UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
INGENIERÍA ELÉCTRICA – CONTROL

CONTINUOUS SLIDING MODE CONTROL OF DIFFERENTIAL DRIVE ROBOT

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRA EN INGENIERÍA

PRESENTA:
FRIDA GAIL ROJAS CONTRERAS

TUTOR PRINCIPAL:
DR. LEONID FRIDMAN, FACULTAD DE INGENIERÍA, UNAM

COTUTOR:
DR. VÍCTOR JAVIER GONZÁLEZ VILLELA, FACULTAD DE INGENIERÍA, UNAM

CIUDAD DE MÉXICO, JULIO 2017

JURADO ASIGNADO:

Presidente:
DR. MARCO ARTEAGA, FACULTAD DE INGENIERÍA, UNAM
Secretario:
DR. JAIME A. MORENO, INSTITUTO DE INGENIERÍA
Vocal:
DR. LEONID FRIDMAN, FACULTAD DE INGENIERÍA, UNAM
1 er. Vocal:
DR. VÍCTOR JAVIER GONZÁLEZ VILLELA, FACULTAD DE INGENIERÍA, UNAM
2 do. Vocal:
DR. JORGE DÁVILA, INSTITUTO POLITÉCNICO NACIONAL

Lugar o lugares donde se realizó la tesis: Facultad de Ingeniería, UNAM

TUTOR DE TESIS:
DR. LEONID FRIDMAN

FIRMA

CONTROL POR MODOS DESLIZANTES CONTINUOS DE ROBOT DIFERENCIAL

Frida Gail Rojas Contreras

Resumen

Seguimiento de trayectoria con rechazo de perturbaciones para un robot diferencial con ruedas considerando dos modelos de grado relativo es analizado. El primer modelo considera las velocidades de ruedas como entradas de control y, en consecuencia, se utiliza un sistema cuyas salidas tienen grado relativo 1 *con respecto a la posición. En este caso, el algoritmo de Super Twisting es implementado. El segundo modelo considera voltajes de ruedas como entradas de control y, en consecuencia, se utiliza un sistema cuyas salidas tienen grado relativo* 2 *con respecto a la posición. En este caso, el algoritmo de Continuous Twisting es implementado. La comparación de ambas estrategias se realiza en simulaciones y con pruebas experimentales.*

Palabras clave: Modos Deslizantes Continuos, Robot Diferencial, Sistema No Holónomo

CONTINUOUS SLIDING MODE CONTROL OF DIFFERENTIAL DRIVE ROBOT

Frida Gail Rojas Contreras

Abstract

Keywords: Continuous Sliding Modes, Nonholonomic System, Differential Drive Robot

Trajectory tracking with disturbance rejection for a differential drive robot with wheels considering two models of relative degree is analyzed. The first model considers wheels velocities as control inputs and hence a system which has positions as outputs with relative degree 1 is used. In this case “Super Twisting Algorithm” is implemented. The second model considers motor voltages as control inputs and hence a system which has outputs with relative degree 2 is used. In this case “Continuous Twisting Algorithm” is implemented. The comparison of the strategies is done in simulation and experimentally.

Acknowledgements

Thanks to professor Leonid Fridman, for his guidance, the patience he showed and faith he had on me during the development of this thesis.

Thanks to professor Victor Javier Gonzalez Villela, for his collaboration with the experimental setup he provided and the knowledge he shared in this thesis.

Thanks to professor Jaime Moreno, for his guidance and patience.

Thanks to PhD Alberto Ismael Castillo Lopez, for his advices and constant and accurate feedback for this thesis.

Thanks to M.Sc. Neftali Elorza Lopez, for his guidance and support during the first stage of the thesis.

Thanks to the sliding mode control laboratory and the control group, for the constant feedback provided during the seminars.

Last but not least, thanks to my family, the woman I most admire in this world, my mother, my best friend, my brother, and my treasured father. Not one thing that I have accomplished could have met the finish line if they had not been there to support me. I love you.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Mobile robots are ubiquitous nowadays in industrial and service applications. Their design depend on the environment in which they will perform their tasks. In the case of terrestrial robots, locomotion can be achieved by using robotic "legs" as the robot shown in Figure 1.1 developed by the Mechatronics Research Group at UNAM.

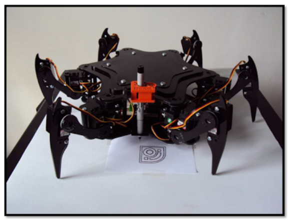
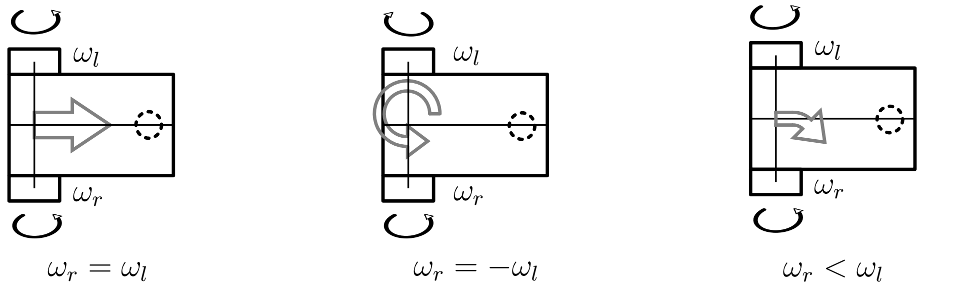


Figure 1.1: Legged terrestrial mobile robot

Locomotion in terrestrial robots can also be achieved using wheels. Particularly, wheeled robots are sought the most because of their versatibility and relatively low cost in terms of maintenance and repairs. Depending on the type of wheels the robot is equipped with, it acquires one of a few mobility configurations that have been broadly studied in [de Wit et al., 2012]. Some examples of wheeled robots are presented in [Carlisle, 2000; Rooks, 2001]. Particularly, the differential drive robot (DDR) is one of the most common configurations of terrestrial robots with wheels.

A DDR consists of two independently actuated wheels, each one driven by a DC motor. Commonly, it is also equipped with a wheel that is not actuaded but moves freely; its only function is to provide support for the DDR's chassis.

Depending on the direction and the magnitude of the speed of the wheels, the DDR can move in three different ways. First, if the wheels rotate with the same speed in the same direction then the robot can move forwards or backwards, as shown in Figure 1.2a. Second, if the wheels rotate with the same speed but in opposite direction then the robot purely rotates around its own axis, as shown in Figure 1.2b. Third, if the wheels rotate with different speed but in the same direction then the robot performs a translation and a rotation at the same time, as illustrated in Figure 1.2c.

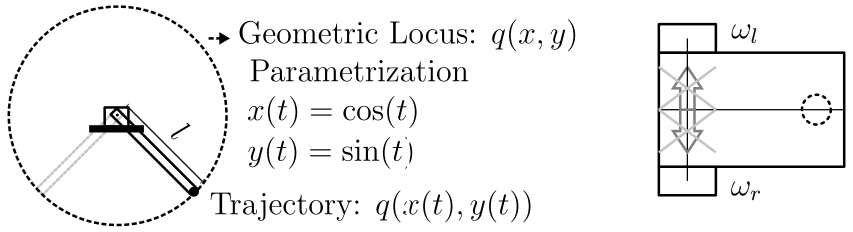


(a) Pure linear displacement **(b) Pure rotation** **(c) Rotation and translation**

Figure 1.2: DDR's motion

This seemingly simple robot has particular dynamics that have certain properties that arise from having nonholonomic constraints (NHC). It is easier to understand NHC once holonomic constraints are explained.

A holonomic constraint is a geometrical constraint, i.e., a constraint that can be expressed by functions of positions. An example of this case are fully actuated serial manipulators for which reaching a certain point in its configuration space is only restricted by the length of its links as shown in Figure 1.3a.



(a) Holonomic **(b) Nonholonomic**

Figure 1.3: Types of constraints

Formally, the differential form of the holonomic constraint is exactly integrable and the result can be expressed only in terms of positions. The latter is equivalent to being able to generate a geometric locus which is a set of points whose location is determined by constraints. So, what is actually occurring is that a geometric locus is generated according to the end effector's geometry. If the geometric locus is parametrized, for example by a temporal variable t, then this parametrization helps to establish a trajectory $q(x(t), y(t))$ in the geometric locus $q(x, y)$. Hence, a trajectory generated by the end effector is a specific point that moves along the space of the geometric locus.

