight b (called the Bias) associated with it. The bias term provides every perceptron with a constant value in addition to the normal inputs that the perceptron receives. The activation function f decides whether a neuron should be activated or not by calculating weighted sum and further adding bias to it. The activation function transform the input into a nonlinear function and so makes the node able to learn and perform more complex tasks.
Some example of activation functions are:

$$f(x) = sigmoid(x) = \frac{1}{1+e^{-x}} \qquad f(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

The sigmoid function takes a real-valued input x and squashes it to interval $[0, 1]$ while the hyperbolic tangent takes a real-valued input x and squashes it to the interval $[-1, 1]$.
A network composed by multiple perceptrons takes the name of Multi layer Perceptron networks and it will be discussed in the following section.

3.2.1 Multi Layer Perceptron

A Multi Layer Perceptron (MLP) is a class of Neural networks that contains multiple perceptrons, or nodes, arranged in layers. Nodes from adjacent layers have links between them, which have associated weights. While a single layer perceptron, explained before, can only learn linear functions, a multi layer perceptron can also learn non-linear functions.

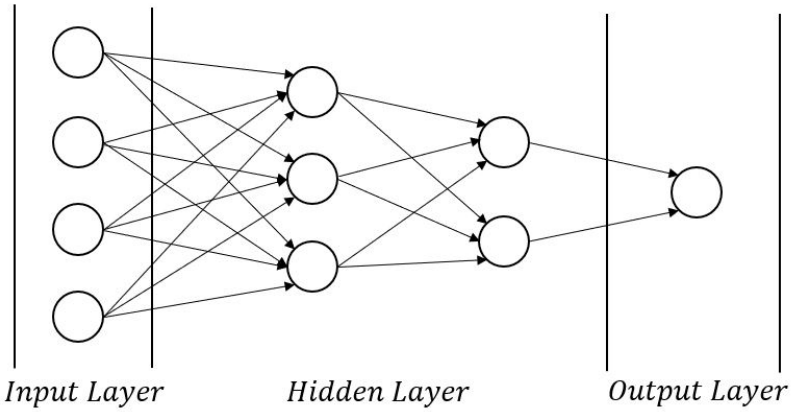

Figure 3.4: Example of Multi Layer Perceptron Network

In a Multi Layer Perceptron Neural Network, three types of layer can be defined:

- The Input layer acquire information from the outside world and passes it to the next layer, called Hidden layer.
- The Hidden layer has nodes that perform computations and transfer information from the input nodes to the output nodes.
- The Output layer receives the information from the Hidden layer and transfer them to the outside world.

Given a set of input $X = (x_1, x_2, \ldots, x_n)$ and a target value y, a Multi Layer Perceptron can learn the relationship between the input and the target, for either classification or regression. Neural networks are capable of adaptation to given data even when the input data set contains noise or missing values. In order to achieve this result, Neural networks use learning algorithms to update the values of the weights and biases, based on the error between the actual output of the network and a desired output. One of the most used learning algorithm is called "Gradient Descend". First a Loss function is defined between the desired and the actual output of the network, then the algorithm calculates its derivatives respect to the bias and the weights and solve a minimum optimization problem, in order to find the best values of bias and weights that will minimize the Loss function. Despite many of these advances, there still remain a number of weak points for these classical Neural networks approaches, namely, the existence of many local minima solutions and the issue of choosing the number of hidden units and hidden layers [14][26]. Major advances have been obtained by means of a new class of Neural networks called Support Vector Machines (SVMs), originally introduced by Vapnik within the area of statistical learning theory and structural risk minimization to solve classification problems [13], which will be introduced in the following section.

3.3 Introduction to Support Vector Machines

Support Vector Machines (SVMs) are a set of supervised learning methods used for classification and regression. Support Vector Machines (SVMs) were first heard in 1992, introduced by Boser, Guyon, and Vapnik and originally used to solve classification problems [14]. To perform a classification, Support Vector Machines need to be first trained with some input data with a known classification by using statistical learning theory. In the training process, the input data is projected into a higher dimensional space where a hyperplane that can classify the projected data is considered instead of a difficult and heavy computing nonlinear function in the original lower dimensional space. The optimization problem that will solve the classification is first evaluated in a primal weight space, where one can work in the higher dimensional feature space, but one solves the resulting problem in the dual space, in which the solution is the same as the primal problem but results easier to calculate. Once the training process is finished, the Support Vector Machine can classify new input data. Support Vector Machine methods have been widely used to solve classification and regression problems and have given comparable accuracy to sophisticated Neural networks in some applications such as hand writing analysis, face analysis and so forth, especially for pattern classification and regression based applications [13][14]. Moreover, Support Vector Machine methods solve convex optimizing problems, typically quadratic programming, where standard Neural networks methods solve optimization problems where the solution is a local minima and not necessary the optimal solution [13][26]. Support Vector Machine were originally developed to solve classification problems, but recently a regression variant called Support Vector Regression has been largely used to solve regression problems and will be introduced later in this work.

3.4 Linear SVM Classifier

First the Support Vector Machine will be introduced for classification problem, later in this work a Regression model will be derived. Given a set of training examples, each marked as belonging to one or the other of two classes, a Support Vector Machine algorithm defines a classifier that can classify new data.

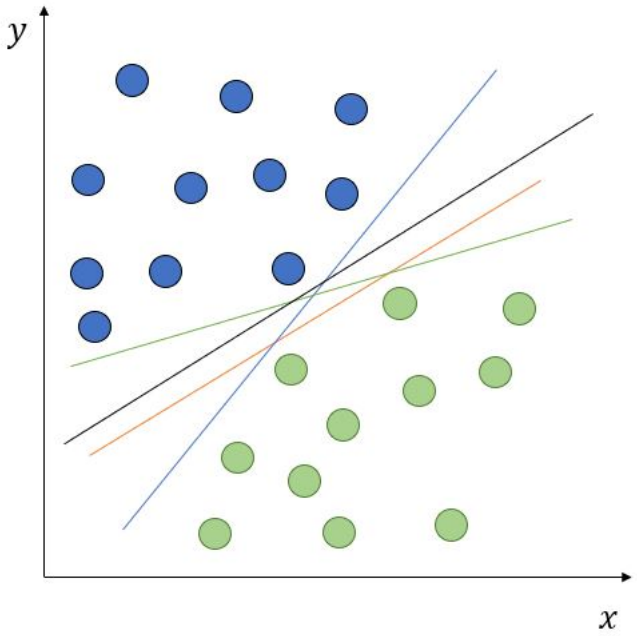

Figure 3.5: Example of Classification problem

To separate the two classes (blue and green dots), there are many possible hyperplanes that can correctly separate the two classes. Classification algorithms aims to define which is the best hyperplane that can separate the two classes. The dimension of the hyperplane depends upon the dimension of the data. For example, if the dimension of the input data is $n = 2$, then the hyperplane is just a line, if the dimension of input data is $n = 3$, then the hyperplane becomes a two-dimensional plane and so on. A Support Vector Machine Classifier can be implemented to find the "best" hyperplane $y = w^T \varphi(x) + b$ that separates the two classes. In order to do so, one would choose a hyperplane that maximizes the margin between the different classes as the further from the hyperplane, giving a greater chance of new data being classified correctly. Therefore, the data points has to be as far away from the hyperplane as possible, while still being on the correct side of it. The vector points closest to the hyperplane are known as the Support Vector points because only these points are contributing to the result of the algorithm. If a data point is not a Support Vector, removing it has no effect on the model. On the other hands, deleting the Support Vectors will then change the position of the hyperplane. In order to find the best margin that divides the two classes, only the Support Vectors will be considered.

The classification problem for two classes can be formulated as follows:

a training set $\{x_k, y_k\}_{k=1}^{N}$ with input data $x_k \in \mathbf{R}^2$ and corresponding binary class labels $y_k \in \{-1, +1\}$, find the best separator $y = w^T x + b$ that allows the best classification between the two classes, labeled by $\{-1, +1\}$:

$$\begin{aligned} w^T x_k + b \geq +1 \qquad & if \quad y_k = +1 \\ w^T x_k + b \leq -1 \qquad & if \quad y_k = -1 \end{aligned}$$

which can be written as:

$$y_k[w^T x_k + b] \geq +1 \qquad k = 1, \ldots, N.$$

In the figure below, the classification problem for two different classes of data (blue and green dots) is shown. In the considered case the hyperplane is "linear" (a line) and takes the form: $y = w^T x + b$

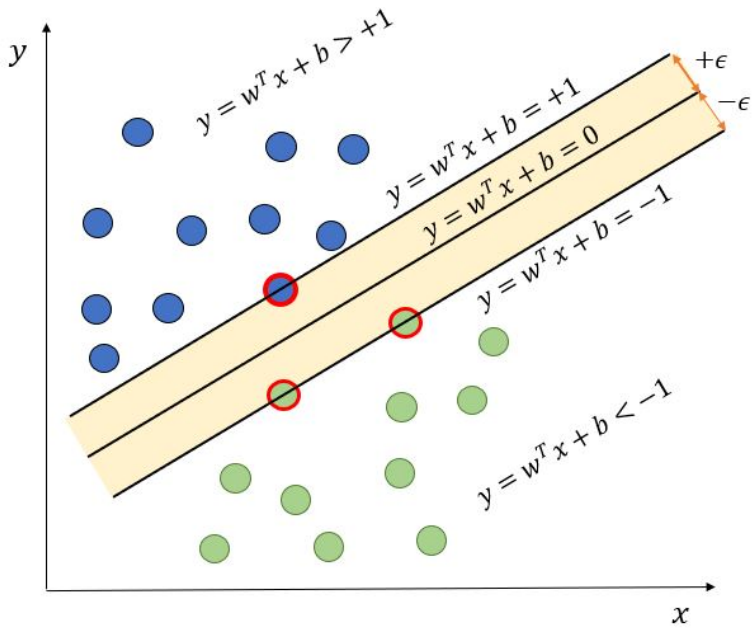

Figure 3.6: Linear SVM

The objective is to maximize the margin 2ϵ between the delimiting functions $w^T x_k + b = +1$ and $w^T x_k + b = -1$, which delimits the two classes.

The geometrical distance between the two delimiting functions is defined as follows:

$$2\epsilon = \frac{2}{\|w\|} = \frac{2}{\sqrt{w^T w}} \tag{3.5}$$

So maximizing the margin $\frac{2}{\|w\|}$ means minimizing the quantity $\frac{1}{2}\sqrt{w^T w}$. Then the following minimization problem can be formulated [14]:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} w^T w \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 \end{aligned}$$

As most of the real-world data are not fully linearly separable, the slack variables ξ_k are introduced in order to allow some data to fall off the margin with a penalization.

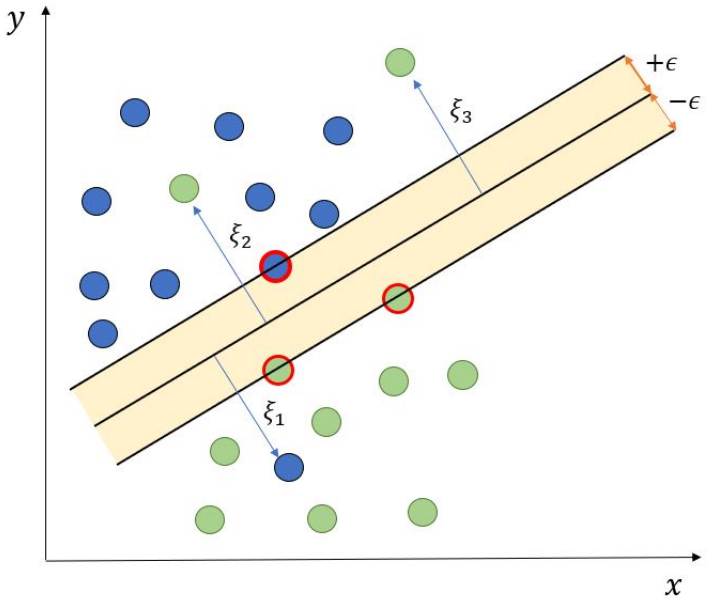

Figure 3.7: Linear SVM with slack variables

The slack variables aim to avoid the over fitting problem, where an unnecessary difficult nonlinear function is used to separate different classes of data that could be separated by a linear separator that allows errors with penalization.

The overfitting problem is shown in the figure below. On the left figure, a linear separator has been used to classify the two sets of input data. On the other hand, on the right figure a high computational nonlinear separator was used to separate the two classes.

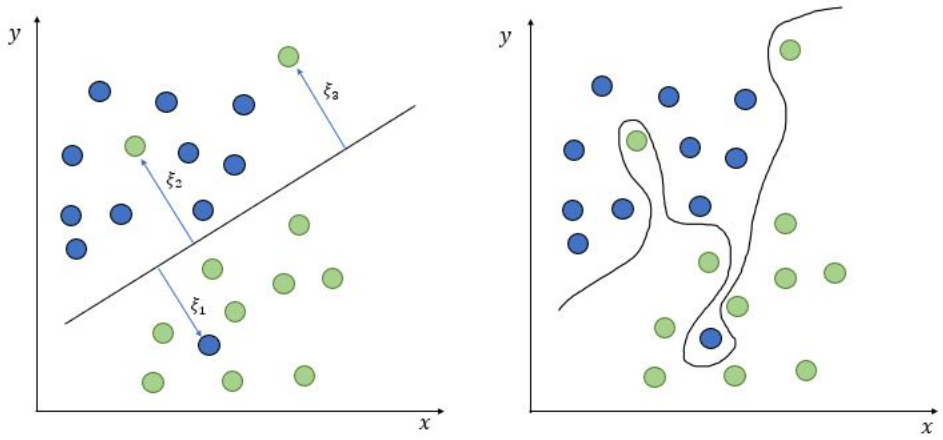

Figure 3.8: Example of over fitting problem

The previously defined optimization problem is reformulated including the slack variables ξ_i:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & y_i(w^T x_i + b) + \xi_i \geq 1 \quad i = 1, \ldots, N \\ & \xi_i \geq 0 \qquad\qquad\qquad\quad i = 1, \ldots, N \end{aligned}$$

In the general case, the relationship between the two classes is nonlinear and so no linear hyperplane can separate the two classes of data even if slack variables are introduced. An example is shown in the figure below:

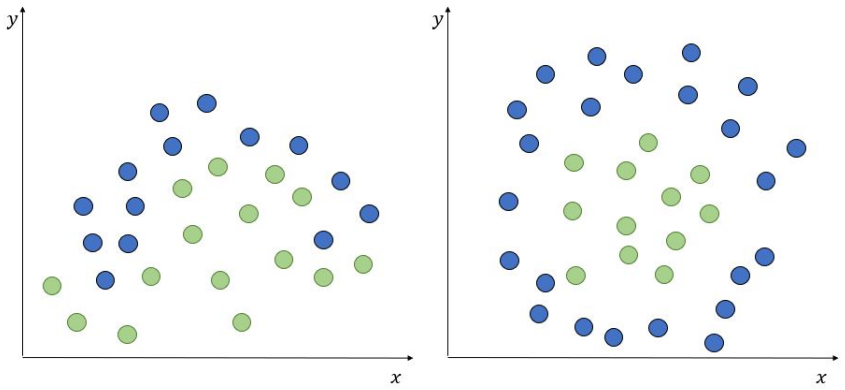

Figure 3.9: Example of non-linearly separable data

To solve this class of non-linear classification problems, SVMs algorithm define a function $\varphi() : \mathbf{R}^n \rightarrow \mathbf{R}^m$, where $n > m$, maps the input space to a so-called higher dimensional feature space where the data become linearly separable. This class of classification problems will be discussed in the next section.

3.5 Nonlinear SVM Classifier

Given a training set $\{x_k, y_k\}_{k=1}^{N}$ with input data $x_k \in \mathbf{R}^n$ and corresponding binary class labels $y_k \in \{-1, +1\}$, the SVM classifier formulation starts from the following assumption:

$$\begin{aligned} w^T\varphi(x_k) + b \geq +1 \qquad & if \quad y_k = +1 \\ w^T\varphi(x_k) + b \leq -1 \qquad & if \quad y_k = -1 \end{aligned}$$

Which can be written as:

$$y_k[w^T\varphi(x_k) + b] \geq +1 \qquad k = 1, \ldots, N.$$

Here $\varphi() : \mathbf{R}^n \rightarrow \mathbf{R}^m$, where $n > m$ is a nonlinear function that maps the input space to a so-called higher dimensional feature space (See figure below).

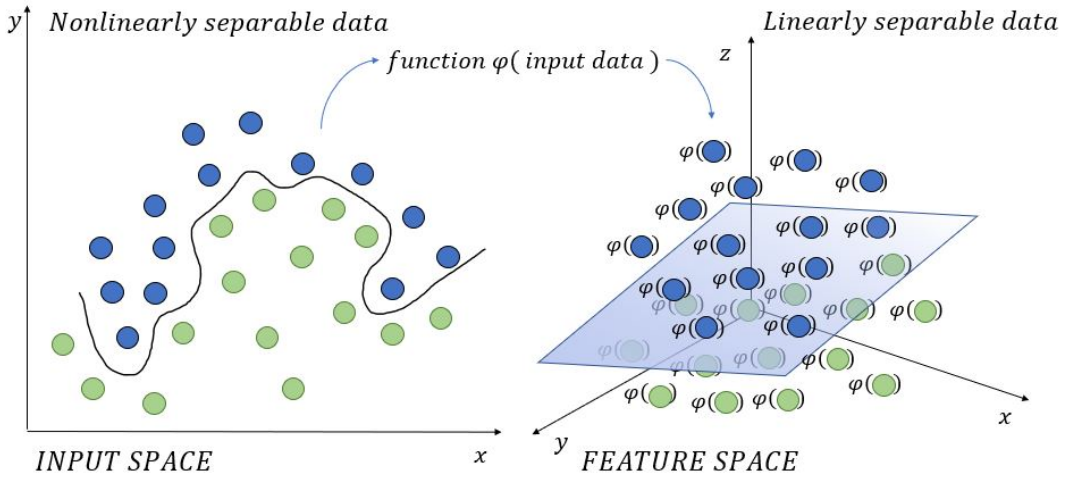

Figure 3.10: Mapping the Input data to the higher dimensional space by using the map $\varphi()$

It is important to note that the dimension m of this space is only defined in an implicit way and it can be infinite dimensional [14]. The term b denotes a bias term. One defines the following optimization problem:

$$\begin{aligned} \min_{w,b,\xi} \quad & J(w, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & y_i(w^T\varphi(x_i) + b) \geq 1 - \xi_i \quad i = 1, \ldots, N \\ & \xi_i \geq 0 \qquad\qquad\qquad\qquad i = 1, \ldots, N \end{aligned}$$

The minimization of $\|\mathbf{w}\|^2$ corresponds to the maximization of the margin between the two classes. C is a positive real constant known as "regularization parameter" and should be considered as a tuning parameter in the algorithm.