However, these are not the type of constraints that a DDR has. Since a DDR is equipped with fixed wheels lateral motion of the DDR is not possible as shown in Figure 1.3b no matter what combination of velocities is given. This restriction of motion can be represented with velocity constraints that are directly linked with pure rolling contraints of the wheels.

Formally, the differential form of these velocity constraints is not integrable and, therefore, these constraints cannot be expressed as constraints solely in terms of the positions of the DDR. Since the constraint cannot be a function of the positions then a geometric locus cannot be defined and therefore no trajectory can exist for lateral motion.

However, the fact that the DDR cannot move in lateral motion does not mean the robot cannot reach any position in its configuration space since it can maneuver its way to the final posture. It only means there are velocity constraints that need to be satisfied during motion. Velocity constraints are also called NHC. Systems that have NHC are called nonholonomic systems and they represent a wide class of mechanical systems.

Summarizing, nonholonomic mechanical systems are systems with constraints on their velocities which cannot be obtained from position constraints. In order to design control laws to drive the DDR from a start state to a goal state in finite time it is necessary to study the DDR as a mathematical model that represents the dynamics of a nonholonomic system.

If the chosen output is the posture of the DDR, depending on where the point of interest is located on the chassis, its dynamics will vary, and consequently, its mathematical model. There is a lot of work in literature in two particular chosen outputs. First, the middle point between the two actuated wheels, which is denoted $\boldsymbol{a} = \begin{bmatrix} x_a, & y_a \end{bmatrix}^\top$. Second, a point located at a distance h from point a in the body symmetry axis, which is denoted as $\boldsymbol{p} = \begin{bmatrix} x, & y \end{bmatrix}^\top$. These chosen outputs are illustrated in Figure 1.4. In both cases, the control inputs are the left and right wheel velocities ω_l and ω_r, respectively.

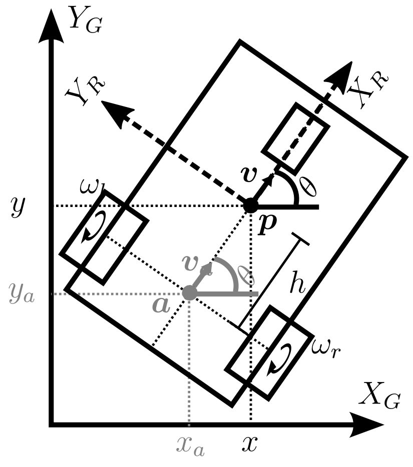


Figure 1.4: Choice of outputs

1.1 Motivation

One way of designing a certain motion for the DDR to follow is by chosing one of these two outputs and performing trajectory tracking. Either way, while in motion, the DDR faces disturbances such as frictions as well as unmodelled dynamics and other uncertainties which are always present in real life applications. These phenomena need to be taken into account when control laws are designed for a DDR.

1.2 State of the art

There are a lot of papers in literature that implement controllers to accomplish trajectory tracking on DDRs. For example, in [d'Andrea Novel et al., 1992] and [Oriolo et al., 2002] the controller is based on dynamic feedback linearization laws which guarantee exponential convergence to the reference trajectory for a model based on the dynamics of point $\boldsymbol{a}$.

As mentioned in a previous section, changing the choice of output to $\boldsymbol{p}$ changes the dynamics of the chosen output as explained with detail in Section 3.1. This is done in [Diaz and Kelly, 2016] and a linear control algorithm is implemented using a proportional term to perform trajectory tracking on a nominal system which guarantees exponential convergence to the reference trajectory.

However, in order for all these controllers to work properly in non ideal conditions, perfect knowledge of the system and disturbances is required.

1.3 Methodology

Having perfect knowledge of the system and perturbations is perhaps an impossible task. In fact it is common to think of a system as a "black box" where the only information available is about the inputs and outputs of the system such as relative degree and no knowledge of its internal dynamics. Particularly in this case, when a controller is designed it is important to include uncertainties and disturbances phenomena.

One technique to deal with uncertainties and disturbances for "black box" problems are sliding mode algorithms. The idea that sliding mode control presents in order to provide robustness and a high level of precision lies on a strategy that consists of a fast response to face changes in the system's behavior and enforce a given constraint for the system.

1.3.1 First generation: Standard sliding modes

First-order sliding modes (FOSM) define a variable of interest which is intended to be driven to zero in finite time in spite of uncertainties and disturbances. In [Mu et al., 2015] trajectory tracking control for a DDR, based on Equation (2.1) using a FOSM technique is presented. In this paper, reference coordinates $x_{ad}, y_{ad}, \theta_{ad}$ and reference inputs v_{ad}, ω_d are stated. Using a nonsingular transformation matrix the error coordinates $x_a - x_{ad}$, $y_a - y_{ad}$ and $\theta_a - \theta_{ad}$ form two sliding surfaces which are considered in the FOSM controller.

The drawback of FOSM control laws is that they are discontinuous controllers that inherently have chattering which means implementing a high frequency control signal that in practical implementations could compromise the actuator's integrity. Moreover, chattering inherently demands energy from the system which makes this type of controllers an expensive solution. Some workarounds have been proposed to decrease chattering phenomenon for FOSM ([Shtessel et al., 2014]), however, they imply loss of accuracy and robustness. Another limitation for FOSM is that in order to assure finite time convergence and disturbance rejection, it is required that the variable of interest is an output of relative degree 1 w.r.t. to the system's control input.

1.3.2 Second generation: Second order sliding modes

Second order sliding modes guarantee finite time convergence to zero for the variable of interest and its derivative in spite of bounded disturbances. In the control design, if the system has an output with relative degree 2 w.r.t. to its inputs there is no need to design a sliding manifold. Their main drawback is that the resulting control signal remains discontinuous and, consequently, chattering phenomenon is present ([Boiko et al., 2004]).

1.3.3 Third generation: Super-Twisting algorithm

Super-Twisting algorithm (STA) is a second order sliding mode that assures the convergence of the variable of interest in finite time in spite of Lipschitz disturbances generating a continuous control signal. For control design, if the system's output has relative degree 1 no sliding surface is required, however, if the the system's output have relative degree greater than one then a sliding surface is needed and finite time convergence cannot longer be guaranteed.

In [Solea and Cernega, 2015] STA is implemented as a part of a control law to drive tracking position and orientation errors derived from a reference model to zero.

1.3.4 Fourth generation: High-order sliding modes

The limitation that relative degree 1 is required and the presence of chattering phenomenon from FOSM motivated the idea of high-order sliding modes (HOSM). This strategy introduces sliding modes that act on a high order time derivative of the system deviation from the given constraint instead of directly influencing only the first deviation derivative as occurs in FOSM. This makes the high-frequency switching appear in a higher derivative of the variable of interest reducing chattering phenomenon but the generated control signal is still discontinuous. A HOSM controller of $k-th$ order drives the variable of interest and its $k-1$ time derivatives to zero.

In [Davila, 2013] exact tracking using HOSM is performed for a nonlinear system in the presence of matched and unmatched disturbances.

1.3.5 Fifth generation: Continuous high order sliding modes

In [Bhat and Bernstein, 2000; Orlov et al., 2011; Sánchez and Moreno, 2014; Efimov and Perruquetti, 2016; Ríos et al., 2016], algorithms based on homogeneity theory guarantee finite time convergence of their proposed errors on nominal systems using continuous control signals.

STA motivates the research of continuous control signals that include sliding mode controllers strength to provide exact disturbance rejection for systems with output of relative degree greater than 1 ([Torres-González et al., 2015; Moreno et al., 2016]). Continuous high order sliding modes (CHOSM) can compensate exactly Lipschitz disturbances using a continuous control signal and also guarantee finite time convergence of the variable of interest to zero. These controllers only require information of the variable of interest and its derivative.

1.4 Contribution

The contribution of this thesis is to solve robust trajectory tracking for DDR by implementing continuous sliding modes algorithms that, on one hand, provide exact disturbance rejection and, on the other hand, have less chattering effect than discontinuous sliding mode controllers.

1.5 Objective

The goal of this thesis is to solve trajectory tracking problem using two models of different relative degree of a DDR by implementing continuous sliding mode algorithms.