The Lagrangian for this problem is given by

$$L(w,b,\xi;\alpha,v) = J(w,\xi) - \sum_{i=1}^{N} \alpha_i \{y_i(w^T\varphi(x_i)+b) - 1 + \xi_i\} - \sum_{i=1}^{N} v_i\xi_i \tag{3.6}$$

with Lagrange multipliers $\alpha_i \geqslant 0$, $v_i \geqslant 0$ where $i = 1,\ldots,N$.
The solution of optimization problem is characterized by the saddle point of the Lagrangian [13], which is defined as follows:

$$\max_{\alpha,v} \min_{w,b,\xi} L(w,b,\xi;\alpha,v) \tag{3.7}$$

One obtains

$$\frac{\partial L}{\partial w} = 0 \quad \longrightarrow w = \sum_{i=1}^{N} \alpha_i y_i \varphi(x_i), \tag{3.8}$$

$$\frac{\partial L}{\partial b} = 0 \quad \longrightarrow \sum_{i=1}^{N} \alpha_i y_i = 0, \tag{3.9}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \quad \longrightarrow 0 \leq \alpha_i \leq C, \quad i = 1,\ldots,N \tag{3.10}$$

By replacing w in the Lagrangian, one obtains the following dual problem (in the Lagrange multipliers α), which is the quadratic programming problem:

$$\max_{\alpha} \quad Q(\alpha) = -\frac{1}{2}\sum_{i,k=1}^{N} y_i y_k \varphi(x_i)^T\varphi(x_k)\alpha_i\alpha_k + \sum_{i=1}^{N}\alpha_i \tag{3.11}$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \qquad 0 \leq \alpha_i \leq C, \quad i = 1,\ldots,N \tag{3.12}$$

One can choose a Kernel function:

$$K(x_i,x_k) = \varphi(x_i)^T\varphi(x_k) \tag{3.13}$$

so that there is no need to compute $\phi(x_k)$, which can be infinite dimensional [14]. This operation takes the name of Kernel Trick, and it will be explained in the next section.
Finally, in the dual space, the nonlinear SVM classifier becomes:

$$y(x) = sign\left[\sum_{i=1}^{N} \alpha_i y_i K(x,x_i) + b\right] \tag{3.14}$$

where $\alpha_i > 0$ follow from the QP problem. The non-zero Lagrange multipliers α_i are called Support Values. The corresponding data points are called Support Vectors and are located close to the hyperplane. The bias term b follows from the called Karush–Kuhn–Tucker (KKT) conditions (Karush 1939, Kuhn and Tucker 1951) [13], and is not further discussed in this work.

3.6 Kernel Functions

Formally, a Kernel function is any function that satisfies Mercer theorem, which states that a given function $k(\mathbf{x_i}, \mathbf{x_j})$ is a kernel function if the associated Kernel Matrix K where $K_{i,j} = k(\mathbf{x_i}, \mathbf{x_j})$ is positive semi-definite, i.e:

$$c^T Kc = \sum_i \sum_j c_i c_j K_{i,j} = \sum_i \sum_j c_i c_j \phi(\mathbf{x_i})^T \phi_(\mathbf{x_j}) =$$

$$= \left(\sum_i \phi(\mathbf{x_i}) \right) \left(\sum_j \phi(\mathbf{x_j}) \right) = \left\| \sum_j \phi(\mathbf{x_j}) \right\|^2 \geq 0$$

3.6.1 Kernel Trick

The Kernel Trick is a mathematical tool that allows to get the same result of an inner product in a high dimensional space but operating in a low dimensional space. In other words, for input variable $\mathbf{x_i}$ and $\mathbf{x_j}$ in the input space A, certain functions $K(\mathbf{x_i}, \mathbf{x_j})$ can be expressed as inner product in another space B, where $dim(B) > dim(A)$. The function $k : A \times A \rightarrow \mathbb{R}$ is often referred to as Kernel or Kernel Function. A Kernel Function is a function which represents a inner product in an extended space. Given the feature map $\phi : x_i \rightarrow \phi(\mathbf{x_i})$ then the computation can be made much simpler using the kernel defined as.

$$k(\mathbf{x_i}, \mathbf{x_j}) = \langle \phi(\mathbf{x_i}) \phi(\mathbf{x_j}) \rangle_B$$

There exists different Kernel functions, which are chosen depending on the problem at hand. Two examples of Kernel Functions will be introduced.

Polynomial Kernel

With n original features and d degree of polynomial, the polynomial kernel yields n^d extended features

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + c)^d$$

Gaussian radial basis function (RBF)

The RBF Kernel is a general-purpose kernel that is generally used when there is no prior knowledge about the input data.

$$K(x_i, x_k) = \exp\left(-\gamma \|x_i - x_k\|^2 \right)$$

where the parameter $\gamma > 0$ is called learning rate and defines the influence of a single training example.

3.7 Support Vector Regression

Support Vector Machine can also be used as a regression method, while maintaining all the main features that characterize the algorithm and it takes the name of Support Vector Regression [14]. The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. In the case of regression, the parameter ϵ is set as the tolerance approximation that a SVM would have requested from the problem. The main idea is to choose the hyperplane that maximizes the margin, considering that the some error is tolerated by the means of the slack variables to avoid overfitting.
Consider a given training set $\{x_k, y_k\}_{k=1}^N$ with input data $x_k \in \mathbf{R}^n$ and output data $y_k \in \mathbf{R}$. The following model is taken

$$f(x) = w^T \varphi(x) + b$$

where the input data are projected to a higher dimensional feature space with the map $\varphi()$ as in the classifier case. In Empirical Risk Minimization one optimizes the cost function containing Vapnik's ϵ-insensitive loss function which is defined as follows:

$$|y - f(x)|_\epsilon = \begin{cases} 0, & \text{if } |y - f(x)| \leq \epsilon, \\ |y - f(x)| - \epsilon, & \text{otherwise.} \end{cases}$$

The optimization problem is defined as follows:

$$\begin{aligned} \min_{w,b,\xi} \quad & J(w, \xi, \xi^*) = \frac{1}{2} w^T w + c \sum_{i=1}^{N} (\xi_i + \xi_i^*) \\ \text{subject to} \quad & y_i - w^T \varphi(x_i) - b \leq \epsilon + \xi_i \\ & w^T \varphi(x_i) + b - y_i \leq \epsilon + \xi_i^* \\ & \xi, \xi^* \geq 0 \quad i = 1, \ldots, N \end{aligned}$$

where in this case ϵ is the accuracy that one demands for the approximation, which can be violated by means of the slack variables ξ and ξ^*.
The conditions for optimality lead to the following dual problem in the Lagrange multipliers α, α^* [14]:

$$\begin{aligned} \max_{\alpha,\alpha_*} \quad & -\frac{1}{2} \sum_{i,k=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_k - \alpha_k^*) K(x_i, x_k) - \epsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i (\alpha_i - \alpha_i^*) \\ \text{subject to} \quad & \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \end{aligned}$$

The kernel trick $K(x_k, x_i) = \phi(x_k)^T \phi(x_i)$ is again applied in the formulation of this quadratic programming problem. Finally, the SVM for nonlinear function estimation model can be derived:

$$f(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) K(x, x_i) + b \tag{3.15}$$

3.8 Least Squares Support Vector Regression

In this work, a Least Square version of the standard Support Vector Regression will be implemented. The Least Squares Support Vector Regression (LS-SVR) uses the squared-error loss function where the standard SVR uses the so-called ϵ insensitive absolute-error loss function.

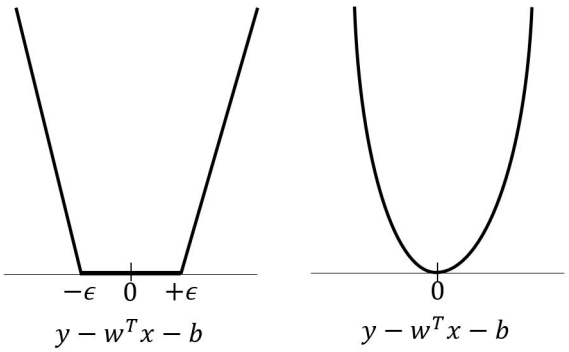

Figure 3.11: On the left, the Loss function for standard SVR, on the right the quadratic Loss function in the LS-SVR

Least-squares SVR loses the sparsity of standard SVR originated from the insensitivity of its loss function to any errors that are smaller than ϵ. This means that in LS-SVR, all the training data are considered in the target function, where in the traditional SVR method, only the Support Vectors are considered. In standard SVR, the dual problem is a quadratic programming problem that needs an iterative solution. On the other hand, the LS-SVR leads to a problem that presents an analytic (i.e. non-iterative) solution, which therefore is computationally more efficient compared to the non least square variant.
The LS-SVR model for function estimation has the following representation in feature space:

$$y(x) = w^T \varphi(x) + b$$

Where $x \in \mathbf{R}^n$, $y \in \mathbf{R}$. The use of nonlinear mapping $\phi()$ is similar to the classifier case. Given a training set $\{x_k, y_k\}_{k=1}^N$ one defines now the optimization problem

$$\min_{w,b,\xi} \quad J(w,\xi) = \frac{1}{2} w^T w + C \frac{1}{2} \sum_{i=1}^{N} e_i^2 \tag{3.16}$$

subject to

$$y_i = w^T \varphi(x_i) + b + e_i, \quad i = 1, \ldots, N$$

One constructs the following Lagrangian[13]:

$$L(w,b,e;\alpha) = J(w,e) - \sum_{i=1}^{N} \alpha_i (w^T \varphi(x_i) + b + e_i - y_i) \tag{3.17}$$

where α_i are Lagrange multipliers.

The conditions for optimality are given by

$$\frac{\partial L}{\partial w} = 0 \quad \longrightarrow w = \sum_{i=1}^{N} \alpha_i \varphi(x_i), \tag{3.18}$$

$$\frac{\partial L}{\partial b} = 0 \quad \longrightarrow \sum_{i=1}^{N} \alpha_i = 0, \tag{3.19}$$

$$\frac{\partial L}{\partial e_i} = 0 \quad \longrightarrow \alpha_i = C e_i \quad i = 1, \ldots, N \tag{3.20}$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \quad \longrightarrow w^T \varphi(x_i) + b + e_i - y_i, \quad i = 1, \ldots, N \tag{3.21}$$

with solution:

$$\left[\begin{array}{c|c} 0 & \vec{1}^{\mathrm{T}} \\ \hline \vec{1} & \Omega + \mathrm{C}^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right]$$

where $y = [y_1, \ldots, y_N]$, $I = [1, \ldots, 1]$ and $\alpha = \alpha_1, \ldots, \alpha_N$.
From application of the Mercer condition one obtains

$$\begin{aligned} \Omega_{i,k} &= \varphi(x_i)^T \varphi(x_k), \qquad i, k = 1, \ldots, N \\ &= K(x_i, x_k) \end{aligned}$$

The resulting LS-SVM model for function estimation becomes

$$y(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i) + b \tag{3.22}$$

where α_k, b are the solution to the linear system.

The Least Squares Support Vector Regression has been widely used in the control field as a regression model for its effectiveness and simplicity in the formulation [17][27][28]. In order to obtain the regression model 3.22, a linear set of equation had to be solved, instead of a quadratic programming problem as in the standard SVR approach. In the next chapter, the external disturbances Δ_i will be estimated by using the LS-SVR regression model in equation 3.22. The approximated functions will be then used in the design of a novel command filtered adaptive backstepping control to reject the original disturbances Δ_i while guaranteeing the desired tracking performance and global stability.

Chapter 4

Control System with Disturbance Compensation

In this chapter, a regression model by Least Squares Support Vector Regression will be used to estimate the unknown disturbances. Moreover, the estimated functions by LS-SVR will be implemented in the control system to reject the external disturbances. The following class of nonlinear system affected by external disturbances will be considered:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \phi_1^T(x_1)\theta_1 + \Delta_1 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 + \phi_2^T(x_1, x_2)\theta_2 + \Delta_2 \\ \vdots \\ \dot{x}_i = f_i(x_1, x_2, \dots x_i) + g_i(x_1, x_2, \dots x_i)x_{i+1} + \phi_i^T(x_1, x_2, \dots x_i)\theta_i + \Delta_i \\ \vdots \\ \dot{x}_n = f_n(x_1, x_2, \dots x_n) + g_n(x_1, x_2, \dots x_n)u + \phi_n^T(x_1, x_2, \dots x_n)\theta_n + \Delta_n \\ y = x_1 \end{cases} \tag{4.1}$$

where Δ_i for $i = 1, \dots, n$ are external disturbances, $x_1, \dots, x_n \in \mathbf{R}$ are the state variables of the system, $u \in \mathbf{R}$ is the control input, $f_1, f_2, \dots, f_i, \dots, f_n \in \mathbf{R}$, $g_1, g_2, \dots, g_i, \dots, g_n \in \mathbf{R}$, $\phi_1, \phi_2, \dots \phi_i \dots, \phi_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are known functions and $\theta_1, \theta_2, \dots, \theta_i, \dots, \theta_n \in \mathbf{R}^T$ for $i = 1, \dots, n$ are unknown parameters. The objective is to globally stabilize the system and achieve the asymptotic tracking between the output of the system $y = x_1$ and the reference signal y_d. As shown in chapter 3, the disturbances deteriorate the performance of the control system and so a LS-SVR estimation model of the disturbances will be used to estimate the unknown disturbance. To train the LS-SVR models, an available set of noisy data $\{x_k, y_k\}_{k=1}^N$ of the external disturbances Δ_i for $i = 1, \dots, n$ will be used.
The LS-SVR identification model $y(x)$ obtained in chapter 4 is:

$$y(x) = \sum_{k=1}^{N} \alpha_k K(x, x_k) + b \tag{4.2}$$

where x_k are the training data of the disturbance, α_k for $i = 1, \dots, N$ and b are the solution of the optimization problem discussed in the previous chapter and $K(x, x_k)$ is a Kernel Function.

In order to obtain the solutions α_k for $i = 1, \ldots, N$ and b, the following linear system, resulted from the optimization problem, will be considered:

$$\left[\begin{array}{c|c} 0 & \vec{1}^{\mathrm{T}} \\ \hline \vec{1} & \Omega + \mathrm{C}^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right]$$

where $y = [y_1, \ldots, y_N]$, $I = [1, \ldots, 1]$, C is a tuning parameter and Ω is:

$$\begin{aligned} \Omega_{i,k} &= \varphi(x_i)^T \varphi(x_k), \qquad i, k = 1, \ldots, N \\ &= K(x_i, x_k) \end{aligned}$$

where $K(x_k, x_l)$ is a Kernel Function.
For this work, the Radial Basis Function Kernel was chosen

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma \left\| \mathbf{x_i} - \mathbf{x_j} \right\|^2\right)$$

where $\gamma > 0$ is a tuning parameter.
In the next section, the command filtered adaptive backstepping control developed in chapter 3 will be extended to the case where the controlled system is affected by three different disturbances Δ_1, Δ_2 and Δ_3.

Third order system

The following third order nonlinear system will be considered:

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \phi_1^T(x_1)\theta_1 + \Delta_1 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 + \phi_2^T(x_1, x_2)\theta_2 + \Delta_2 \\ \dot{x}_3 = f_3(x_1, x_2, x_3) + g_3(x_1, x_2, x_3)u + \phi_3^T(x_1, x_2, x_3)\theta_3 + \Delta_3 \\ y = x_1 \end{cases} \tag{4.3}$$

where $x_1, x_2, x_3 \in \mathbf{R}$ represent the state variables of the system, $u \in \mathbf{R}$ is the control input, $f_1, f_2, f_3 \in \mathbf{R}$, $g_1, g_2, g_3 \in \mathbf{R}$ and $\phi_1, \phi_2, \phi_3 \in \mathbf{R}^T$ are known functions, $\theta_1, \theta_2, \theta_3 \in \mathbf{R}^T$ are unknown parameters and Δ_1, Δ_2 and Δ_3 are unknown disturbances.
The objective is to globally stabilize the system and achieve the asymptotic tracking between the output of the system $y = x_1$ and the reference signal y_d.

First, the following function $\psi_1(x)$, $\psi_2(x)$ and $\psi_3(x)$, used as estimations of the three unknown disturbances Δ_1, Δ_2 and Δ_3 respectively, needs to be trained and then tested:

$$\psi_1(x_{t,1}) = \sum_{k=1}^{N1} \alpha_{k,1} K_1(x_{t,1}, x_{k,1}) + b_1 \tag{4.4}$$

$$\psi_2(x_{t,2}) = \sum_{k=1}^{N2} \alpha_{k,2} K_2(x_{t,2}, x_{k,2}) + b_2 \tag{4.5}$$

$$\psi_3(x_{t,3}) = \sum_{k=1}^{N3} \alpha_{k,3} K_3(x_{t,3}, x_{k,3}) + b_3 \tag{4.6}$$

where $x_{k,1}$,$x_{k,2}$ and $x_{k,3}$ are the training data of the disturbances, $\alpha_{k,1}$ for $i = 1, \ldots, N1$, $\alpha_{k,2}$ for $i = 1, \ldots, N2$, $\alpha_{k,3}$ for $i = 1, \ldots, N3$ and b_1,b_2 and b_3 are the solution of the optimization problem discussed in the previous chapter and $K_1(x_{t,1}, x_{k,1})$, $K_2(x_{t,2}, x_{k,2})$ and $K_3(x_{t,3}, x_{k,3})$ are Kernel Functions. The variables $x_{t,1}$,$x_{t,2}$ and $x_{t,3}$ are the testing data used to validate the performance of the SVR models once they are trained. Both the training data and testing data are acquired for a specific accumulation time t_{acc}. The testing dataset is generally smaller than the training dataset. In order to train the SVR models, for each SVR model an optimization problem (3.16) discussed in the previous chapter is considered. For each of the three optimization problems, the solutions α_i for $i = 1, \ldots, N$ and b of the following linear system, resulted from the optimization problem, will be considered:

$$\left[\begin{array}{c|c} 0 & \vec{1}^{\mathrm{T}} \\ \hline \vec{1} & \Omega + \mathrm{C}^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right]$$

where $y = [y_1, \ldots, y_N]$ are training data of the function that needs to be estimated, $I = [1, \ldots, 1]$, C is a tuning parameter and Ω is:

$$\begin{aligned} \Omega_{i,k} &= \varphi(x_i)^T \varphi(x_k), \qquad i,k = 1, \ldots, N \\ &= K(x_i, x_k) \end{aligned}$$

where $K(x_i, x_k)$ is a Kernel Function.
The Radial Basis Function Kernel was chosen for all the three identification models.

$$K_1(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma_1 \left\| \mathbf{x_i} - \mathbf{x_j} \right\|^2\right)$$

$$K_2(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma_2 \left\| \mathbf{x_i} - \mathbf{x_j} \right\|^2\right)$$

$$K_3(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma_3 \left\| \mathbf{x_i} - \mathbf{x_j} \right\|^2\right)$$

where $\gamma_1, \gamma_2, \gamma_3, > 0$ are tuning parameters.
For the first training of the estimation functions $\psi_1(x)$, $\psi_2(x)$ and $\psi_3(x)$, the SVR

parameters C, also called regularization parameter, and γ, the learning rate of the RBF function, are set to a initial value. Once the training is finished, the SVR models will be tested with the testing data.
After the testing, to evaluate the performance of the SVR, in a real world case study the tuning parameters of the SVR are chosen in order to guarantee the best tracking between the output of the system and the reference signal and so obtain the lowest error between the two signals. In the simulations, the unknown disturbance functions that generates the noisy samples are available, and they will be used to evaluate the performance of the SVRs. The parameters of the SVRs will be then tuned in order to guarantee the best fit between the approximated functions by the SVRs and the real noise-less disturbances. Once the results are satisfying, the functions $\psi_1(x)$, $\psi_2(x)$ and $\psi_3(x)$ can be used in the control laws of a novel control system in order to compensate the original disturbances Δ_1, Δ_2 and Δ_3. In the following section, a command filtered adaptive backstepping algorithm similar to the one developed in chapter 2 will be implemented for the third order system (4.3) considered in this section. The difference with the previously developed command filtered adaptive backstepping control system is that the functions Δ_1, Δ_2 and Δ_3 are added to the controlled system and the SVR models $\psi_1(x), \psi_2(x)$ and $\psi_3(x)$ will be used in the control laws to reject the original disturbances.

STEP 1

The backstepping algorithm starts by defining the tracking error between the output $y = x_1$ and the reference signal $y = y_d$ as:

$$\tilde{x_1} = x_1 - y_d \tag{4.7}$$

To derive the stabilizing function α_1 that will act as a virtual input to stabilize the first system and guarantee $x_1 = y_d$ asymptotically, the Lyapunov Function $V_1(\tilde{x_1})$ and its derivative will be derived.
First, derivative of the tracking error $\tilde{x_1}$ is considered:

$$\dot{\tilde{x}}_1 = \dot{x}_1 - \dot{y}_d = f_1 + g_1 x_2 + \phi_1^T \theta_1 + \psi_1(x) - \dot{y}_d \tag{4.8}$$

Since θ_1 is unknown, an estimate value $\hat{\theta}_1$ of θ_1 is considered and then an the estimation law for $\hat{\theta}_1$ will be determined. Moreover, an estimation error $\tilde{\theta}$ between the real value of θ_1 and its estimation $\hat{\theta}_1$ is derived as follows

$$\tilde{\theta_1} = \theta_1 - \hat{\theta}_1 \tag{4.9}$$

The following Lyapunov Function $V_1(\tilde{x_1}, \tilde{\theta_1})$, which also includes the estimation error $\tilde{\theta_1}$, will be considered:

$$V_1(\tilde{x_1}, \tilde{\theta_1}) = \frac{1}{2}\tilde{x_1}^2 + \frac{1}{2}\tilde{\theta_1}^T \Gamma_1^{-1} \tilde{\theta_1} \tag{4.10}$$

where Γ_1 is a diagonal positive definite matrix.
The derivative of the Lyapunov Function $V_1(\tilde{x_1}, \tilde{\theta_1})$ will be now calculated:

$$\dot{V}_1(\tilde{x_1}, \tilde{\theta_1}) = \tilde{x_1}\dot{\tilde{x}}_1 + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 = \tilde{x_1}(f_1 + g_1 x_2 + \phi_1^T \theta_1 + \psi_1(x) - \dot{y}_d) + \tilde{\theta}_1^T \Gamma_1^{-1} \dot{\tilde{\theta}}_1 \tag{4.11}$$

In order to achieve $\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) < 0$ and so $\tilde{x}_1 = 0$ asymptotically, one would choose the following stabilizing function $x_2 = \alpha_1$:

$$\alpha_1 = \alpha_1^* = -\frac{1}{g_1}(f_1 + \phi_1^T\theta_1 + \psi_1(x) - \dot{y}_d - k_1\tilde{x}_1) \tag{4.12}$$

where $k_1 > 0$ is a tuning parameter.
Because θ_1 is not available, one chooses the following stabilizing function α_1 where instead of θ_1, its estimated value $\hat{\theta}_1$ will be used:

$$\alpha_1 = -\frac{1}{g_1}(f_1 + \phi_1^T\hat{\theta}_1 + \psi_1(x) - \dot{y}_d - k_1\tilde{x}_1) \tag{4.13}$$

By substituting $x_2 = \alpha_1$ in equation 4.11:

$$\begin{aligned}\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) &= \tilde{x}_1(-k_1\tilde{x}_1 + \phi_1^T\theta_1 - \phi_1^T\hat{\theta}_1) + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 \\ &= \tilde{x}_1(-k_1\tilde{x}_1 + \phi_1^T\tilde{\theta}_1) + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 \\ &= -k_1\tilde{x}_1^2 + \tilde{\theta}_1^T(\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{x}_1\phi_1)\end{aligned} \tag{4.14}$$

by choosing the following value for $\dot{\tilde{\theta}}_1$:

$$\dot{\tilde{\theta}}_1 = -\Gamma_1\tilde{x}_1\phi_1 \tag{4.15}$$

The derivative of the Lyapunov Function $\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1)$ becomes:

$$\dot{V}_1(\tilde{x}_1, \tilde{\theta}_1) = -k_1\tilde{x}_1^2 < 0 \tag{4.16}$$

This guarantees $\tilde{x}_1 \to 0$ asymptotically.
Based on 4.15 the update law for the estimated parameter $\hat{\theta}_1$ can be derived:

$$\dot{\hat{\theta}}_1 = -\dot{\tilde{\theta}}_1 = \Gamma_1\tilde{x}_1\phi_1 \tag{4.17}$$

$$\hat{\theta}_1 = \Gamma_1\int\tilde{x}_1\phi_1 \tag{4.18}$$

STEP 2

The next step is to pass α_1 to a command filter to obtain $x_{2,c}$ and $\dot{x}_{2,c}$.
The following command filter in the form of a second order system is considered:

$$\begin{cases}\dot{q}_{1,1} = \omega_n q_{1,2} \\ \dot{q}_{1,2} = -2\zeta\omega_n q_{1,2} - \omega_n(q_{1,1} - \alpha_1)\end{cases} \tag{4.19}$$

where $x_{2,c} = q_{1,1}$ and $\dot{x}_{2,c} = \omega_n q_{1,2}$ are the outputs the command filter.
The tracking error $\tilde{x}_2$ between x_2 and $x_{2,c}$ is defined as follows:

$$\tilde{x}_2 = x_2 - x_{2,c} \tag{4.20}$$

and its derivative $\dot{\tilde{x}}_2$:

$$\dot{\tilde{x}}_2 = \dot{x}_2 - \dot{x}_{2,c} \tag{4.21}$$

Because a command filter was used to filter α_1 a compensation tracking error v_1 of $\tilde{x}_1$ is derived, and it will be used instead of $\tilde{x}_1$ in the next calculations:

$$v_1 = \tilde{x}_1 - \xi_1 \tag{4.22}$$

The $\dot{\xi}_1$ signal is defined as:

$$\dot{\xi}_1 = -k_1\xi_1 + g_1(x_{2,c} - \alpha_1), \qquad \xi_1(0) = 0 \tag{4.23}$$

The derivative of v_1 will be now calculated as it will be used in the calculation of the next Lyapunov function:

$$\begin{aligned}\dot{v}_1 &= \dot{\tilde{x}}_1 - \dot{\xi}_1 \\ &= [f_1 + g_1x_2 + \phi_1^T\theta_1 + \psi_1(x) - \dot{y}_d] - [-k_1\xi_1 + g_1(x_{2,c} - \alpha_1)] \\ &= f_1 + g_1x_2 + \phi_1^T\theta_1 + \psi_1(x) - \dot{y}_d + k_1\xi_1 - g_1x_{2,c} + g_1\alpha_1\end{aligned}$$

by substituting $\tilde{x}_2 = x_2 - x_{2,c}$ and α_1 with equation 4.13:

$$\begin{aligned}\dot{v}_1 &= g_1\tilde{x}_2 + \phi_1^T\theta_1 + k_1\xi_1 - k_1\tilde{x}_1 - \phi_1^T\hat{\theta}_1 \\ &= -k_1(\tilde{x}_1 - \xi_1) + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1 \\ &= -k_1v_1 + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1\end{aligned} \tag{4.24}$$

The following Lyapunov Function $V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ will be chosen, where instead of $\tilde{x}_1$ its compensated signal v_1 is included

$$V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = \frac{1}{2}\tilde{x}_1^2 + \frac{1}{2}\tilde{x}_2^2 + \frac{1}{2}\tilde{\theta}_1^T\Gamma_1^{-1}\tilde{\theta}_1 + \frac{1}{2}\tilde{\theta}_2^T\Gamma_2^{-1}\tilde{\theta}_2 \tag{4.25}$$

Where Γ_1 and Γ_2 are diagonal positive definite matrices.
The derivative of $V_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ is considered:

$$\begin{aligned}&\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = v_1\dot{v}_1 + \tilde{x}_2\dot{\tilde{x}}_2 + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 \\ &= v_1(-k_1v_1 + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1) + \tilde{x}_2(f_2 + g_2x_3 + \phi_2^T\theta_2 + \psi_2(x) - \dot{x}_{2,c}) + \\ &+ \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 \\ &= -k_1v_1^2 + \tilde{x}_2(g_1v_1 + f_2 + g_2x_3 + \phi_2^T\theta_2 + \psi_2(x) - \dot{x}_{2,c}) + \\ &+ \tilde{\theta}_1^T(\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + v_1\phi) + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2\end{aligned} \tag{4.26}$$

The following values for the stabilizing function $x_3 = \alpha_2$ and $\dot{\tilde{\theta}}_1$ are chosen:

$$\alpha_2 = -\frac{1}{g_2}(f_2 + \phi_2^T\hat{\theta}_2 + \psi_2(x) - \dot{x}_{2c} - k_2\tilde{x}_2) \tag{4.27}$$

$$\dot{\tilde{\theta}}_1 = -\Gamma_1v_1\phi_1 \tag{4.28}$$

where $k_2 > 0$ and Γ_1 are tuning parameters.
By substituting $x_3 = \alpha_2$ and $\dot{\tilde{\theta}}_1$ into the Lyapunov Function $\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$:

$$\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = -k_1\tilde{x}_1^2 - k_2\tilde{x}_2^2 + \tilde{\theta}_2^T(\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{x}_2\phi) \tag{4.29}$$

Finally, the following value for $\dot{\tilde{\theta}}_2$ is chosen:

$$\dot{\tilde{\theta}}_1 = -\Gamma_2\tilde{x}_2\phi_2 \tag{4.30}$$

The derivative of the Lyapunov Function $\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2)$ becomes:

$$\dot{V}_2(v_1, \tilde{x}_2, \tilde{\theta}_1, \tilde{\theta}_2) = -k_1\tilde{x}_1^2 - k_2\tilde{x}_2^2 < 0 \tag{4.31}$$

This guarantees:

- $v_1 = \tilde{x}_1 - \xi_1 \to 0$ asymptotically
- $\tilde{x}_2 = x_2 - \dot{x}_{2,c} \to 0$ asymptotically

The new adaptive laws to estimate $\hat{\theta}_1$ and $\hat{\theta}_2$ can be derived:

$$\dot{\hat{\theta}}_1 = -\dot{\tilde{\theta}}_1 = \Gamma_1 v_1 \phi_1 \tag{4.32}$$

$$\hat{\theta}_1 = \Gamma_1 \int v_1 \phi_1 \tag{4.33}$$

$$\dot{\hat{\theta}}_2 = -\dot{\tilde{\theta}}_2 = \Gamma_2 \tilde{x}_2 \phi_2 \tag{4.34}$$

$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2 \phi_2 \tag{4.35}$$

Based on these intermediate results, it's already possible to determine a control for second order system nonlinear system.
The control can be achieve considering $\alpha_2 = u$.

STEP 3

The backstepping procedure will be iterated for the third and last time because the considered system is a third order system ($n = 3$). The second stabilizing function α_2 will be passed to a command filter to obtain $x_{3,c}$ and $\dot{x}_{3,c}$.
The following command filter will be considered:

$$\begin{cases} \dot{q}_{2,1} = \omega_n q_{2,2} \\ \dot{q}_{2,2} = -2\zeta\omega_n q_{2,2} - \omega_n (q_{2,1} - \alpha_2) \end{cases} \tag{4.36}$$

where $x_{3,c} = q_{2,1}$ and $\dot{x}_{3,c} = \omega_n q_{2,2}$ are the outputs of the command filter.
The tracking error $\tilde{x}_3$ between x_3 and $x_{3,c}$ is defined as follows:

$$\tilde{x}_3 = x_3 - x_{3,c} \tag{4.37}$$

and its derivative

$$\dot{\tilde{x}}_3 = \dot{x}_3 - \dot{x}_{3,c} \tag{4.38}$$

Because a command filter has been used to filter α_2, a compensation tracking error v_2 of $\tilde{x}_2$ is defined and it will be used instead of $\tilde{x}_2$:

$$v_2 = \tilde{x}_3 - \xi_2 \tag{4.39}$$

The $\dot{\xi}_2$ signal is defined as:

$$\dot{\xi}_2 = -k_2 \xi_2 + g_2 (x_{3,c} - \alpha_2), \qquad \xi_2(0) = 0 \tag{4.40}$$

The derivative of v_2 will be calculated as it will be necessary in the calculation of the derivative of the next Lyapunov function.

$$\begin{aligned} \dot{v}_2 &= \dot{\tilde{x}}_2 - \dot{\xi}_2 \\ &= [f_2 + g_2 x_3 + \phi_2^T \theta_2 + \psi_2(x) - \dot{x}_{2,c}] - [-k_2 \xi_2 + g_2 (x_{3,c} - \alpha_2)] \\ &= f_2 + g_2 x_3 + \phi_2^T \theta_2 + \psi_2(x) - \dot{x}_{2,c} + k_2 \xi_2 - g_2 x_{3,c} + g_2 \alpha_2 \end{aligned}$$

Substituting $\dot{\tilde{x}}_2 = \dot{x}_2 - \dot{x}_{2,c}$ and α_2 in equation (4.27) leads to:

$$\begin{aligned}\dot{v}_2 &= g_2\tilde{x}_3 + \phi_2^T\theta_2 + k_2\xi_2 - k_2\tilde{x}_3 - \phi_2^T\hat{\theta}_2 \\ &= -k_2(\tilde{x}_3 - \xi_2) + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2 \\ &= -k_2v_2 + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2 \end{aligned} \tag{4.41}$$

The third Lyapunov Function $V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ is defined as follows:

$$V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = \frac{1}{2}\tilde{v}_1^2 + \frac{1}{2}\tilde{v}_2^2 + \frac{1}{2}\tilde{x}_3^2 + \frac{1}{2}\tilde{\theta}_1^T\Gamma_1^{-1}\tilde{\theta}_1 + \frac{1}{2}\tilde{\theta}_2^T\Gamma_2^{-1}\tilde{\theta}_2 + \frac{1}{2}\tilde{\theta}_3^T\Gamma_3^{-1}\tilde{\theta}_3 \tag{4.42}$$

Where Γ_1, Γ_2 and Γ_3 are diagonal positive definite matrices.
The derivative of $V_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ will be considered in order to guarantee $v_1, v_2, \tilde{x}_3 = 0$ asymptotically.