1.6 Thesis structure

Chapter 2 analyzes the properties of the choice of $\boldsymbol{a}$ as the output of the DDR and the workaround to implement continuous sliding modes in a DDR.

Chapter 3 discusses modelling and control design to solve trajectory tracking for a DDR based on a system with wheel velocities as control inputs. Particularly, continuous sliding mode algorithm “Super Twisting algorithm” advantages are shown. At the end of the chapter, simulations and experiments show the implemented controllers performances.

Chapter 4 discusses modelling and control design to solve trajectory tracking for a DDR based on a system with motor voltages as control inputs. Particularly, continuous sliding mode algorithm “Continuous Twisting algorithm” advantages are shown. At the end of the chapter, simulations and experiments show the implemented controllers performances.

Chapter 5 presents the discussion of the simulation and experimental results, the conclusions of this thesis and future work.

Chapter 2

Theoretical framework

If the chosen output is the posture of the robot, depending on where the point of interest is located on the chasis, its dynamics will vary. The property of interest for these dynamics is the relative degree of XY positions as outputs with respect to wheel velocities as inputs.

2.1 Modelling for output $\boldsymbol{a}$

Consider the nominal kinematic model for a differential drive robot (DDR), shown in Figure 2.1, as

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix}}_{g(\boldsymbol{X_a})} \begin{bmatrix} v_a \\ \omega \end{bmatrix}, \tag{2.1}$$

where the posture state vector is denoted by $\boldsymbol{X_a} = \begin{bmatrix} x_a, & y_a, & \theta \end{bmatrix}^\top$. Vector $\boldsymbol{a} = \begin{bmatrix} x_a, & y_a \end{bmatrix}^\top \in \mathbb{R}^2$ and θ denotes the orientation, both w.r.t. the global frame $X_G Y_G$. The control inputs for the DDR are v_a and ω which represent the linear and angular velocity, respectively.

Notice that $g(\boldsymbol{X_a}) \in \mathbb{R}^{3\times 2}$ is not a square matrix. In other words, System (2.1) has 2 control inputs and one would like to control 3 outputs. This is not possible using only 2 control inputs and, therefore, one of the 3 outputs cannot be controlled.

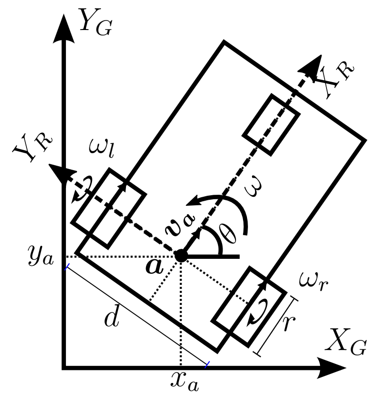


Figure 2.1: DDR in point $\boldsymbol{a}$

System (2.1) is expressed in terms of linear and angular velocity. However, the actual control inputs of a DDR with wheels are its right and left wheel velocities, ω_r and ω_l, respectively. Transformation M maps these 2 wheel velocities to the linear and angular velocities as follows

$$\begin{bmatrix} v_a \\ \omega \end{bmatrix} = \underbrace{\begin{bmatrix} +a_1 & +a_1 \\ +a_2 & -a_2 \end{bmatrix}}_{M} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \tag{2.2}$$

where $a_1 = r/2 > 0$ and $a_2 = r/d > 0$ with r the radius of the wheel and d the distance between the wheels. Notice that $\det\big(M\big) = -2a_1a_2 \neq 0$ and, consequently, M^{-1} exists.

Rewritting System (2.1) in terms of ω_r and ω_l using Equation (2.2) yields

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} +a_1\cos(\theta) & +a_1\cos(\theta) \\ +a_1\sin(\theta) & +a_1\sin(\theta) \\ +a_2 & -a_2 \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}. \tag{2.3}$$

For analysis purposes, Equation (2.3) can be rewritten in a more general form as

$$\begin{aligned} \dot{\boldsymbol{X}}_{\boldsymbol{a}} =& \boldsymbol{g_1}(\boldsymbol{X_a})u_1 + \boldsymbol{g_1}(\boldsymbol{X_a})u_2 \\ \boldsymbol{Y} =& \boldsymbol{h}(\boldsymbol{X_a}) \end{aligned} \tag{2.4}$$

where the state vector is $\boldsymbol{X_a} \in \mathbb{R}^3$, the measured output is $\boldsymbol{Y} \in \mathbb{R}^3$, the control inputs are $u_1 = \omega_r$ and $u_2 = \omega_l$ and

$$\boldsymbol{g_1}(\boldsymbol{X_a}) = \begin{bmatrix} +a_1\cos(\theta) \\ +a_1\sin(\theta) \\ +a_2 \end{bmatrix}, \quad \boldsymbol{g_2}(\boldsymbol{X_a}) = \begin{bmatrix} +a_1\cos(\theta) \\ +a_1\sin(\theta) \\ -a_2 \end{bmatrix}, \quad \boldsymbol{h}(\boldsymbol{X_a}) = \begin{bmatrix} h_1(\boldsymbol{X_a}) \\ h_2(\boldsymbol{X_a}) \\ h_3(\boldsymbol{X_a}) \end{bmatrix} = \begin{bmatrix} x_a \\ y_a \\ \theta_a \end{bmatrix}$$

2.2 Analysis of relative degree

Sliding mode controllers require the knowledge of the relative degree for the outputs of Equation (2.4) which is a MIMO system.

Definition 2.1 (Relative degree for MIMO system) *([Sastry and Bodson, 2011, p. 286]) For the multiple input multiple output (MIMO) case, consider the square system of the form*

$$\begin{aligned}\dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g_1}(\boldsymbol{x})u_1 + \ldots + \boldsymbol{g_p}(\boldsymbol{x})u_p \\ y_i &= h_i(\boldsymbol{x}),\ i = 1, \ldots, p\end{aligned} \tag{2.5}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^p$, $\boldsymbol{y} \in \mathbb{R}^p$ and $\boldsymbol{f}, \boldsymbol{g_i}, h_j$ are assumed smooth. Let $L_e w = \left[\partial w(\boldsymbol{x})/\partial \boldsymbol{x}\right] e(\mathbf{x})$ denote the Lie derivative of w along e. Differentiating the $j-th$ output y_j w.r.t. time yields

$$y_j^{r_j} = L_f h_j + \sum_{i=1}^{p} Lg_i(L_f^{r_j-1} h_j)u_i \tag{2.6}$$

with at least one of the $L_{g_i}(L_f^{r_j-1}h_j) \neq 0$ for some $\boldsymbol{x}$. Define $p \times p$ matrix $B(\boldsymbol{x})$ as

$$B(\boldsymbol{x}) = \begin{bmatrix} L_{\boldsymbol{g_1}} L_f^{r_j-1} h_1 & \ldots & L_{\boldsymbol{g_p}} L_f^{r_j-1} h_1 \\ \vdots & \vdots & \vdots \\ L_{\boldsymbol{g_1}} L_f^{r_j-1} h_p & \ldots & L_{\boldsymbol{g_p}} L_f^{r_j-1} h_p \end{bmatrix} \tag{2.7}$$

Then, Equation (2.6) *may be written as*

$$\begin{bmatrix} y_1^{(r_1)} \\ \vdots \\ y_p^{(r_p)} \end{bmatrix} = \begin{bmatrix} L_f^{r_1} h_1 \\ \vdots \\ L_f^{r_p} h_p \end{bmatrix} + B(\boldsymbol{x}) \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \tag{2.8}$$

If $B(\boldsymbol{x}) \in \mathbb{R}^{p\times p}$ is bounded away from singularity, decoupling can be achieved by linearization since $B^{-1}(\boldsymbol{x})$ exists $\forall\, \boldsymbol{x}$ and has bounded norm. The control law that achieves the latter is referred to as a static feedback linearizing control law.

However, if $B(x)$ is singular and the drift term $\boldsymbol{f}(\boldsymbol{x})$ in Equation (2.6) *is not in the range of $B(\boldsymbol{x})$, linearization and decoupling may still be achieved using a dynamic state feedback control law. To do so this technique requires adding dynamics to the controller and its aimed to influence the relative degree of the outputs of the system.*

In tis case, $m = 2$ and $p = 3$. To make $m = p$ as Definition 2.1 requires for Equation (2.4), measured output $\boldsymbol{Y}$ is reduced to controlled output $\bar{\boldsymbol{Y}} = \begin{bmatrix} Y_1, & Y_2 \end{bmatrix}^\top \mathbb{R}^{p=2}$. Notice that Equation (2.4) is a driftless system since $\boldsymbol{f}(\boldsymbol{x})$ from Equation (2.5) is $\mathbf{0}$. Differentiating w.r.t. time output $\bar{\boldsymbol{Y}}$ yields

$$\begin{aligned}\dot{\bar{Y}}_1 &= \left[L_{g_1}\left(L_f^0 h_1(\boldsymbol{X_a})\right), \quad L_{g_2}\left(L_f^0 h_1(\boldsymbol{X_a})\right)\right] \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \left[\cos(\theta_a), \quad \cos(\theta_a)\right] \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \\ \dot{\bar{Y}}_2 &= \left[L_{g_1}\left(L_f^0 h_2(\boldsymbol{X_a})\right), \quad L_{g_2}\left(L_f^0 h_2(\boldsymbol{X_a})\right)\right] \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \left[\sin(\theta_a), \quad \sin(\theta_a)\right] \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}\end{aligned} \tag{2.9}$$

Hence, one can form matrix $B(\boldsymbol{X_a})$ from Equation (2.7) for System (2.4) as

$$B(\boldsymbol{X_a}) = \begin{bmatrix} \cos(\theta_a), & \cos(\theta_a) \\ \sin(\theta_a), & \sin(\theta_a) \end{bmatrix}. \tag{2.10}$$

Matrix $B(\boldsymbol{X_a})$ is singular since $\det\big(B(\boldsymbol{x})\big) = 0$. This means a singularity is present for System (2.4). Consequently, the decoupling matrix $B(\boldsymbol{x})$ looses rank ([Isidori, 2013]) and this causes the outputs of System (2.4) to not have a well defined relative degree ([Hirschorn, 2002]).

As mentioned in Definition 2.1, one way to overcome the singularity problem is using dynamic feedback linearization. Another strategy to deal with the problem is to use discontinuous controllers but they inherently have chattering effect. Alternatively, since relative degree is not well defined for output $\boldsymbol{a}$, then proposing another output, denoted by $\boldsymbol{p}$, that yields a well defined relative degree is possible as will be seen in the next chapter.

Chapter 3

Velocities as control inputs

3.1 Modelling

[Diaz and Kelly, 2016] propose a paremeter h that locates point $\boldsymbol{p} = \begin{bmatrix} x, & y \end{bmatrix}^\top \in \mathbb{R}^2$ w.r.t. the $X_G Y_G$ coordinate frame, as illustrated in Figure 3.1, is defined.

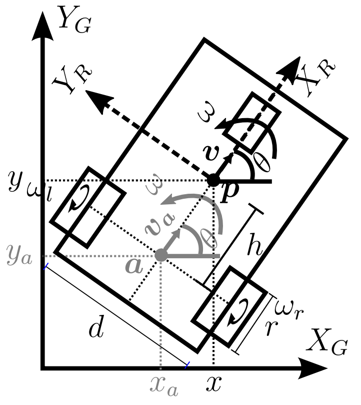


Figure 3.1: DDR in point $\boldsymbol{p}$

Geometrically, point $\boldsymbol{p}$ is described as

$$\begin{aligned} x =& x_a + h\cos(\theta) \\ y =& y_a + h\sin(\theta) \end{aligned} \tag{3.1}$$

Differentiating Equation (3.1) w.r.t. time yields,

$$\begin{aligned} \dot{x} =& \dot{x}_a - h\dot{\theta}\sin(\theta) \\ \dot{y} =& \dot{y}_a + h\dot{\theta}\cos(\theta) \\ \dot{\theta} =& \omega \end{aligned} \tag{3.2}$$

Substituting wheel mapping from Equation (2.2) and the known dynamics from point

$\boldsymbol{a}$ from Equation (2.3) in Equation (3.2) results in

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} +a_1 \cos(\theta) - h a_2 \sin(\theta) & +a_1 \cos(\theta) + h a_2 \sin(\theta) \\ +a_1 \sin(\theta) + h a_2 \cos(\theta) & +a_1 \sin(\theta) - h a_2 \cos(\theta) \\ +a_2 & -a_2 \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}. \tag{3.3}$$

Analogously to Equation (2.4), Equation (3.3) can be rewritten in a more general form as

$$\begin{aligned} \dot{\boldsymbol{X}} &= \boldsymbol{g_1}(\boldsymbol{X}) u_1 + \boldsymbol{g_2}(\boldsymbol{X}) u_2 \\ \boldsymbol{Y} &= \boldsymbol{h}(\boldsymbol{X}) \end{aligned} \tag{3.4}$$

where the state vector is $\boldsymbol{X} = \begin{bmatrix} x, & y, & \theta \end{bmatrix}^\top \in \mathbb{R}^3$, the measured output is $\boldsymbol{Y} = \begin{bmatrix} x, & y, & \theta \end{bmatrix}^\top \in \mathbb{R}^3$, the control inputs are $u_1 = \omega_r$ and $u_2 = \omega_l$ and

$$\boldsymbol{g_1}(\boldsymbol{X}) = \begin{bmatrix} +a_1 \cos(\theta) - h a_2 \sin(\theta) \\ +a_1 \sin(\theta) + h a_2 \cos(\theta) \\ +a_2 \end{bmatrix}, \quad \boldsymbol{g_2}(\boldsymbol{X}) = \begin{bmatrix} +a_1 \cos(\theta) + h a_2 \sin(\theta) \\ +a_1 \sin(\theta) - h a_2 \cos(\theta) \\ -a_2 \end{bmatrix},$$

$$\boldsymbol{h}(\boldsymbol{X}) = \begin{bmatrix} h_1(\boldsymbol{X}) \\ h_2(\boldsymbol{X}) \\ h_3(\boldsymbol{X}) \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}.$$

Once again, to make the $m = p = 2$ for Equation (3.4), measured output $\boldsymbol{Y}$ is reduced to controlled output $\bar{\boldsymbol{Y}} = \begin{bmatrix} Y_1, & Y_2 \end{bmatrix}^\top \in \mathbb{R}^2$. Notice that Equation (3.4) is also a driftless system, i.e., $\boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{0}$. Differentiating w.r.t. time output $\bar{\boldsymbol{Y}}$ yields

$$\begin{aligned} \dot{Y}_1 &= \begin{bmatrix} L_{g_1}\left(L_f^0 h_1(\boldsymbol{X})\right), & L_{g_1}\left(L_f^0 h_1(\boldsymbol{X})\right) \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \\ &= \begin{bmatrix} +a_1 \cos(\theta) - h a_2 \sin(\theta), & +a_1 \cos(\theta) + h a_2 \sin(\theta) \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}, \\ \dot{Y}_2 &= \begin{bmatrix} L_{g_1}\left(L_f^0 h_2(\boldsymbol{X})\right), & L_{g_2}\left(L_f^0 h_2(\boldsymbol{X})\right) \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \\ &= \begin{bmatrix} +a_1 \sin(\theta) + h a_2 \cos(\theta), & +a_1 \sin(\theta) - h a_2 \cos(\theta) \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}. \end{aligned} \tag{3.5}$$

Hence, matrix $B(\boldsymbol{X})$ from Equation (2.7) becomes

$$B(\boldsymbol{X}) = \begin{bmatrix} +a_1 \cos(\theta) - h a_2 \sin(\theta), & +a_1 \cos(\theta) + h a_2 \sin(\theta) \\ +a_1 \sin(\theta) + h a_2 \cos(\theta), & +a_1 \sin(\theta) - h a_2 \cos(\theta) \end{bmatrix}, \tag{3.6}$$

which is a nonsingular matrix since $\det\left(B(\boldsymbol{X})\right) = -2 a_1 a_2 h \neq 0$ if and only if $h \neq 0$. In other words, the relative degree for outputs of Equation (3.4) is well defined. Particularly, the relative degree of this system is $\begin{bmatrix} 1, & 1 \end{bmatrix}$ for outputs XY positions and zero dynamics is represented by the evolution of the orientation θ.