$$\begin{aligned}&\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = v_1\dot{v}_1 + v_2\dot{v}_2 + \tilde{x}_3\dot{\tilde{x}}_3 + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \\ &= v_1(-k_1v_1 + g_1\tilde{x}_2 + \phi_1^T\tilde{\theta}_1) + v_2(-k_2v_1 + g_2\tilde{x}_3 + \phi_2^T\tilde{\theta}_2) + \\ &+ \tilde{x}_3(f_2 + g_3u + \phi_3^T\theta_3 + \psi_3(x) - \dot{x}_{3,c}) + \tilde{\theta}_1^T\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + \tilde{\theta}_2^T\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \\ &= -k_1v_1^2 + -k_2v_1^2 + \tilde{x}_3(g_2v_1 + f_3 + g_3u + \phi_3^T\theta_3 + \psi_3(x) - \dot{x}_{3,c}) + \\ &+ \tilde{\theta}_1^T(\Gamma_1^{-1}\dot{\tilde{\theta}}_1 + v_1\phi_1) + \tilde{\theta}_2^T(\Gamma_2^{-1}\dot{\tilde{\theta}}_2 + v_2\phi_2) + \tilde{\theta}_3^T\Gamma_3^{-1}\dot{\tilde{\theta}}_3 \end{aligned} \tag{4.43}$$

One chooses the following values for the real input signal u, $\dot{\tilde{\theta}}_1$ and $\dot{\tilde{\theta}}_2$:

$$u = -\frac{1}{g_3}(f_3 + \phi_3^T\hat{\theta}_3 + \psi_3(x) - \dot{x}_{3,c} - k_3\tilde{x}_3) \tag{4.44}$$

$$\dot{\tilde{\theta}}_1 = -\Gamma_1v_1\phi_1 \tag{4.45}$$

$$\dot{\tilde{\theta}}_2 = -\Gamma_2v_2\phi_2 \tag{4.46}$$

where $k_3 > 0$, Γ_1 and Γ_2 are tuning parameters.
By substituting the defined values for u, $\dot{\tilde{\theta}}_1$ and $\dot{\tilde{\theta}}_2$ in $\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ one obtains:

$$\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = -k_1v_1^2 - k_2v_2^2 - k_3\tilde{x}_3 + \tilde{\theta}_3^T(\Gamma_3^{-1}\dot{\tilde{\theta}}_3 + \tilde{x}_3\phi_3) \tag{4.47}$$

Finally, the following value for $\dot{\tilde{\theta}}_3$ is chosen:

$$\dot{\tilde{\theta}}_3 = -\Gamma_3\tilde{x}_3\phi_3 \tag{4.48}$$

The derivative of the Lyapunov Function $\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3)$ becomes:

$$\dot{V}_3(v_1, v_2, \tilde{x}_3, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3) = -k_1\tilde{v}_1^2 - k_2\tilde{v}_2^2 - k_3\tilde{x}_3^2 < 0 \tag{4.49}$$

This guarantees:

- $v_1 = \tilde{x}_1 - \xi_1 \to 0$ asymptotically
- $v_2 = x_2 - \xi_2 \to 0$ asymptotically
- $\tilde{x}_3 = x_3 - \dot{x}_{3,c} \to 0$ asymptotically

The new Adaptive Laws to estimate $\hat{\theta}_2$ and $\hat{\theta}_3$ can be derived:

$$\dot{\hat{\theta}}_2 = -\dot{\tilde{\theta}}_2 = \Gamma_1 v_2 \phi_2 \tag{4.50}$$

$$\hat{\theta}_2 = \Gamma_2 \int v_2 \phi_2 \tag{4.51}$$

$$\dot{\hat{\theta}}_3 = -\dot{\tilde{\theta}}_3 = \Gamma_3 \tilde{x}_3 \phi_3 \tag{4.52}$$

$$\hat{\theta}_3 = \Gamma_3 \int \tilde{x}_3 \phi_3 \tag{4.53}$$

The important results will be summarized:
Stabilizing functions α_1,α_1 and the real control input u:

$$\alpha_1 = -\frac{1}{g_1}(f_1 + \phi_1^T \hat{\theta}_1 + \psi_1(x) - y_d - k_1 \tilde{x}_1)$$
$$\alpha_2 = -\frac{1}{g_2}(f_2 + \phi_2^T \hat{\theta}_2 + \psi_2(x) - \dot{x}_{2,c} - k_2 \tilde{x}_2)$$
$$u = -\frac{1}{g_3}(f_3 + \phi_3^T \hat{\theta}_3 + \psi_3(x) - \dot{x}_{3,c} - k_3 \tilde{x}_3)$$

Adaptive laws for $\hat{\theta}_1, \hat{\theta}_2$ and $\hat{\theta}_3$

$$\hat{\theta}_1 = \Gamma_2 \int v_1 \phi_1$$
$$\hat{\theta}_2 = \Gamma_2 \int v_2 \phi_2$$
$$\hat{\theta}_3 = \Gamma_3 \int \tilde{x}_3 \phi_3$$

The adaptive laws for $\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3$ can be expressed by using the non-compensated tracking errors $\tilde{x}_i$:

$$\hat{\theta}_1 = \Gamma_2 \int \tilde{x}_1 \phi_1$$
$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2 \phi_2$$
$$\hat{\theta}_3 = \Gamma_3 \int \tilde{x}_3 \phi_3$$

This alternative formulation is less accurate than the previous one, but it will simplify the modeling in the simulations.

Chapter 5

Simulations

In this chapter, simulations of the developed adaptive backstepping control systems will be shown.

5.1 Command Filtered Adaptive Backstepping Control

Second order system control case

The following second order nonlinear system will be considered:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + \theta_1 x_1^2 + (1 + 0.1x_1^2)x_2 \\ \dot{x}_2 = x_1 x_2 + \theta_2[\sqrt{x} + 0.8sin(x_1)] + (2 + cos(x_1))u \\ y = x_1 \end{cases} \tag{5.1}$$

where x_1,x_2 are the state variables, y is the output of the system and the parameters θ_1 and θ_2 are unknown. The initial condition is $x_0 = [x_{10}, x_{20}]^T = [0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$.
The stabilizing function α_1 and the real control input u are defined as following:

$$\alpha_1 = -\frac{1}{1 + 0.1x_1^2}(0.5x_1 + x_1^2\hat{\theta}_1 - y_d - k_1\tilde{x}_1)$$
$$u = -\frac{1}{2 + cos(x_1)}(x_1 x_2 + [\sqrt{x} + 0.8sin(x_1)]\hat{\theta}_2 - \dot{x}_{2,c} - k_2\tilde{x}_2)$$

where k_1 and k_2 are tuning parameters.
The unknown parameters θ_1 and θ_2 will be estimated by using the following adaptive laws for $\hat{\theta}_1$ and $\hat{\theta}_2$:

$$\hat{\theta}_1 = \Gamma_1 \int \tilde{x}_1 x_1^2$$
$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2[\sqrt{x} + 0.8sin(x_1)]$$

where Γ_1 and Γ_2 are tuning parameters.

The Command Filter parameters were chosen as $\zeta_1 = 1$ and $\omega_n = 300$.

Constant unknown parameters

First, the case where the unknown parameters θ_1 and θ_2 are constant will be considered. The following plot shows a comparison between the output of the system and the reference signal.

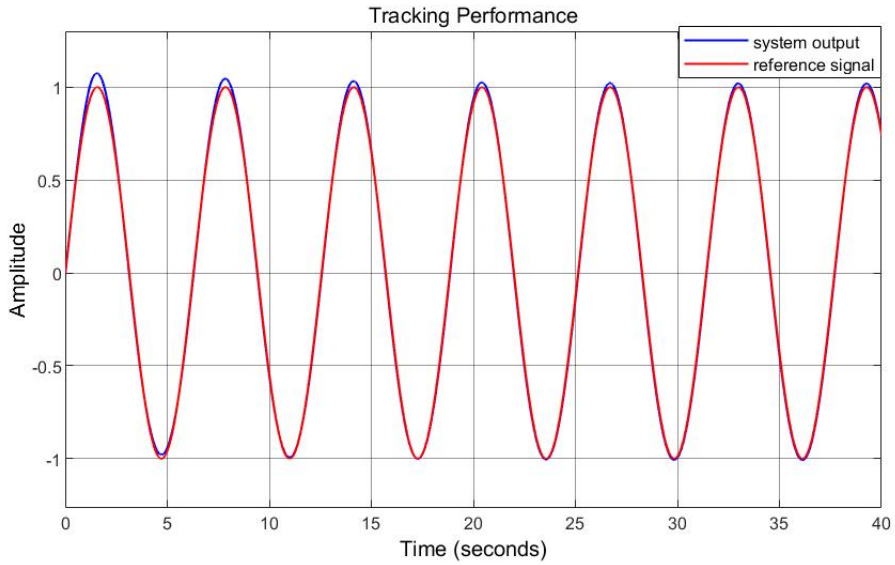

Figure 5.1: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Constant unknown parameters original values: θ_1=2 and θ_2=3.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10.

The following plot shows the error between the output of the system and the reference signal

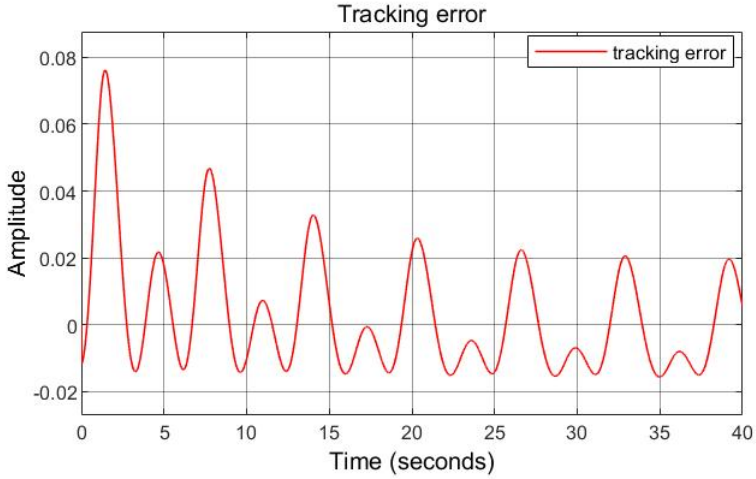

Figure 5.2: Error between system output and reference signal

Variable unknown parameters

In this section, the case where the unknown parameters change during the execution will be considered in order to test the robustness of the implemented adaptive control laws.
Tracking performance plot:

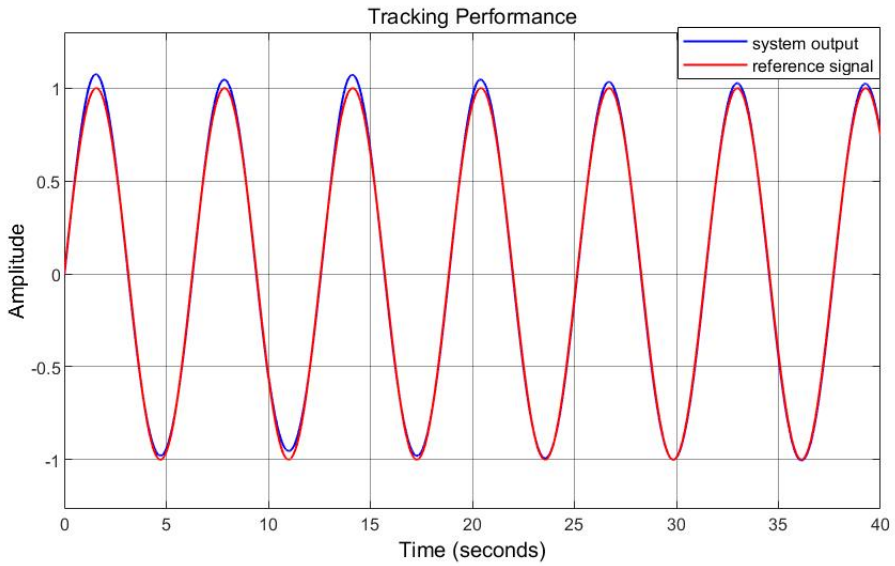

Figure 5.3: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Unknown parameters original values: θ_1=3 and θ_2=2, from $t = 10s$: θ_1=5 and from $t = 20s$: θ_2=5.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10.

Tracking error plot:

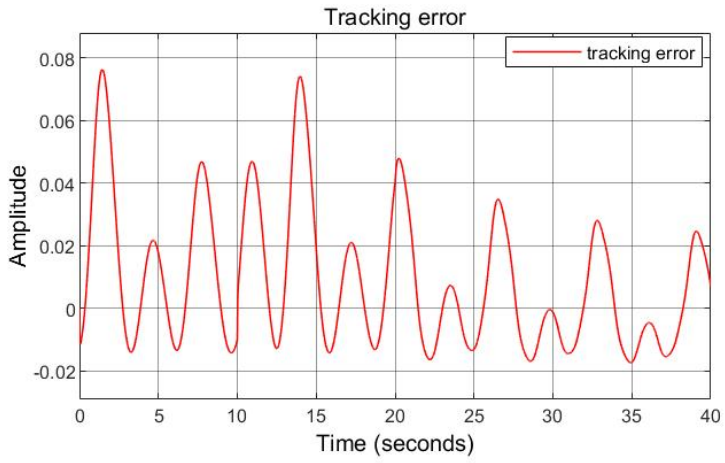

Figure 5.4: Error between system output and reference signal

In the previous section a deterioration in the tracking performance was shown as the value of the unknown parameters suddenly changed during the execution. To overcome the problem, higher values of the adaptive gains Γ_1 and Γ_2 were chosen.
Tracking performance plot:

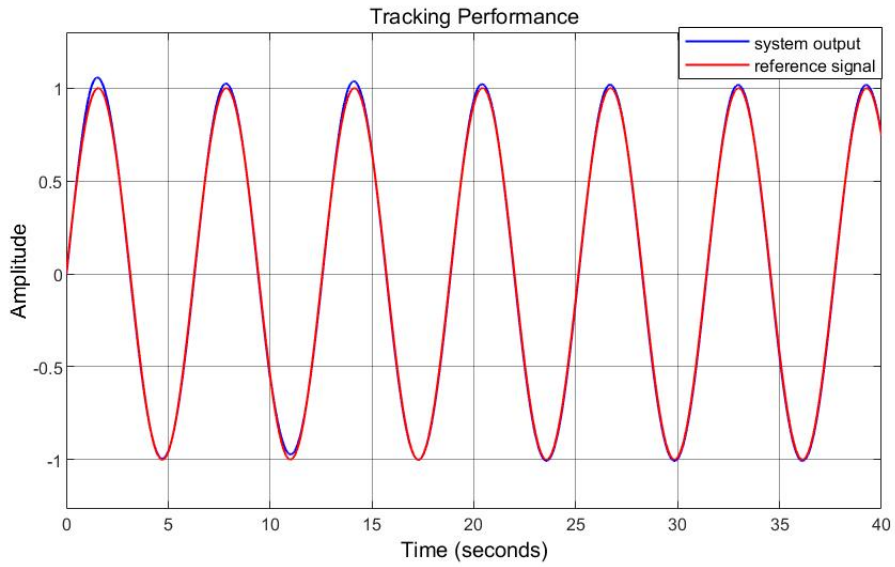

Figure 5.5: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Unknown parameters original values: θ_1=3 and θ_2=2,
  from $t = 10s$: θ_1=5 and from $t = 20s$: θ_2=5.
- Adaptive Control tuning parameters: Γ_1=25 and Γ_2=25.

Tracking error plot:

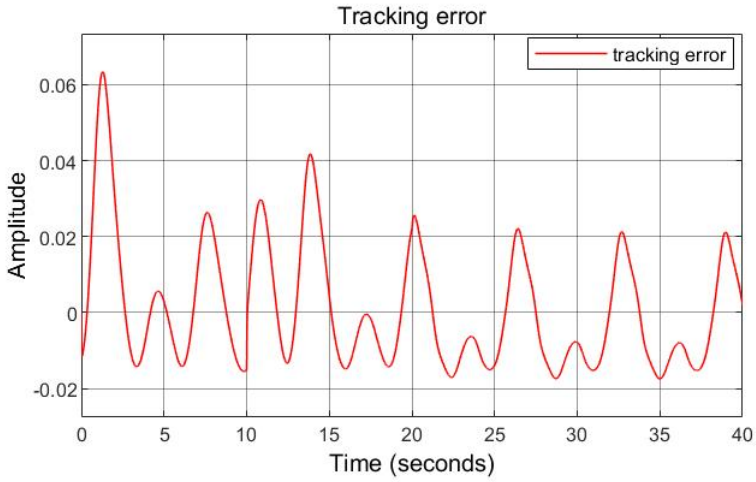

Figure 5.6: Error between system output and reference signal

Third order system control case

The following third order nonlinear system will be considered:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + \theta_1 x_1^2 + (1 + 0.1x_1^2)x_2 \\ \dot{x}_2 = x_1 x_2 + \theta_2[\sqrt{x} + 0.8sin(x_1)] + (2 + cos(x_1))x_3 \\ \dot{x}_3 = \theta_3(1 + x_3^2) + (2 + 0.5x_3^2)u \\ y = x_1 \end{cases} \tag{5.2}$$

where x_1,x_2 and x_3 are the state variables, y is the output of the system and the parameters θ_1, θ_2 and θ_3 are unknown. The initial condition is $x_0 = [x_{10}, x_{20}, x_{30}]^T = [0, 0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$.
The stabilizing functions α_1, α_2 and the real control input u are defined as following:

$$\alpha_1 = -\frac{1}{1 + 0.1x_1^2}(0.5x_1 + x_1^2\hat{\theta}_1 - y_d - k_1\tilde{x}_1)$$
$$\alpha_2 = -\frac{1}{2 + cos(x_1)}(x_1 x_2 + [\sqrt{x} + 0.8sin(x_1)]\hat{\theta}_2 - \dot{x}_{2,c} - k_2\tilde{x}_2)$$
$$u = -\frac{1}{2 + 0.5x_3^2}((1 + x_3^2)\hat{\theta}_3 - \dot{x}_{3,c} - k_3\tilde{x}_3)$$

where k_1, k_2 and k_3 are tuning parameters.
The parameters θ_1, θ_2 and θ_3 will be estimated by using the following adaptive laws for $\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3$

$$\hat{\theta}_1 = \Gamma_1 \int \tilde{x}_1 x_1^2$$
$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2[\sqrt{x} + 0.8sin(x_1)]$$
$$\hat{\theta}_2 = \Gamma_3 \int \tilde{x}_3(1 + x_3^2)$$

where Γ_1 and Γ_2 and Γ_3 are tuning parameters.
The parameters of the two implemented Command Filters were chosen as:
$\zeta_1, \zeta_2 = 1$ and $\omega_{n1}, \omega_{n2} = 300$.