3.2 Disturbed case

It is assumed that the disturbances are matched to the control inputs in order to preserve the nonholonomic constraints (NHC) inherent to the kinematic model. The disturbed version of System (2.1) can be written as

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_a + \rho_a \\ \omega + \rho_\omega \end{bmatrix}, \tag{3.7}$$

where $\rho_a = \rho_a(t)$ and $\rho_\omega = \rho_\omega(t)$ are matched disturbances. It is assumed that these disturbances are Lipschitz functions, i.e., the perturbation time derivatives are globally bounded by

$$|\dot{\rho_a}| \leq L_{v_a}, \quad |\dot{\rho_\omega}| \leq L_\omega.$$

Assuming that for disturbance terms

$$\begin{bmatrix} v_a \\ \omega \end{bmatrix} + \begin{bmatrix} \rho_a \\ \rho_\omega \end{bmatrix} = M \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} + M \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix}, \tag{3.8}$$

occurs with $\rho_r = \rho_r(t, \omega_r)$, $\rho_l = \rho_l(t, \omega_l)$ and

$$|\dot{\rho_r}| \leq L_r, \quad |\dot{\rho_l}| \leq L_l.$$

Using Equation (3.8), System (3.7) can be rewritten in terms of wheel velocities as follows

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} +a_1\cos(\theta) & +a_1\cos(\theta) \\ +a_1\sin(\theta) & +a_1\sin(\theta) \\ +a_2 & -a_2 \end{bmatrix} \begin{bmatrix} \omega_r + \rho_r \\ \omega_l + \rho_l \end{bmatrix}. \tag{3.9}$$

Analogously to the nominal case for point $\boldsymbol{p}$, differentiating Equation (3.1) w.r.t. time yields,

$$\begin{aligned} \dot{x} &= \dot{x}_a - h\dot{\theta}\sin(\theta), \\ \dot{y} &= \dot{y}_a + h\dot{\theta}\cos(\theta), \\ \dot{\theta} &= \omega + \rho_\omega. \end{aligned} \tag{3.10}$$

Substituting wheel mapping from Equation (3.8) and the known disturbed dynamics of point $\boldsymbol{a}$ from Equation (3.9) in Equation (3.10) results in

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} +a_1\cos(\theta) - ha_2\sin(\theta) & +a_1\cos(\theta) + ha_2\sin(\theta) \\ +a_1\sin(\theta) + ha_2\cos(\theta) & +a_1\sin(\theta) - ha_2\cos(\theta) \\ +a_2 & -a_2 \end{bmatrix} \begin{bmatrix} \omega_r + \rho_r \\ \omega_l + \rho_l \end{bmatrix}. \tag{3.11}$$

3.3 Control Design

In order to avoid verbose notation, Equation (3.11) is rewritten in a compact form as follows

$$\begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} D(\theta) \\ \boldsymbol{\phi}^\top \end{bmatrix} \boldsymbol{u} + \underbrace{\begin{bmatrix} D(\theta) \\ \boldsymbol{\phi}^\top \end{bmatrix} \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix}}_{\boldsymbol{\rho}} = \begin{bmatrix} D(\theta) \\ \boldsymbol{\phi}^\top \end{bmatrix} \boldsymbol{u} + \boldsymbol{\rho}, \tag{3.12}$$

where $\dot{\boldsymbol{p}} = \begin{bmatrix}\dot{x}, & \dot{y}\end{bmatrix}^\top \in \mathbb{R}^2$, $D(\theta) = B(\boldsymbol{X})$, $\boldsymbol{\phi}^\top = \begin{bmatrix}+a_2, & -a_2\end{bmatrix}$, $\boldsymbol{u} = \begin{bmatrix}\omega_r, & \omega_l\end{bmatrix}^\top \in \mathbb{R}^2$ and $\boldsymbol{\rho} = \begin{bmatrix}\rho_1, & \rho_2, & \rho_3\end{bmatrix}^\top \in \mathbb{R}^3$ are disturbances for which $|\dot{\rho}_i| \leq L_i,\ i = 1, 2, 3.$

3.3.1 Control objective

Since a disturbed model is considered, sliding mode technique is implemented because of its robusteness in the sense of exact disturbance compensation properties. Trajectory tracking for a DDR based on System (3.12) implementing a continuous sliding mode algorithm is intended. In other words, control objective is to drive position errors to zero in finite-time regardless of perturbations.

Suppose $\boldsymbol{p_d}(t) = \begin{bmatrix}x_d(t), & y_d(t)\end{bmatrix}^\top \in \mathbb{R}^2$ is continuous and differentiable such that $\dot{\boldsymbol{p}}_{\boldsymbol{d}}(t) = \begin{bmatrix}\dot{x}_d(t), & \dot{y}_d(t)\end{bmatrix}^\top \in \mathbb{R}^2$ exists. Also, $\boldsymbol{p_d}(t)$ and $\dot{\boldsymbol{p}}_{\boldsymbol{d}}(t)$ are known. In the following (t) is omitted to avoid verbose notation. Attempting to do trajectory tracking for x and y position coordinates forces orientation dynamics to be the zero dynamics of System (3.12) making θ a measured uncontrolled output.

3.3.2 Error dynamics

Define error variable $\boldsymbol{e_1}$ as

$$\boldsymbol{e_1} = \begin{bmatrix}x \\ y\end{bmatrix} - \begin{bmatrix}x_d \\ y_d\end{bmatrix} = \boldsymbol{p} - \boldsymbol{p_d} \tag{3.13}$$

Differentiating Equation (3.13) once w.r.t. time, yields the following error dynamics

$$\dot{\boldsymbol{e}}_{\boldsymbol{1}} = \dot{\boldsymbol{p}} - \dot{\boldsymbol{p}}_{\boldsymbol{d}} = D(\theta)\boldsymbol{u} + \begin{bmatrix}\rho_1 \\ \rho_2\end{bmatrix} - \dot{\boldsymbol{p}}_{\boldsymbol{d}} \tag{3.14}$$

Let

$$\boldsymbol{u} = D^{-1}(\theta)\left\{\boldsymbol{\nu} + \dot{\boldsymbol{p}}_{\boldsymbol{d}}\right\}. \tag{3.15}$$

Then Equation (3.14) can be rewritten as

$$\dot{\boldsymbol{e}}_{\boldsymbol{1}} = \boldsymbol{\nu} + \begin{bmatrix}\rho_1 \\ \rho_2\end{bmatrix} \tag{3.16}$$

System (3.16) has the form of two independently disturbed single integrators. In this case, "Super Twisting Algorithm" (STA) (Levant [1998]; Moreno [2009]) can be implemented as follows

$$\begin{aligned}\boldsymbol{\nu} &= -K_1 \lceil \boldsymbol{e_1} \rfloor^{1/2} + \boldsymbol{\eta} \\ \dot{\boldsymbol{\eta}} &= -K_2 \lceil \boldsymbol{e_1} \rfloor^{0}\end{aligned} \tag{3.17}$$

where $K_i = \mathrm{diag}(k_{ix}, k_{iy})$ with $i = 1, 2$ and

$$\lceil \boldsymbol{e_1} \rfloor^{\gamma} = \begin{bmatrix}|x - x_d|^\gamma \mathrm{sign}(x - x_d) \\ |y - y_d|^\gamma \mathrm{sign}(y - y_d)\end{bmatrix} \in \mathbb{R}^2, \text{ and } \boldsymbol{\eta} = \begin{bmatrix}\eta_x \\ \eta_y\end{bmatrix} \in \mathbb{R}^2,$$

Notice that Equation (3.17) is designed as a two independent control; the first channel affects the position tracking error on the x coordinate and the second channel affects coordinate y.

The closed loop from Equation (3.16)-Equation (3.17) can be rewritten as

$$\begin{aligned}\dot{\boldsymbol{e}}_{\boldsymbol{1}} &= -K_1 \lceil \boldsymbol{e}_{\boldsymbol{1}} \rfloor^{1/2} + \underbrace{\boldsymbol{\eta} + \begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix}}_{\boldsymbol{e}_{\boldsymbol{2}}} \\ \dot{\boldsymbol{e}}_{\boldsymbol{2}} &= -K_2 \lceil \boldsymbol{e}_{\boldsymbol{1}} \rfloor^{0} + \begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix}.\end{aligned} \tag{3.18}$$

STA guarantees a second order sliding mode (Levant [1998]; Moreno [2009])

$$\boldsymbol{e}_{\boldsymbol{1}} = \boldsymbol{0},\ \boldsymbol{e}_{\boldsymbol{2}} = \boldsymbol{0} \rightarrow \boldsymbol{\eta} = -\begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix}. \tag{3.19}$$

The block diagram of the closed loop is shown in Figure 3.2.

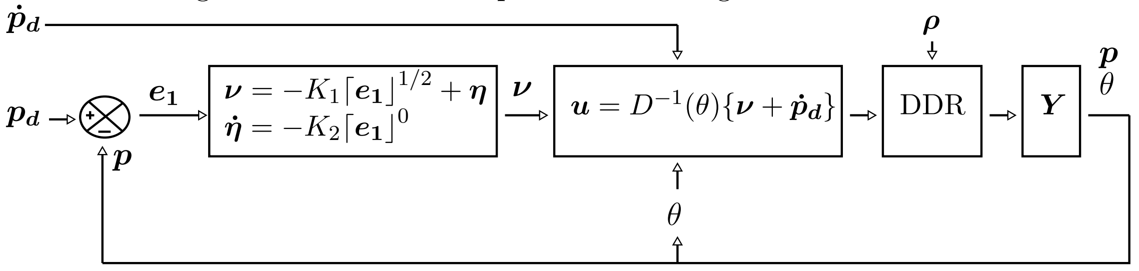

$\dot{\boldsymbol{p}}_{\boldsymbol{d}}$
$\boldsymbol{p}_{\boldsymbol{d}}$
+
−
$\boldsymbol{p}$
$\boldsymbol{e}_{\boldsymbol{1}}$
$\boldsymbol{\nu} = -K_1\lceil \boldsymbol{e}_{\boldsymbol{1}}\rfloor^{1/2} + \boldsymbol{\eta}$
$\dot{\boldsymbol{\eta}} = -K_2\lceil \boldsymbol{e}_{\boldsymbol{1}}\rfloor^{0}$
$\boldsymbol{\nu}$
$\boldsymbol{u} = D^{-1}(\theta)\{\boldsymbol{\nu} + \dot{\boldsymbol{p}}_{\boldsymbol{d}}\}$
θ
$\boldsymbol{\rho}$
DDR
$\boldsymbol{Y}$
$\boldsymbol{p}$
θ

Figure 3.2: Relative degree 1: Block diagram

3.3.3 Zero dynamics

As mentioned earlier for Equation (3.1), controlling x and y tracking error forces to leave θ uncontrolled. Therefore, zero dynamics ([Khalil, 2002, p. 516]) for System (3.12) corresponds to θ dynamics and it is given by

$$\begin{aligned}\dot{\theta} &= \boldsymbol{\phi}^{\top}\boldsymbol{u} + \boldsymbol{\phi}^{\top}\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix} \\ &= \boldsymbol{\phi}^{\top}\left\{D^{-1}(\theta)\left[-K_1\lceil \boldsymbol{e}_{\boldsymbol{1}}\rfloor^{1/2} + \boldsymbol{\eta}\right] + \dot{\boldsymbol{p}}_{\boldsymbol{d}}\right\} + \boldsymbol{\phi}^{\top}\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix} \\ ,\dot{\boldsymbol{\eta}} &= -K_2\lceil \boldsymbol{e}_{\boldsymbol{1}}\rfloor^{0}\end{aligned} \tag{3.20}$$

Equation (3.20) shows that state θ is affected by exogenous signals. During transient response little can be said about θ dynamics, however, depending on the desired trajectory some observations can be made about θ dynamics when the second order sliding mode Equation (3.19) is reached. From Equation (3.19), control law $\boldsymbol{\nu}$ in Equation (3.16) becomes

$$\boldsymbol{\nu} = -\begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix} = -D(\theta)\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix}. \tag{3.21}$$

Using Equation (3.21) one can obtain $\boldsymbol{u}$ from Equation (3.15) as

$$\boldsymbol{u} = D^{-1}(\theta)\left\{-D(\theta)\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix} + \dot{\boldsymbol{p}}_{\boldsymbol{d}}\right\} = -\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix} + D^{-1}(\theta)\dot{\boldsymbol{p}}_{\boldsymbol{d}}. \tag{3.22}$$

and substitute Equation (3.22) in Equation (3.20). After doing some computations, zero dynamics yields

$$\dot{\theta} = \boldsymbol{\phi}^{\top} D^{-1}(\theta)\dot{\boldsymbol{p}}_{\boldsymbol{d}} = \begin{bmatrix}-\frac{\sin(\theta)}{h}, & +\frac{\cos(\theta)}{h}\end{bmatrix}\dot{\boldsymbol{p}}_{\boldsymbol{d}}. \tag{3.23}$$

Rewritting Equation (3.23) yields

$$-\sin(\theta)\dot{x}_d + \cos(\theta)\dot{y}_d - h\dot{\theta} = 0, \tag{3.24}$$

which is in the form of a NHC. Phisically, point $\boldsymbol{p}$ dynamics is fixed w.r.t. to point $\boldsymbol{a}$ dynamics by the introduced parameter h, hence, the NHC is present in a different form for both dynamics. As mentioned in Introduction chapter, the general form of this equation is not integrable. The following paragraphs study the evolution of θ given specific trajectories, such as, lines and circles because a complex trajectory can be designed by combining these two motions.

Regulation

The case where

$$\begin{aligned} x_d(t) &= b_x \\ y_d(t) &= b_y \end{aligned} \Rightarrow \begin{aligned} \dot{x}_d(t) &= 0 \\ \dot{y}_d(t) &= 0 \end{aligned} \Rightarrow \begin{aligned} \ddot{x}_d(t) &= 0 \\ \ddot{y}_d(t) &= 0 \end{aligned} \tag{3.25}$$

where $b_x, b_y \in \mathbb{R}$. This case represents the motion when only a goal point is defined and the transition towards that point is not defined. The NHC from Equation (3.24) becomes

$$h\dot{\theta} = -\sin(\theta)\cancelto{0}{\dot{x}_d} + \cos(\theta)\cancelto{0}{\dot{y}_d} \Rightarrow \dot{\theta} = 0 \Rightarrow \theta(t) = c_1, \tag{3.26}$$

where $c_1 \in \mathbb{R}$ which states that when the goal point is reached, the DDR will stop rotating.

Tracking

The case when the DDR tracks straight lines where

$$\begin{aligned} x_d(t) &= m_x t + b_x \\ y_d(t) &= m_y t + b_y \end{aligned} \Rightarrow \begin{aligned} \dot{x}_d(t) &= m_x \\ \dot{y}_d(t) &= m_y \end{aligned} \Rightarrow \begin{aligned} \ddot{x}_d(t) &= 0 \\ \ddot{y}_d(t) &= 0 \end{aligned} \tag{3.27}$$

where $m_x, m_y, b_x, b_y \in \mathbb{R}$ and $t \in \mathbb{R}_+$. The NHC from Equation (3.24) becomes

$$h\dot{\theta} = -\sin(\theta)m_x + \cos(\theta)m_y \Rightarrow \theta(t) = 2\text{atan}\left(\frac{c_2\tanh\left(\frac{c_2}{2}\left(\frac{t}{h} + c_3\right)\right) - m_x}{m_y}\right), \tag{3.28}$$

where $c_3 \in \mathbb{R}$ and $c_2 = \sqrt{m_x^2 + m_y^2}$.