Constant unknown parameters

First, the case where the unknown parameters θ_1 and θ_2 and θ_3 are constant will be considered. The following plot shows a comparison between the output of the system and the reference signal.

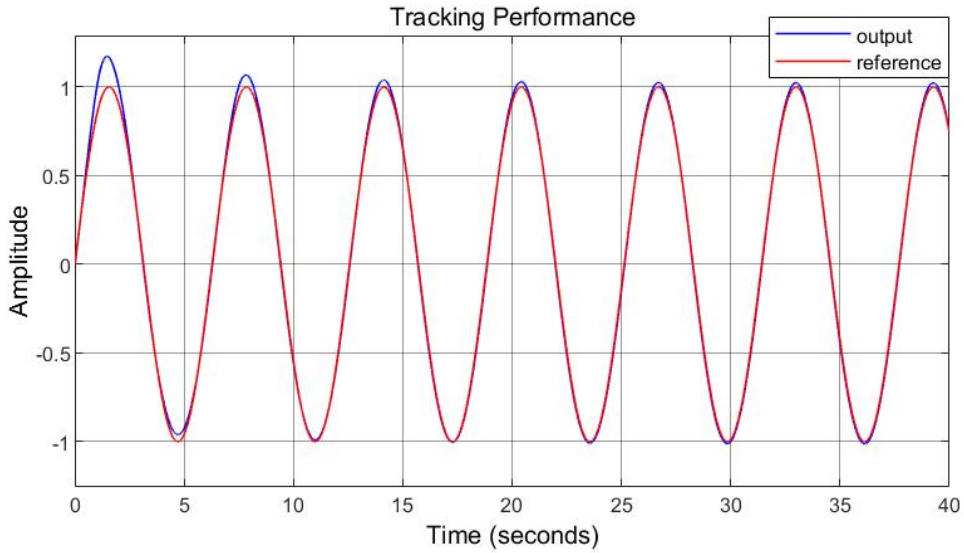

Figure 5.7: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Constant unknown parameters original values: θ_1=5, θ_2=3 and θ_3=4.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10 and Γ_3=10.

The following plot shows the error between the output of the system and the reference signal

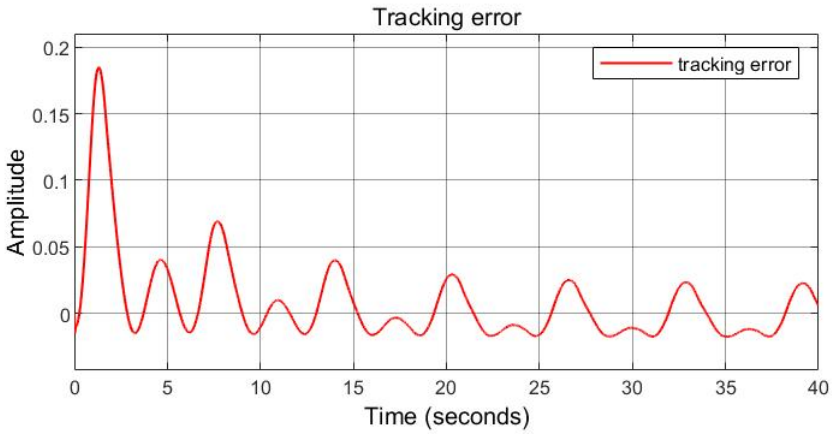

Figure 5.8: Error between system output and reference signal

Variable unknown parameters

In this section, the case where the unknown parameters change during the execution will be considered in order to test the robustness of the implemented adaptive control laws.
Tracking performance plot:

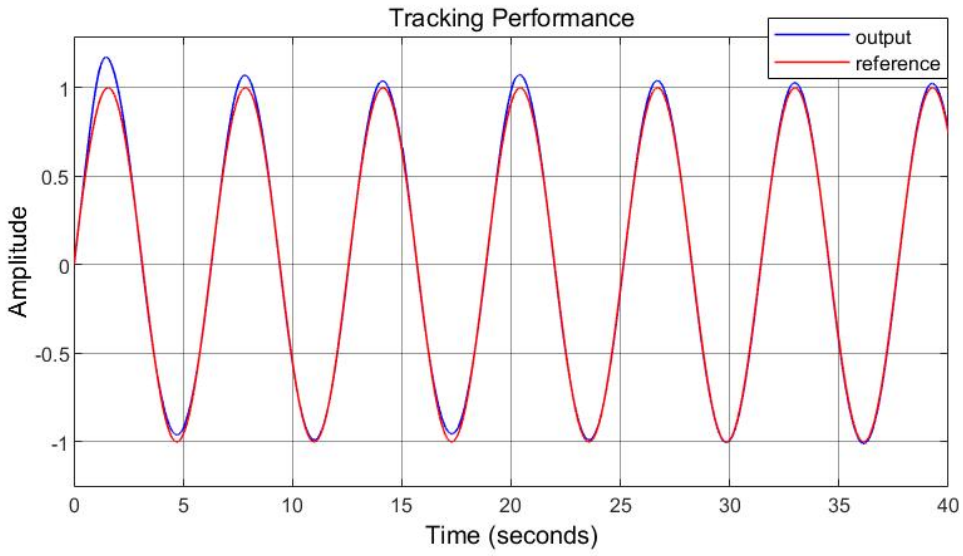

Figure 5.9: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Unknown parameters original values: θ_1=5 and θ_2=5 and θ_3=4 from $t = 5s$: θ_2=8 , from $t = 15s$: θ_1=7 and from $t = 20s$: θ_3=8.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10 and Γ_3=10.

Tracking error plot:

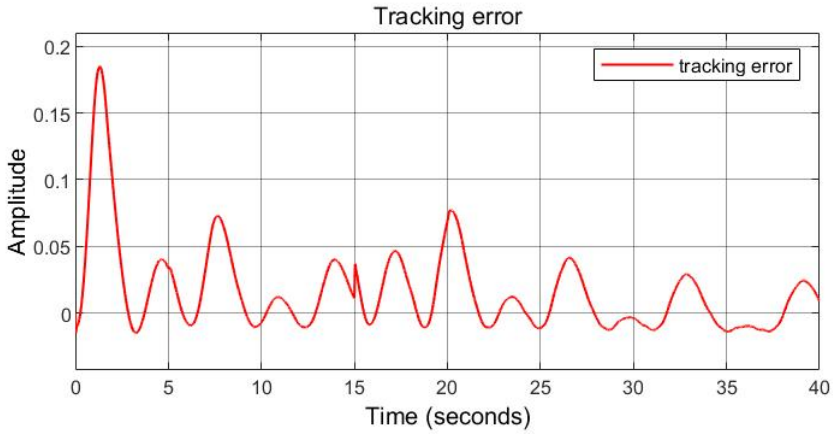

Figure 5.10: Error between system output and reference signal

In the previous section, a deterioration of the performance was shown, as the value of the parameter suddenly changed during the execution. To improve the performance, higher values of the adaptive gains Γ_1 and Γ_2 were chosen.
Tracking performance plot:

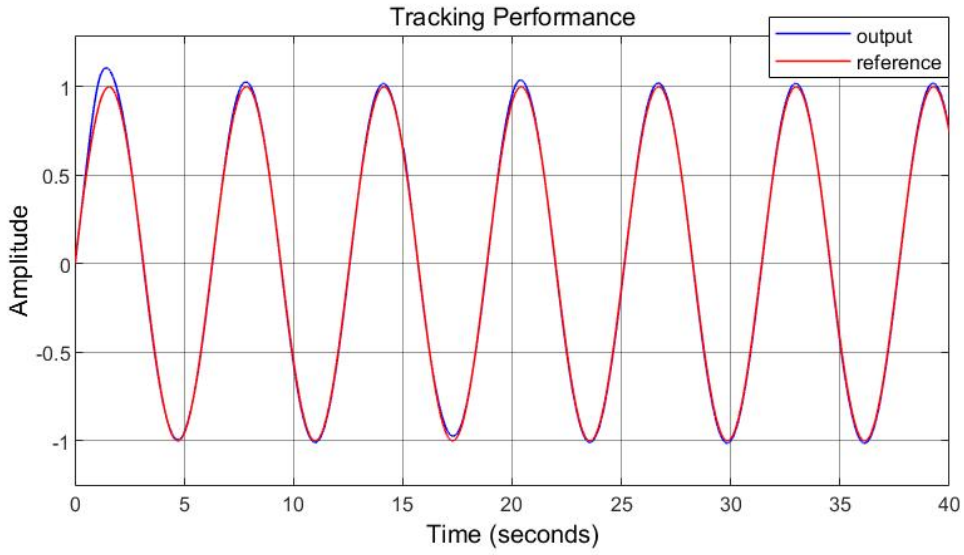

Figure 5.11: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Unknown parameters original values: θ_1=5 and θ_2=5 and θ_3=4 from $t = 5s$: θ_2=8 , from $t = 15s$: θ_1=7 and from $t = 20s$: θ_3=8.
- Adaptive Control tuning parameters: Γ_1=30 and Γ_2=20 and Γ_3=20.

Tracking error plot:

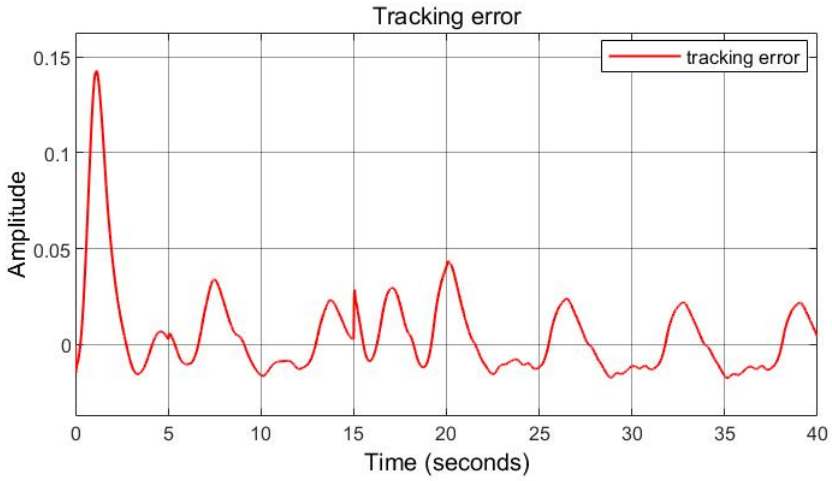

Figure 5.12: Error between system output and reference signal

5.2 Command Filtered Adaptive Backstepping Control with SVR

Second order system control case

The following second order nonlinear system will be considered:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + \theta_1 x_1^2 + (1 + 0.1x_1^2)x_2 + \Delta_1 \\ \dot{x}_2 = x_1 x_2 + \theta_2[\sqrt{x} + 0.8sin(x_1)] + (2 + cos(x_1))u + \Delta_2 \\ y = x_1 \end{cases} \tag{5.3}$$

where x_1,x_2 are the state variables, y is the output of the system and the parameters θ_1 and θ_2 are unknown. The initial condition is $x_0 = [x_{10}, x_{20}]^T = [0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$.
The stabilizing function α_1 and the real control input u are defined as following:

$$\alpha_1 = -\frac{1}{1 + 0.1x_1^2}(0.5x_1 + x_1^2\hat{\theta}_1 + \psi_1 - y_d - k_1\tilde{x}_1)$$
$$u = -\frac{1}{2 + cos(x_1)}(x_1 x_2 + [\sqrt{x} + 0.8sin(x_1)]\hat{\theta}_2 + \psi_2 - \dot{x}_{2,c} - k_2\tilde{x}_2)$$

where k_1 and k_2 are tuning parameters.
The unknown parameters θ_1 and θ_2 will be estimated by using the following adaptive laws for $\hat{\theta}_1$ and $\hat{\theta}_2$:

$$\hat{\theta}_1 = \Gamma_1 \int \tilde{x}_1 x_1^2$$
$$\hat{\theta}_2 = \Gamma_2 \int \tilde{x}_2[\sqrt{x} + 0.8sin(x_1)]$$

where Γ_1 and Γ_2 are tuning parameters.

The Command Filter parameters were chosen as $\zeta_1 = 1$ and $\omega_n = 300$.
The unknown disturbances Δ_1 and Δ_2 will be estimated by the following SVRs models:

$$\psi_1(x_{t,1}) = \sum_{k=1}^{N} \alpha_{k,1} e^{\left(-\gamma_1 \left\| x_{t,1} - x_{k,1} \right\|^2\right)} + b_1 \tag{5.4}$$

$$\psi_2(x_{t,2}) = \sum_{k=1}^{N} \alpha_{k,2} e^{\left(-\gamma_2 \left\| x_{t,2} - x_{k,2} \right\|^2\right)} + b_2 \tag{5.5}$$

where:

- $\alpha_{k,1}$ for $i = 1, \ldots, N$, $\alpha_{k,2}$ for $i = 1, \ldots, N$, and b_1 and b_2 are the solution of the optimization problem discussed in the previous chapter. The number of samples N depends on the accumulation t_{acc} and the sampling time.
- $\gamma_1, \gamma_2 > 0$ are tuning parameters.
- The samples $x_{k,1}$ and $x_{k,2}$ are the training data and $x_{t,1}$ and $x_{t,2}$ are the testing data, and both are acquired during the accumulation time t_{acc}.

LS-SVR parameter tuning

The value of the regularization parameter C was initially set to $C = 1000$. The impact of choosing a different values of C was not enough important on the function approximation and so it has been fixed to its initial chosen value $C = 1000$. On the other hand, in the case of the learning rate γ, different plots will show the big impact on the approximated functions when choosing different values.
The approximated functions ψ_1 and ψ_2 are generated by using a training data and validated with a testing data. In this work, the training data consists of pairs $\{x_i, y_i\}_1^N$ that have been accumulated for $t_{acc} = 40s$ using a sampling time $T_s = 0.05$, resulting in $N = 800$ samples. The unknown disturbances are considered to be only time dependent functions, so the input is the *sample time* and the output is the *amplitude*. For this reason, the plain data is in the form of $\{sample\ time_i, amplitude_i\}_{i=1}^N$. The training data $\{x_i\}_{i=1}^N$ was accumulated for $t_{acc} = 40s$ using a sampling time $T_s = 0.1$, resulting in $N = 400$ samples.
The effects of choosing different values for the learning rate will be shown in the case of the disturbance Δ_1. For the other disturbance Δ_2, only the plot that shows the estimated function with the properly tuned values will be shown.
The following plot shows an example where the chosen learning rate parameter was too high ($\gamma_1 = 3$).

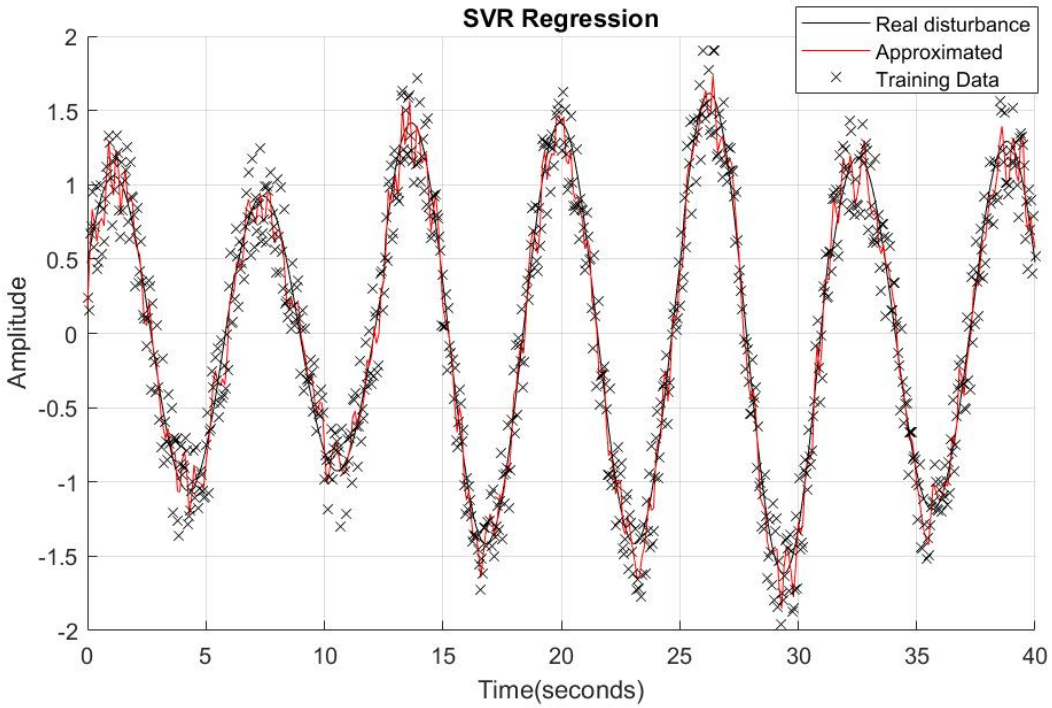

Figure 5.13: High value of the learning rate for ψ_1

The following plot shows an example where the chosen learning rate parameter was too low ($\gamma_1 = 0.01$).

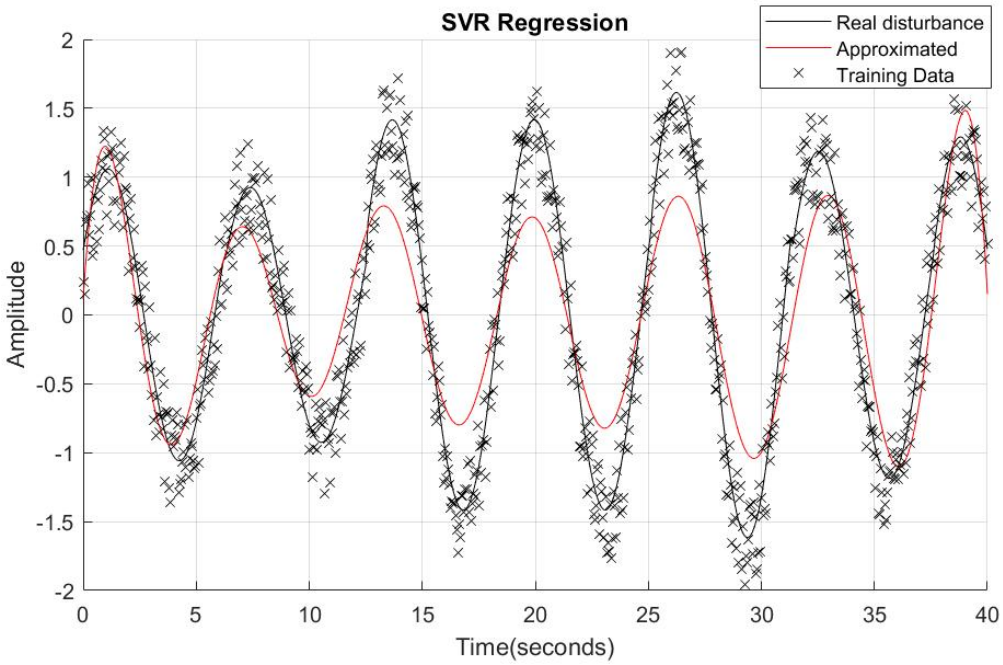

Figure 5.14: Low value of the learning rate for ψ_1

By choosing $\gamma_1 = 0.1$, the following plot shows that the estimated function ψ_1 fits the original disturbance Δ_1.

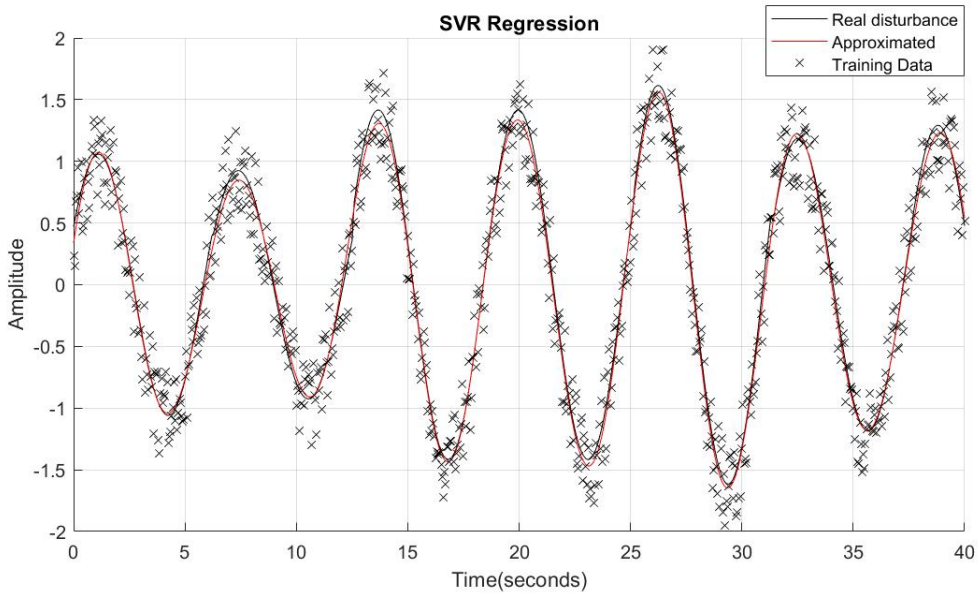

Figure 5.15: Correft value of the learning rate for ψ_1

By choosing $\gamma_2 = 0.06$, the following plot shows that the estimated function ψ_2 fits the original disturbance Δ_2.

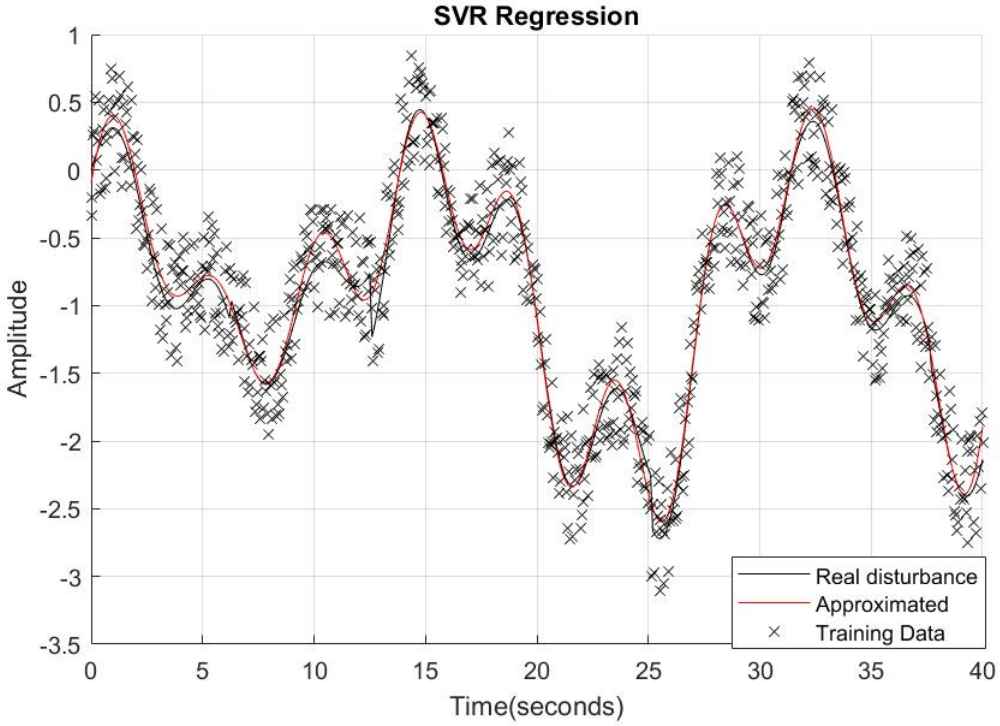

Figure 5.16: Correct value of the learning rate for ψ_2

The Support Vector Regression parameters chosen for the following simulations are as follows:

- $\gamma_1 = 0.1$
- $\gamma_2 = 0.06$
- C_1, C_2=1000

In the following simulations, the disturbances Δ_1 and Δ_2 will be added to the second order system previously considered in the simulations. It will be shown how the compensation of the disturbances in the control law is crucial in order to obtain a good tracking performance.
The previous simulations of the second order system in the "non disturbance case", showed that asymptotically the error stays in the range $e = |0.02|$. The objective of the next sections will be to achieve the same performance when the disturbances are present and then compensated in the system.

Constant unknown parameters

Without disturbance compensation

First, the case where the unknown parameters θ_1 and θ_2 are constant will be considered. The following plot shows a comparison between the output of the system and the reference signal.

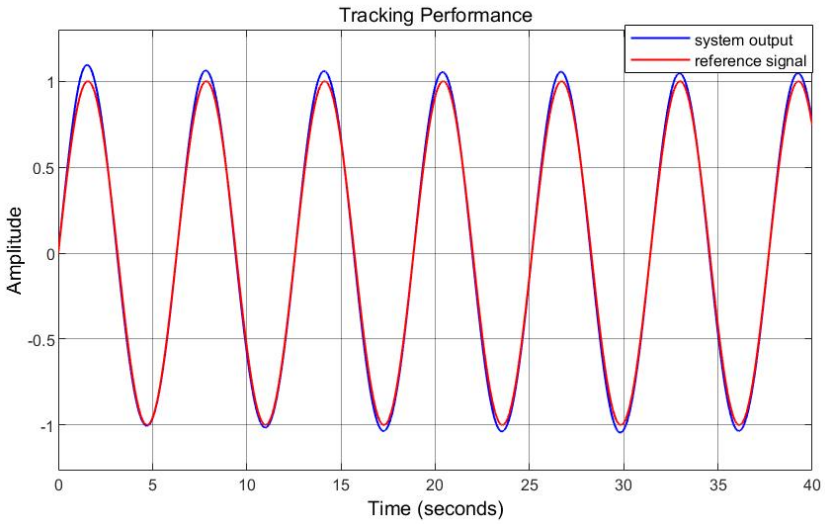

Figure 5.17: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Constant unknown parameters original values: θ_1=2 and θ_2=3.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10.

The following plot shows the error between the output of the system and the reference signal

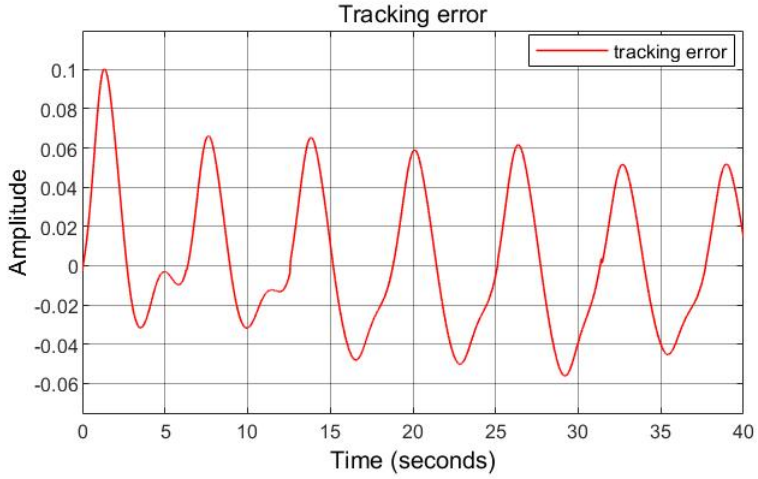

Figure 5.18: Error between system output and reference signal

With disturbance compensation

Now the LS-SVRs models ψ_1 and ψ_2 will be implemented in the control system to compensate the disturbances.
Tracking performance plot:

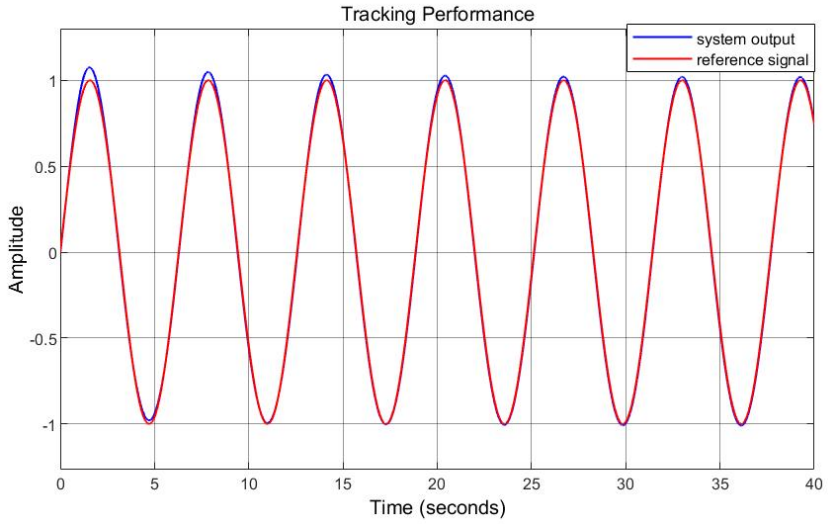

Figure 5.19: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Constant unknown parameters original values: θ_1=2 and θ_2=3.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10.
- LS-SVR parameters: $\gamma_1 = 0.1$,$\gamma_2 = 0.06$, $C_1 = 1000$ and $C_2 = 1000$.

The following plot shows the error between the output of the system and the reference signal

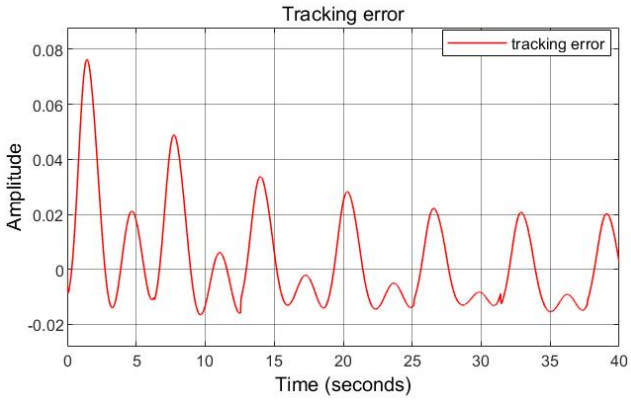

Figure 5.20: Error between system output and reference signal

Variable unknown parameters

Without disturbance compensation

In this section, the case where the unknown parameters change during the execution will be considered in order to test the robustness of the implemented adaptive control laws.
Tracking performance plot:

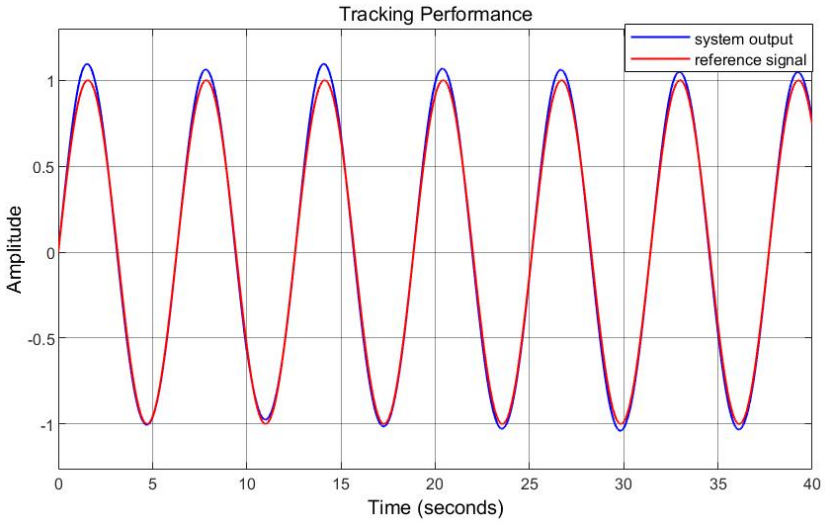

Figure 5.21: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$ and $k_2 = 30$.
- Unknown parameters original values: θ_1=3 and θ_2=2, from $t = 10s$: θ_1=5 and from $t = 20s$: θ_2=5.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10.

Tracking error plot:

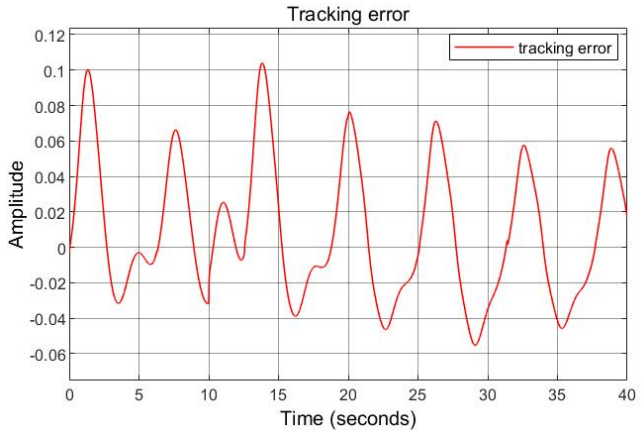

Figure 5.22: Error between system output and reference signal

To improve the performance, higher values of the adaptive gains can be chosen. However, the simulations will show that the tracking performance is deteriorated by the disturbances when compared to the case where the disturbances are not present in the system.
Tracking performance plot:

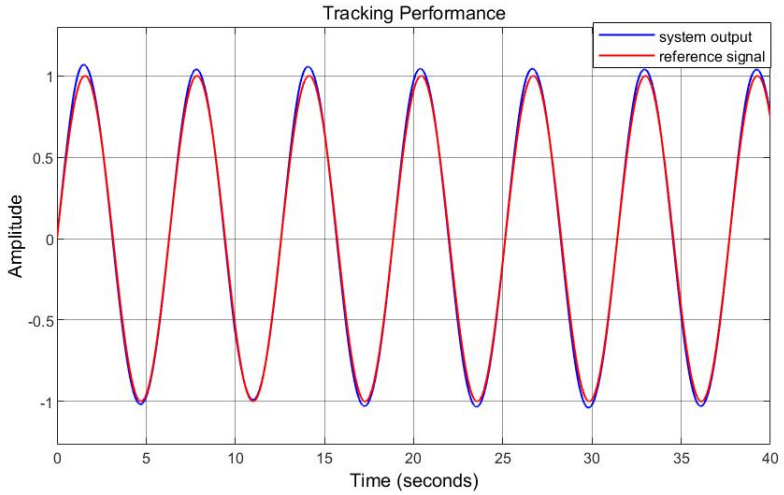

Figure 5.23: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gain $k_1 = 40$ and $k_2 = 40$.
- Unknown parameters original values: θ_1=3 and θ_2=2, from $t = 10s$: θ_1=5 and from $t = 20s$: θ_2=5.
- Adaptive Control tuning parameters Γ_1=25 and Γ_2=25.