General case

Rewritting NHC from Equation (3.24) as

$$h\dot{\theta} = -\sin(\theta)\dot{x}_d + \cos(\theta)\dot{y}_d. \tag{3.29}$$

If point $\boldsymbol{p}$ is point $\boldsymbol{a}$, then, $h = 0$ and Equation (3.29) can be rewritten as

$$-\sin(\theta_d)\dot{x}_d + \cos(\theta_d)\dot{y}_d = 0, \tag{3.30}$$

from which solving for θ_d yields

$$\theta_d = \operatorname{atan}\left(\frac{\dot{y}_d}{\dot{x}_d}\right), \tag{3.31}$$

which implies

$$\frac{\sin(\theta_d)}{\cos(\theta_d)} = \frac{\dot{y}_d}{\dot{x}_d}, \tag{3.32}$$

and, consequently,

$$\dot{x}_d = \frac{\cos(\theta_d)}{\sin(\theta_d)}\dot{y}_d, \quad \dot{y}_d = \frac{\sin(\theta_d)}{\cos(\theta_d)}\dot{x}_d. \tag{3.33}$$

Substituting Equation (3.33) in Equation (3.29) yields

$$h\dot{\theta} = -\sin(\theta)\cos(\theta_d)\left(\frac{\dot{y}_d}{\sin(\theta_d)}\right) + \cos(\theta)\sin(\theta_d)\left(\frac{\dot{x}_d}{\cos(\theta_d)}\right). \tag{3.34}$$

Additionaly, Equation (3.32) also implies

$$\frac{\dot{x}_d}{\cos(\theta_d)} = \frac{\dot{y}_d}{\sin(\theta_d)} = f_d(t). \tag{3.35}$$

Substituting Equation (3.35) in Equation (3.34) yields

$$\begin{aligned} h\dot{\theta} &= -f_d(t)\sin(\theta)\cos(\theta_d) + f_d(t)\sin(\theta_d)\cos(\theta), \\ h\dot{\theta} &= -f_d(t)\left(\sin(\theta - \theta_d)\right). \end{aligned} \tag{3.36}$$

Now, let

$$\begin{aligned} \varepsilon &= \theta - \theta_d, \\ \dot{\varepsilon} &= \dot{\theta} - \dot{\theta}_d. \end{aligned} \tag{3.37}$$

Taking into account Equation (3.37), Equation (3.36) can be rewritten as

$$h(\dot{\theta} - \dot{\theta}_d) = -f_d(t)\left(\sin(\theta - \theta_d)\right) - h\dot{\theta}_d. \tag{3.38}$$

Then, substituting Equation (3.37) in Equation (3.38) yields

$$h\dot{\varepsilon} = -f_d(t)\sin(\varepsilon) - h\dot{\theta}_d. \tag{3.39}$$

Assume

$$\begin{aligned}\varepsilon =& h\varepsilon_1 + h^2\varepsilon_2 + \dots,\\ \dot{\varepsilon} =& h\dot{\varepsilon}_1 + h^2\dot{\varepsilon}_2 + \dots\end{aligned} \tag{3.40}$$

Substituting Equation (3.40) in Equation (3.39) and taking into account $\sin(\varepsilon) \approx \varepsilon$ for sufficiently small ε yields

$$\begin{aligned}h\left(h\dot{\varepsilon}_1 + h^2\dot{\varepsilon}_2\right) \approx& - f_d(t)\left(h\varepsilon_1 + h^2\varepsilon_2\right) - h\dot{\theta}_d + \dots\\ h^2\dot{\varepsilon}_1 + h^3\dot{\varepsilon}_2 \approx& - hf_d(t)\varepsilon_1 - h^2 f_d(t)\varepsilon_2 - h\dot{\theta}_d + \dots\end{aligned} \tag{3.41}$$

It is reasonable to assume that distance h is a small quantity because h is usually located on the robot chasis and that distance in meters makes h a small parameter. If $0 < h < 1$ then $h^k \to 0$ with $k \geq 3$ and Equation (3.41) can be rewritten as

$$\dot{\varepsilon}_1 h^2 \approx - \left(f_d(t)\varepsilon_1 + \dot{\theta}_d\right) h - f_d(t)\varepsilon_2 h^2. \tag{3.42}$$

Now, in order for the left side to approximate the right side of Equation (3.42) the following equations must be satisfied

$$\begin{aligned}0 =& - \left(f_d(t)\varepsilon_1 + \dot{\theta}_d\right),\\ \dot{\varepsilon}_1 =& - f_d(t)\varepsilon_2.\end{aligned} \Rightarrow \begin{aligned}\varepsilon_1 =& - \frac{\dot{\theta}_d}{f_d(t)} \Rightarrow \dot{\varepsilon}_1 = \frac{-f_d(t)\ddot{\theta}_d + \dot{f}_d(t)\dot{\theta}_d}{f_d^2(t)},\\ \varepsilon_2 =& - \frac{\dot{\varepsilon}_1}{f_d(t)}.\end{aligned} \tag{3.43}$$

From Equation (3.43), ε_1 and ε_2 can be written in terms of known functions $f_d(t)$ and θ_d and their respective derivatives as

$$\varepsilon_1 = - \frac{\dot{\theta}_d}{f_d(t)}, \qquad \varepsilon_2 = \frac{f_d(t)\ddot{\theta}_d - \dot{f}_d(t)\dot{\theta}_d}{f_d^3(t)}. \tag{3.44}$$

Finally, from Equation (3.37) and Equation (3.40), an approximation of θ can be obtained as

$$\begin{aligned}\theta =& \varepsilon + \theta_d,\\ \theta =& h\left(-\frac{\dot{\theta}_d}{f_d(t)}\right) + h^2\left(\frac{f_d(t)\ddot{\theta}_d - \dot{f}_d(t)\dot{\theta}_d}{f_d^3(t)}\right) + \theta_d + \dots\end{aligned} \tag{3.45}$$

Figure 3.3 shows the comparation between θ_d, the theoretical tangent function from point $\boldsymbol{a}$, the measured output θ and its approximation using the correction terms

$$\begin{aligned}\theta_1 \approx& h\varepsilon_1 + \theta_d,\\ \theta_2 \approx& h\varepsilon_1 + h^2\varepsilon_2 + \theta_d,\end{aligned} \tag{3.46}$$

respectively, when tracking a circular trajectory of radius R. As $h \to 0$ signal θ and its approximations θ_1 and θ_2 tend to be θ_d. However, as h tends to R signal θ deviates from θ_d and only the first correction term θ_1 represents a decent approximation of θ.

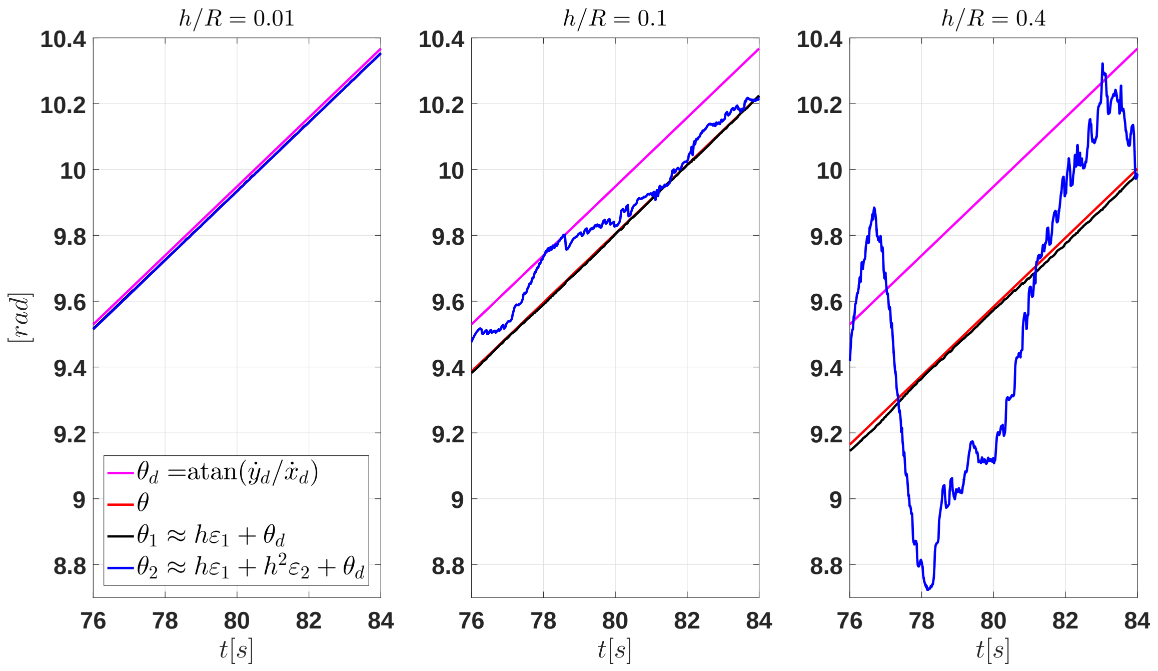


Figure 3.3: Comparison of ε and correction terms

Given that the reference trajectory is a circle of radius R, another aspect to take into account is the relation between R and h. There are three cases for that relation which are illustrated in Figure 3.4.