Tracking error plot:

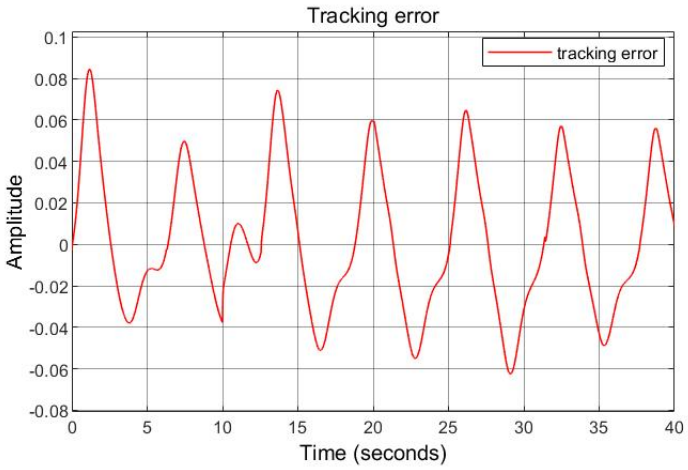

Figure 5.24: Error between system output and reference signal

With disturbance compensation

In this section, the compensation of the disturbances by Least Squares Support Vector Regression will be implemented. It will be shown how finally the performance can reach a satisfying level.
Tracking performance plot:

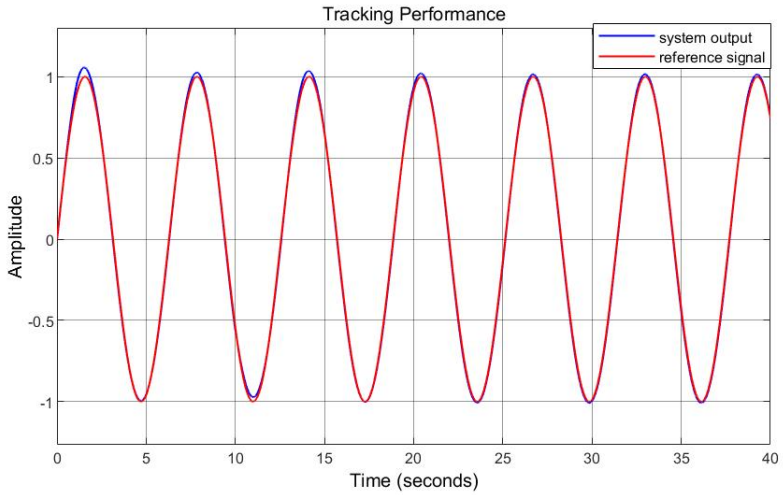

Figure 5.25: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 40$ and $k_2 = 40$.
- Unknown parameters original values: θ_1=3 and θ_2=2, from $t = 10s$: θ_1=5 and from $t = 20s$: θ_2=5.
- Adaptive Control tuning parameters Γ_1=25 and Γ_2=25.
- LS-SVR parameters: $\gamma_1 = 0.1$,$\gamma_2 = 0.06$, $C_1 = 1000$ and $C_2 = 1000$.

Tracking error plot :

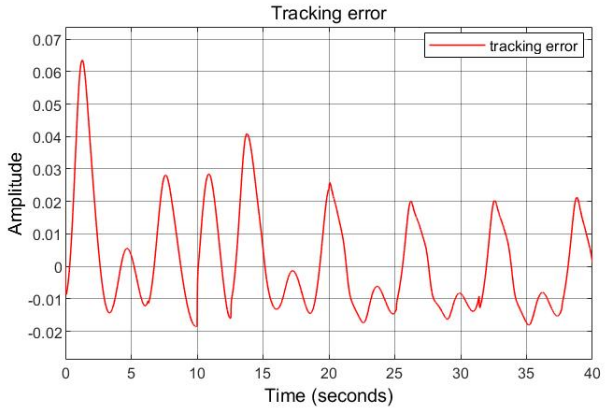

Figure 5.26: Error between system output and reference signal

Third order system control case

The following third order non system will be considered for the simulations:

$$\begin{cases} \dot{x}_1 = 0.5x_1 + \theta_1 x_1^2 + (1 + 0.1x_1^2)x_2 + \Delta_1 \\ \dot{x}_2 = x_1 x_2 + \theta_2[\sqrt{x} + 0.8sin(x_1)] + (2 + cos(x_1))x_3 + \Delta_2 \\ \dot{x}_3 = \theta_3(1 + x_3^2) + (2 + 0.5x_3^2)u + \Delta_3 \\ y = x_1 \end{cases} \tag{5.6}$$

where x_1,x_2 and x_3 are the state variables, y is the output of the system and the parameters θ_1, θ_2 and θ_3 are unknown. The initial condition is $x_0 = [x_{10}, x_{20}, x_{30}]^T = [0, 0, 0]^T$ and the desired reference signal of the system is $y_d = sin(t)$.
The stabilizing functions α_1, α_2 and the real control input u are chosen as follows:

$$\begin{aligned} \alpha_1 &= -\frac{1}{1 + 0.1x_1^2}(0.5x_1 + x_1^2\hat{\theta}_1 + \psi_1 - y_d - k_1\tilde{x}_1) \\ \alpha_2 &= -\frac{1}{2 + cos(x_1)}(x_1 x_2 + [\sqrt{x} + 0.8sin(x_1)]\hat{\theta}_2 + \psi_2 - \dot{x}_{2,c} - k_2\tilde{x}_2) \\ u &= -\frac{1}{2 + 0.5x_3^2}((1 + x_3^2)\hat{\theta}_3 + \psi_3 - \dot{x}_{3,c} - k_3\tilde{x}_3) \end{aligned}$$

where k_1, k_2 and k_3 are tuning parameters.
The parameters θ_1, θ_2 and θ_3 will be estimated by using the following adaptive laws for $\hat{\theta}_1, \hat{\theta}_2$ and $\hat{\theta}_3$

$$\begin{aligned} \hat{\theta}_1 &= \Gamma_1 \int \tilde{x}_1 x_1^2 \\ \hat{\theta}_2 &= \Gamma_2 \int \tilde{x}_2[\sqrt{x} + 0.8sin(x_1)] \\ \hat{\theta}_2 &= \Gamma_3 \int \tilde{x}_3(1 + x_3^2) \end{aligned}$$

where Γ_1, Γ_2 and Γ_3 are tuning parameters.
The parameters of the two implemented Command Filters were chosen as:
$\zeta_1, \zeta_2 = 1$ and $\omega_{n1}, \omega_{n2} = 300$.
The following function $\psi_1(x)$, $\psi_2(x)$ and $\psi_3(x)$ are used as estimations of the three unknown disturbances Δ_1, Δ_2 and Δ_3 respectively:

$$\psi_1(x_{t,1}) = \sum_{k=1}^{N} \alpha_{k,1} e^{\left(-\gamma_1 \left\| x_{t,1} - x_{k,1} \right\|^2\right)} + b_1 \tag{5.7}$$

$$\psi_2(x_{t,2}) = \sum_{k=1}^{N} \alpha_{k,2} e^{\left(-\gamma_2 \left\| x_{t,2} - x_{k,2} \right\|^2\right)} + b_2 \tag{5.8}$$

$$\psi_3(x_{t,3}) = \sum_{k=1}^{N} \alpha_{k,3} e^{\left(-\gamma_3 \left\| x_{t,3} - x_{k,3} \right\|^2\right)} + b_3 \tag{5.9}$$

where:

- $\alpha_{k,1}$ for $i = 1, \ldots, N$, $\alpha_{k,2}$ for $i = 1, \ldots, N$, $\alpha_{k,3}$ for $i = 1, \ldots, N$ and b_1, b_2 and b_3 are the solution of the optimization problem discussed in the previous chapter. The number of samples N depends on the accumulation t_{acc} and the sampling time.
- $\gamma_1, \gamma_2, \gamma_3 > 0$ are tuning parameters.
- The samples $x_{k,1}$,$x_{k,2}$ and $x_{k,3}$ are the training data and $x_{t,1}$,$x_{t,2}$ and $x_{t,3}$ are the testing data, and both are acquired during the accumulation time t_{acc}.

LS-SVR parameter tuning

The value of the regularization parameter C was initially set to $C = 1000$. The impact of choosing a different values of C was not enough important on the function approximation and so it has been fixed to its initial chosen value $C = 1000$. The approximated functions ψ_1, ψ_2 and ψ_3 are generated by using a training data and validated with a testing data. In this work, the training data consists of pairs $\{x_i, y_i\}_1^N$ that have been accumulated for $t_{acc} = 40s$ using a sampling time $T_s = 0.05$, resulting in $N = 800$ samples. The unknown disturbances are considered to be only time dependent functions, so the input is the *sample time* and the output is the *amplitude*. For this reason, the plain data is in the form of $\{sample\ time_i, amplitude_i\}_{i=1}^N$. The training data $\{x_i\}_{i=1}^N$ was accumulated for $t_{acc} = 40s$ using a sampling time $T_s = 0.1$, resulting in $N = 400$ samples. The effect on the estimated functions by SVR when choosing different values for the learning rates was shown in the second order system control case. For this reason, in this section only the plots showing a satisfying fit between the original disturbances and the estimated functions will be shown. By choosing $\gamma_1 = 0.1$, the following plot shows that the estimated function ψ_1 fits the original disturbance Δ_1.

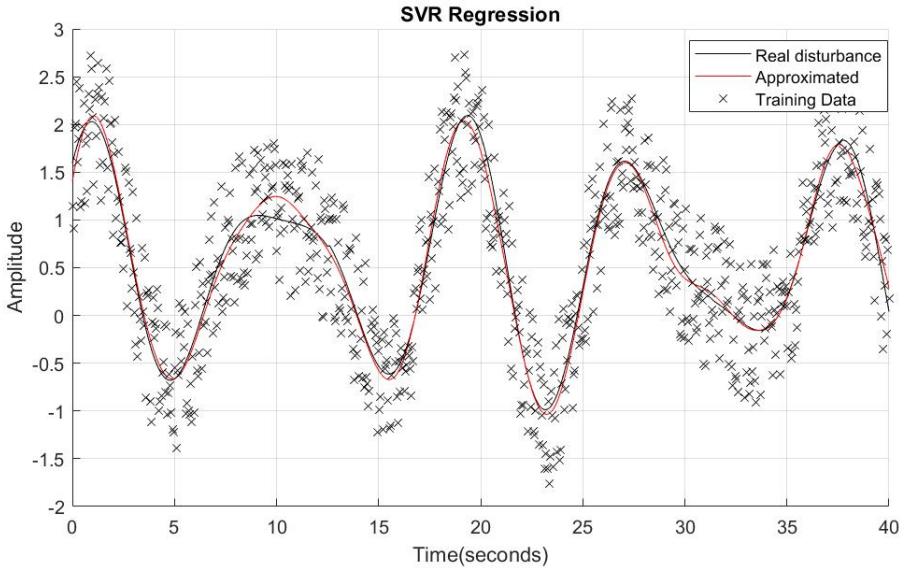

Figure 5.27: Correct value of the learning rate for ψ_1

By choosing $\gamma_2 = 0.8$, the following plot shows that the estimated function ψ_2 fits the original disturbance Δ_2.

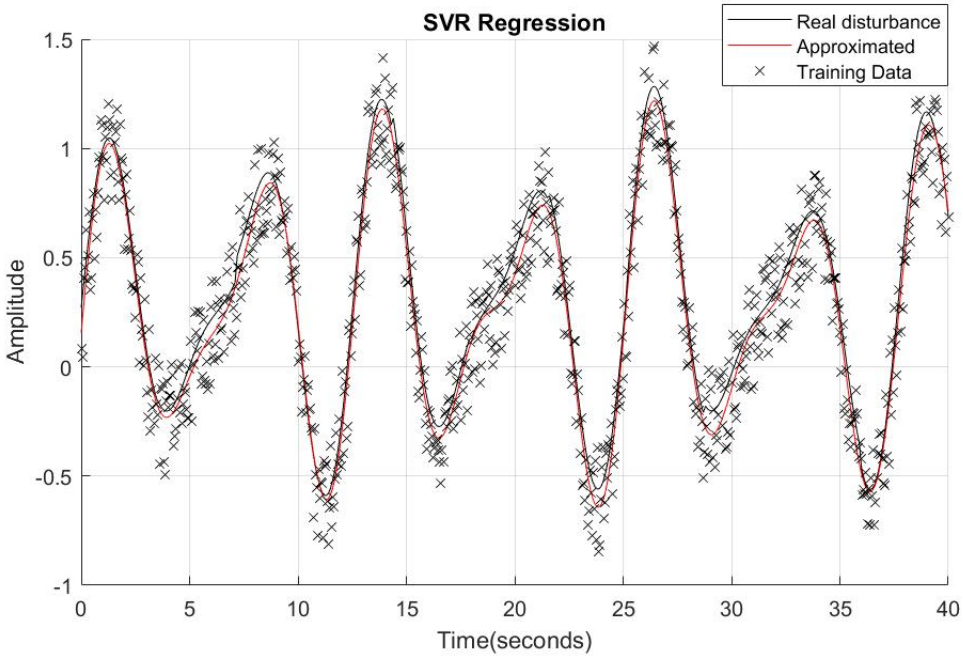

Figure 5.28: Correct value of the learning rate for ψ_2

By choosing $\gamma_3 = 0.14$, the following plot shows that the estimated function ψ_3 fits the original disturbance Δ_3.

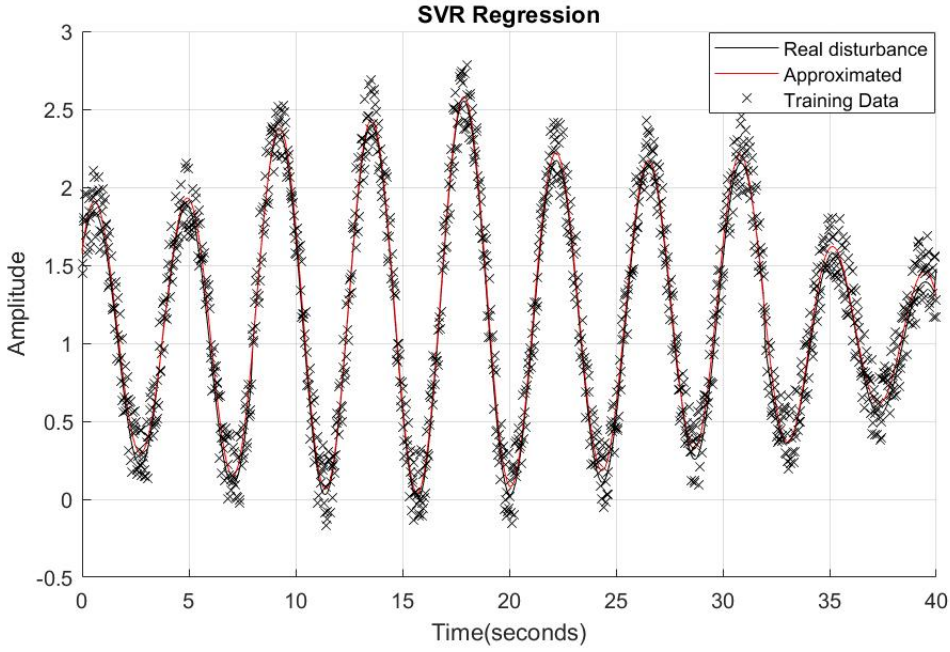

Figure 5.29: Correct value of the learning rate for ψ_3

Constant unknown parameters

Without disturbance compensation

First, the case where the unknown parameters θ_1, θ_2 and θ_3 are constant will be considered.
The following plot shows a comparison between the output of the system and the reference signal.

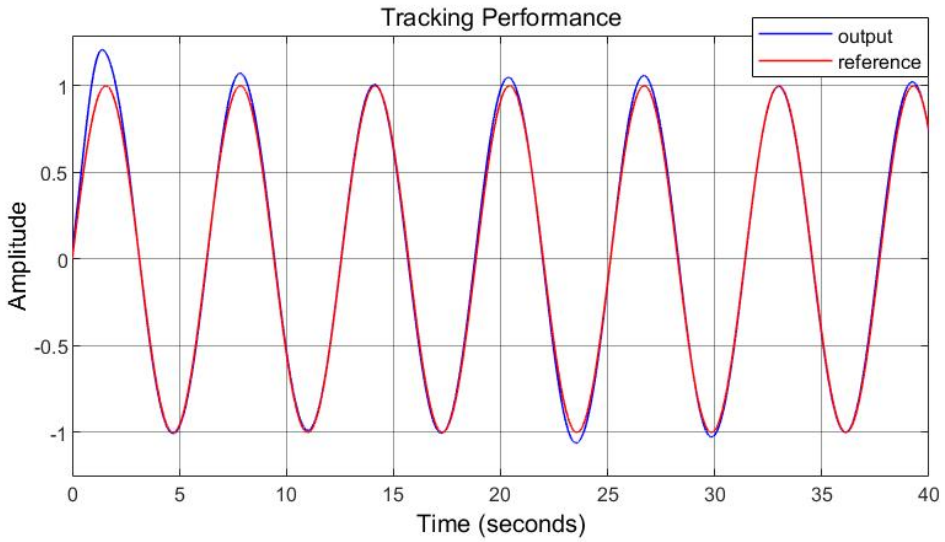

Figure 5.30: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Constant unknown parameters original values: θ_1=5, θ_2=3 and θ_2=4.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10 and Γ_3=10.

The following plot shows the error between the output of the system and the reference signal

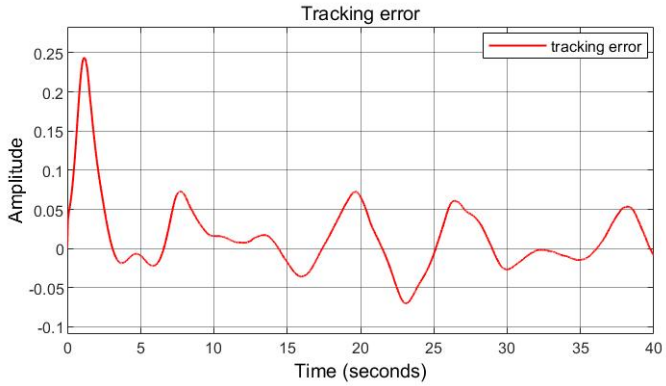

Figure 5.31: Error between system output and reference signal

With disturbance compensation

In this section, the LS-SVRs models ψ_1, ψ_2 and ψ_3 will be implemented in the control system to compensate the disturbances.
Tracking performance plot:

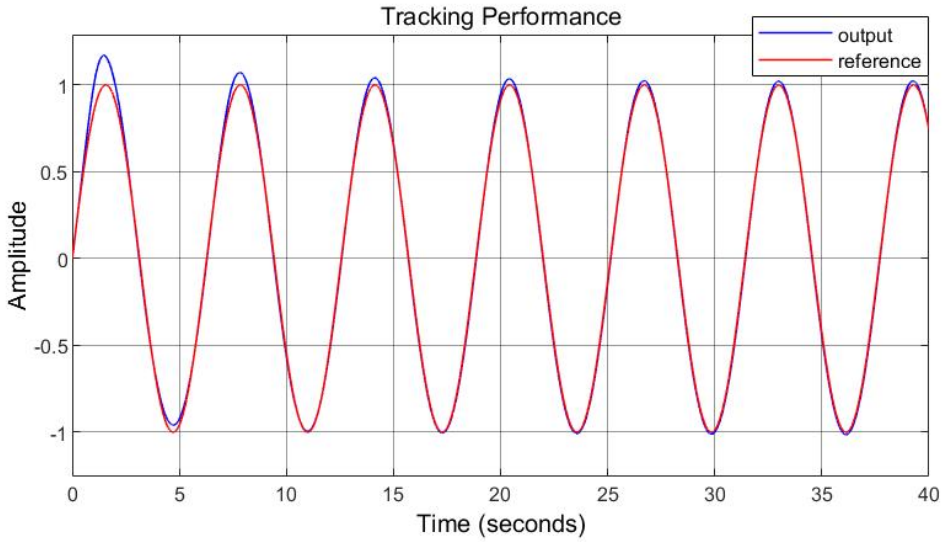

Figure 5.32: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Constant unknown parameters original values: θ_1=5, θ_2=3 and θ_2=4.
- Adaptive Control tuning parameters: Γ_1=10 and Γ_2=10 and Γ_3=10.
- LS-SVR parameters: $\gamma_1 = 0.1$, $\gamma_2 = 0.8$, $\gamma_3 = 0.14$ and $C_1 = 1000$, $C_2 = 1000$, $C_3 = 1000$.

The following plot shows the error between the output of the system and the reference signal

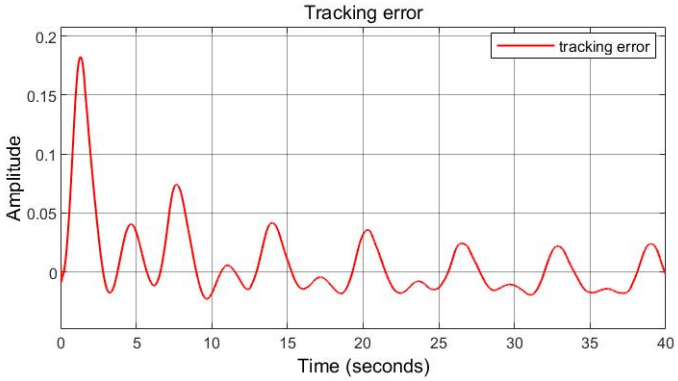

Figure 5.33: Error between system output and reference signal

Variable unknown parameters

Without disturbance compensation

In this section, the case where the unknown parameters change during the execution will be considered in order to test the robustness of the implemented adaptive control laws.
Tracking performance plot:

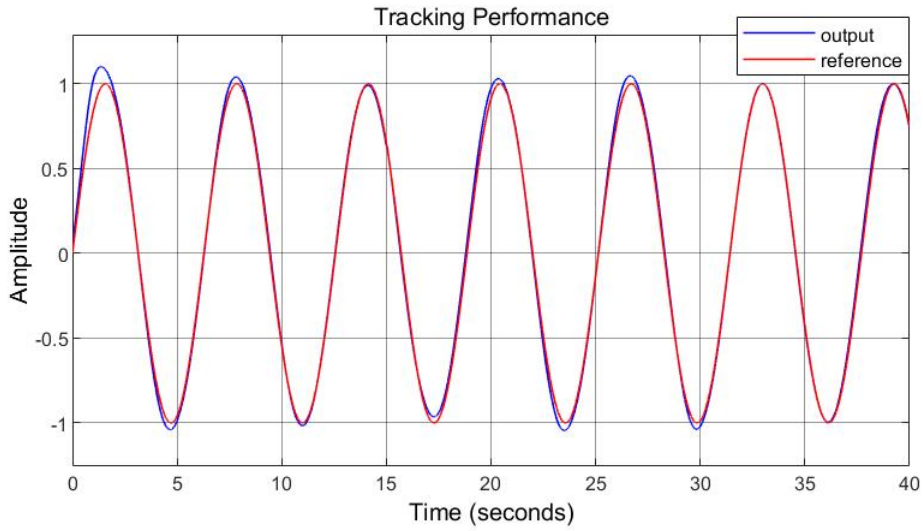

Figure 5.34: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Unknown parameters original values: θ_1=5 and θ_2=5 and θ_3=4 from $t = 5s$: θ_2=8 , from $t = 15s$: θ_1=7 and from $t = 20s$: θ_3=8
- Adaptive Control tuning parameters: Γ_1=30 and Γ_2=20 and Γ_3=20

Tracking error plot:

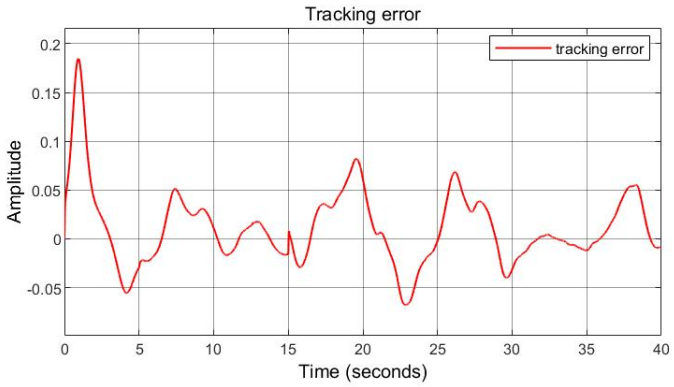

Figure 5.35: Error between system output and reference signal

With disturbance compensation

Finally, the compensation of the disturbances by Least Squares Support Vector Regression will be implemented to the case where the unknown parameters vary during the execution. It will be shown that the tracking performance is comparable to the non disturbance case.
Tracking performance plot:

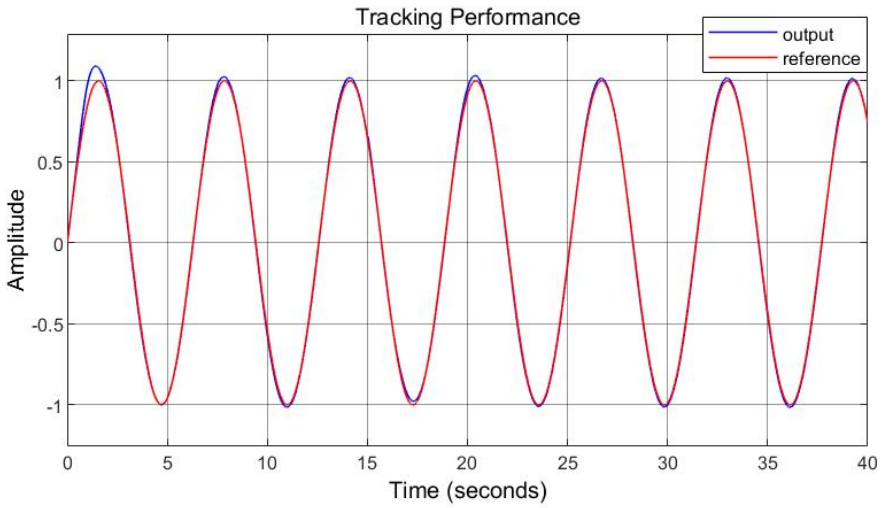

Figure 5.36: A comparison of the system output and reference signal

The tuning parameters were chosen as:

- Backstepping gains: $k_1 = 30$, $k_2 = 30$ and $k_3 = 20$.
- Unknown parameters original values: θ_1=5 and θ_2=5 and θ_3=4 from $t = 5s$: θ_2=8 , from $t = 15s$: θ_1=7 and from $t = 20s$: θ_3=8
- Adaptive Control tuning parameters: Γ_1=30 and Γ_2=20 and Γ_3=20
- LS-SVR parameters: $\gamma_1 = 0.1$, $\gamma_2 = 0.8$, $\gamma_3 = 0.14$ and $C_1 = 1000$, $C_2 = 1000$, $C_3 = 1000$.

Tracking error plot :

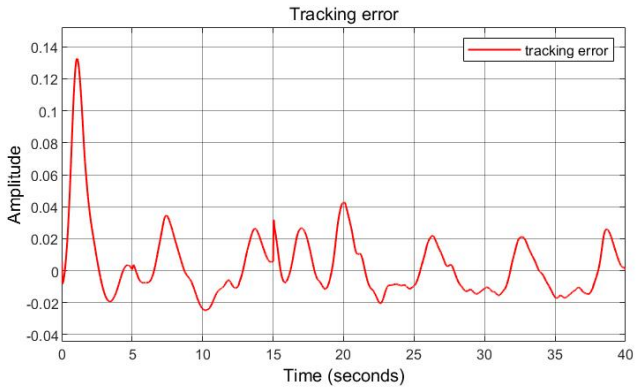

Figure 5.37: Error between system output and reference signal

Chapter 6

Conclusion and Future Work

A Command Filtered Adaptive Backstepping control for nonlinear systems with unknown parameters and an extension to the case where external disturbances are included in the controlled system have been developed in this thesis. In this work, the control systems have been designed for a second and third order nonlinear systems, but the control design can be extended to higher order systems by recursively applying the developed backstepping approach. The command filters avoided the high computational complexity required in the standard backstepping control when calculating the stabilizing functions. Moreover, an adaptive extension of the developed command filtered backstepping control has been designed in the case where the controlled system presented unknown parameters. In the simulations section, the results showed that the adaptive control laws could estimate the unknown parameters both in the case where these were unknown but constant and where they were varying during the execution. When no disturbances were considered in the controlled systems, the tracking error asymptotically had an absolute value of $e = |0.02|$, which has been used as an optimal target value for the tracking error in the case where the disturbances were included in the controlled systems. Later, a degradation in the performance was shown when the command filtered adaptive backstepping control developed in chapter 2 was used to control the system in the case where external disturbances were added to the system. To improve the tracking performance in the presence of disturbances, a novel Command Filtered Adaptive Backstepping control that included estimations of the original external disturbances was developed. The external disturbances have been estimated using a Support Vector Machine regression method called Least Squares Support Vector Regression. The Least Squares Support Vector Regression models were trained and tested using an available noisy set of data of the disturbances and then implemented to the control system to reject the original disturbances. The results showed that the novel controller was able to guarantee a tracking error value with asymptotic value $e = |0.02|$ as in the non disturbance case. The Least Squares Support Vector Regression tuning parameters had to be chosen by Trial and Error, so future works could potentially focus on finding a way to obtain the optimal parameters by using a more efficient method. Moreover, the LS-SVR was implemented to estimate only time dependent function, so for future works the developed control system could be tested in the case where the external disturbances are multivariate functions.

Bibliography

[1] J. Zhou and C. Wen, "Adaptive backstepping control," in *Adaptive Backstepping Control of Uncertain Systems* (J. Zhou and C. Wen, eds.), pp. 9–31, Springer, Berlin, Heidelberg, 2008.

[2] I. Landau, R. Lozano, M. M'Saad, and A. Karimi, *Adaptive Control: Algorithms, Analysis and Applications*. Springer, London, 2011.

[3] T. Ertuğrul, M. A. Adli, and M. U. Salamci, "Model reference adaptive control design for helicopters using gain scheduled reference models," *2016 17th International Carpathian Control Conference*, pp. 182–187, 2016.

[4] J. P. V. S. Cunha, L. Hsu, R. Costa, and F. Lizarralde, "Output-feedback model-reference sliding mode control of uncertain multivariable systems," *IEEE Transactions on Automatic Control* ***48***, pp. 2245–2250, 2003.

[5] S. Soleymanpour and M. Navabi, "Integrated adaptive backstepping attitude control of spacecraft," *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation*, pp. 949–954, 2017.

[6] V. K. Singh and V. Kumar, "Adaptive backstepping control design for stabilization of inverted pendulum," *2014 Students Conference on Engineering and Systems*, pp. 1–5, 2014.

[7] M. R. Junaid, L. M. Beebi, and C. R. Ashima, "Backstepping and adaptive backstepping control on robotic arm," *2015 International Conference on Control Communication and Computing*, pp. 1–6, 2015.

[8] L. Yinan, Z. Shengxiu, C. Lijia, and Z. Chao, "Adaptive backstepping control for nonlinear systems using support vector regression," in *Intelligence Computation and Evolutionary Computation* (Z. Du, ed.), pp. 13–23, Springer, Berlin, Heidelberg, 2013.

[9] W. Dong, J. A. Farrell, M. M. Polycarpou, V. Djapic, and M. Sharma, "Command filtered adaptive backstepping," *IEEE Transactions on Control Systems Technology* ***20***, pp. 566–580, 2012.

[10] S. Wang, M. Dai, and Y. Wang, "Robust adaptive backstepping sliding mode control for a class of uncertain nonlinear system," *2018 Chinese Automation Congress*, pp. 3534–3538, 2018.

[11] C. Kwan and F. Lewis, "Robust backstepping control of nonlinear systems using neural networks," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* ***30***, pp. 753–766, 2000.

[12] Y. Li, S. Qiang, X. Zhuang, and O. Kaynak, "Robust and adaptive backstepping control for nonlinear systems using rbf neural networks," *IEEE Transactions on Neural Networks* ***15***, pp. 693–701, 2004.

[13] J. A. K. Suykens, "Support vector machines: A nonlinear modelling and control perspective," *European Journal of Control* ***7***, pp. 311–327, 2001.

[14] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing* ***14***, pp. 199–222, 2004.

[15] Y. Shi, D. Sun, Q. Wang, S. Nian, and L. Xiang, "A nonlinear model predictive control based on least squares support vector machines narx model," *2007 International Conference on Machine Learning and Cybernetics*, pp. 721–725, 2007.

[16] F. Abdessemed, "SVM-based control system for a robot manipulator," *International Journal of Advanced Robotic Systems* ***9***, pp. 247–265, 2012.

[17] T. A. Mahmoud and L. M. Elshenawy, "Direct adaptive control based on ls-svm inverse model for nonlinear systems," *2014 19th International Conference on Methods and Models in Automation and Robotics*, pp. 693–698, 2014.

[18] Y. Zhang, "Support vector machine classification algorithm and its application," in *Information Computing and Applications* (C. Liu, L. Wang, and A. Yang, eds.), pp. 179–186, Third International Conference, ICICA 2012, Part 2, Springer, Berlin, Heidelberg, 2012.

[19] E. Finoki, V. Nerguizian, and M. Saad, "Simulation of non linear flight control using back stepping method," *Proceedings of the 2nd International Conference of Control, Dynamic Systems, and Robotics*, 2015.

[20] B. Ibari, L. Benchikh, A. R. H. Elhachimi, and Z. Ahmed-Foitih, "Backstepping approach for autonomous mobile robot trajectory tracking," *Indonesian Journal of Electrical Engineering and Computer Science* **2**, p. 478–485, 2016.

[21] T. Nomura, Y. Kitsuka, H. Suemitsu, and T. Matsuo, "Adaptive backstepping control for a two-wheeled autonomous robot," *2009 ICCAS-SICE*, pp. 4687–4692, 2009.

[22] C. Pukdeboon, "A review of fundamentals of lyapunov theory," *Jounal of Applied Sciences* ***10***, pp. 55–61, 2011.

[23] J. A. Farrell, M. Polycarpou, M. Sharma, and W. Dong, "Command filtered backstepping," *IEEE Transactions on Automatic Control* ***54***, pp. 1391–1395, 2009.

[24] A. El Kharki, Z. Boulghasoul, L. Et-Taaj, Z. Kandoussi, and A. Elbacha, "Real time implementation of backstepping control for high performances induction motor drive," *2019 4th World Conference on Complex Systems*, pp. 1–8, 2019.

[25] C. L. P. Chen, G. Wen, Y. Liu, and Z. Liu, "Observer-based adaptive backstepping consensus tracking control for high-order nonlinear semi-strict-feedback multiagent systems," *IEEE Transactions on Cybernetics* ***46***, pp. 1591–1601, 2016.

[26] S. Zhaoqing and C. Yao, "Nonlinear adaptive direct generalized predictive control based on LS-SVR algorithm," *2012 International Conference on Industrial Control and Electronics Engineering*, pp. 1543–1545, 2012.

[27] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters* ***9***, pp. 293–300, 1999.

[28] J. A. K. Suykens, J. Vandewalle, and B. De Moor, "Optimal control by least squares support vector machines," *Neural Networks* ***14***, pp. 23–35, 2001.