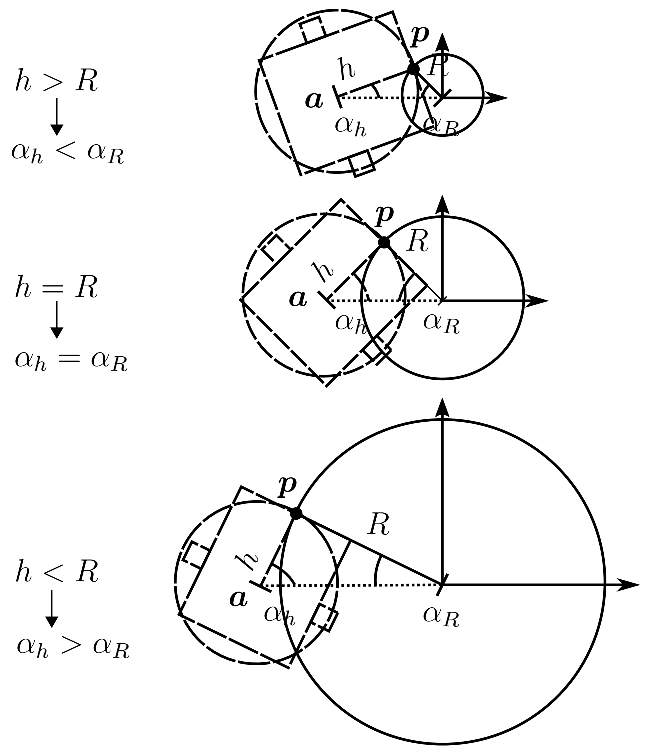


Figure 3.4: Relation h/R

From sine law

$$\frac{h}{\alpha_R} = \frac{R}{\alpha_h} \Rightarrow \alpha_R = \left(\frac{h}{R}\right)\alpha_h. \tag{3.47}$$

Then, the following cases need to be analyzed

- $h > R \Leftrightarrow h/R > 1 \Leftrightarrow \alpha_R > \alpha_h$: This ressembles the behavior of a rod fixed from one end and the other end connected to a point in the circle with radius R in the sense that the DDR has no liberty to maneuver. Tracking position can be achieved but orientation evolution will be far from the tangent behavior.

- $h = R \Leftrightarrow h/R = 1 \Leftrightarrow \alpha_h = \alpha_R$: The only way this tracking could happen is if h is small enough to allow the DDR to rotate without making a displacement in the XY coordinates.

- $h < R \Leftrightarrow h/R < 1 \Leftrightarrow \alpha_R < \alpha_h$: The DDR can maneuver along the trajectory.

In order to allow the DDR to maneuver across the trajectory space only the case $h < R$ is considered for the following simulations and experimental tests.

3.4 Simulations

The parameters of the robot are $h = 0.1$m, $r = 0.05$m and $d = 0.272$m. In order to show STA strengths, this control law is compared with two other control algorithms. First, a linear PID control law (LIN) which has the following structure

$$\begin{aligned} \boldsymbol{\nu} =& -K_1\boldsymbol{e_1} - K_2\boldsymbol{\eta} - K_3\dot{\boldsymbol{e}}_{\boldsymbol{1}} \\ \dot{\boldsymbol{\eta}} =& \boldsymbol{e_1} \end{aligned} \tag{3.48}$$

where $K_1 = \text{diag}(k_{1x}, k_{1y})$, $K_2 = \text{diag}(k_{2x}, k_{2y})$ and $K_3 = \text{diag}(k_{3x}, k_{3y})$.

Second, a first order sliding mode algorithm (SGN) which has the following structure

$$\boldsymbol{\nu} = -K_1 \lceil \boldsymbol{e_1} \rfloor^0 \tag{3.49}$$

where $K_1 = \text{diag}(k_{1x}, k_{1y})$ and

$$\lceil \boldsymbol{e_1} \rfloor^0 = \begin{bmatrix} \text{sign}(x - x_d) \\ \text{sign}(y - y_d) \end{bmatrix} \in \mathbb{R}^2.$$

The wheels position in the XY plane can be found from geometry by using an homogeneous transformation. Let $R(x_s, y_s, \phi)$ define a translation and rotation transformation as

$$R(x_s, y_s, \phi) = \begin{bmatrix} +\cos(\phi) & -\sin(\phi) & x_s \\ +\sin(\phi) & +\cos(\phi) & y_s \\ 0 & 0 & 1 \end{bmatrix}.$$

W.r.t to the global frame $X_G Y_G$, the position of the right wheel (RW) can be found as

$$\begin{bmatrix} x_{RW} \\ y_{RW} \\ 1 \end{bmatrix} = R(x_a, y_a, \theta) \begin{bmatrix} 0 \\ +\frac{d}{2} \\ 1 \end{bmatrix}.$$

Analougously, the position of the left wheel (LW) can be found as

$$\begin{bmatrix} x_{LW} \\ y_{LW} \\ 1 \end{bmatrix} = R(x_a, y_a, \theta) \begin{bmatrix} 0 \\ -\frac{d}{2} \\ 1 \end{bmatrix}.$$

Regulation

The reference trajectory is a goal point described by Equation (3.25) with $b_x = +0.7$ m and $b_y = -0.35$ m and initial conditions $x_0 = -0.7$ m and $y_0 = +0.7$ m and $\theta_0 = 3\pi/4$. Figure 3.5 shows the DDR's motion in the XY plane using the control signals shown in Figure 3.8. Evolution in time of controlled position outputs is shown in Figure 3.6 and the measured orientation output is shown in Figure 3.7.

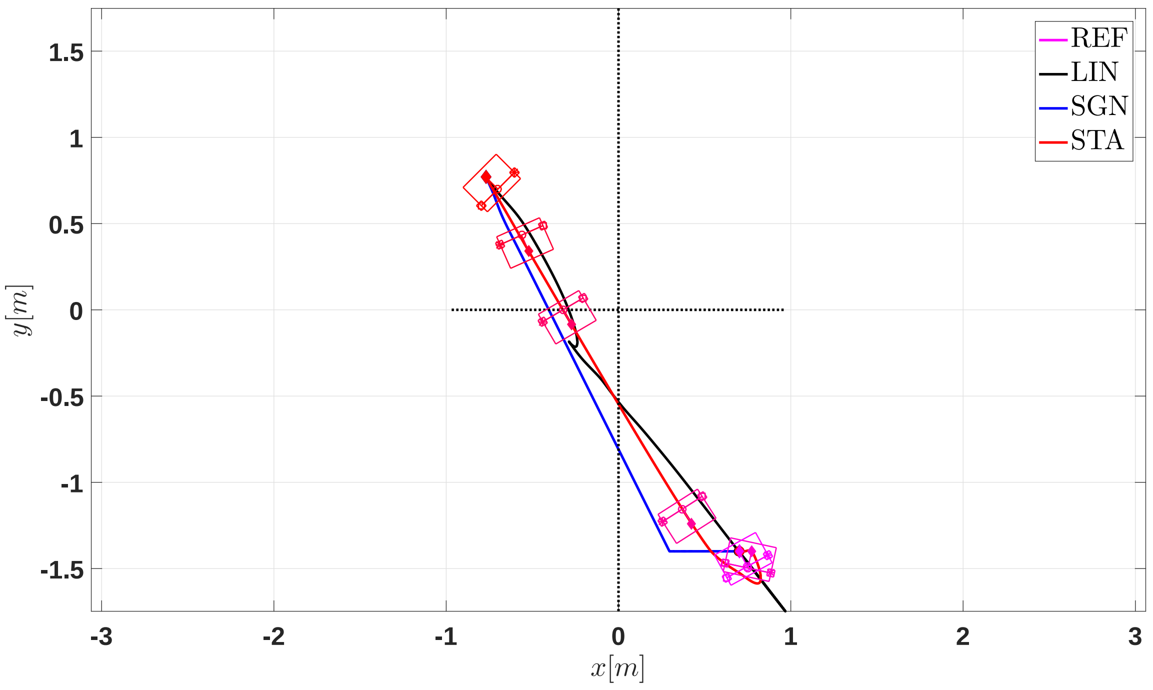


Figure 3.5: Simulation: Regulation. XY plane. Controlled output signals

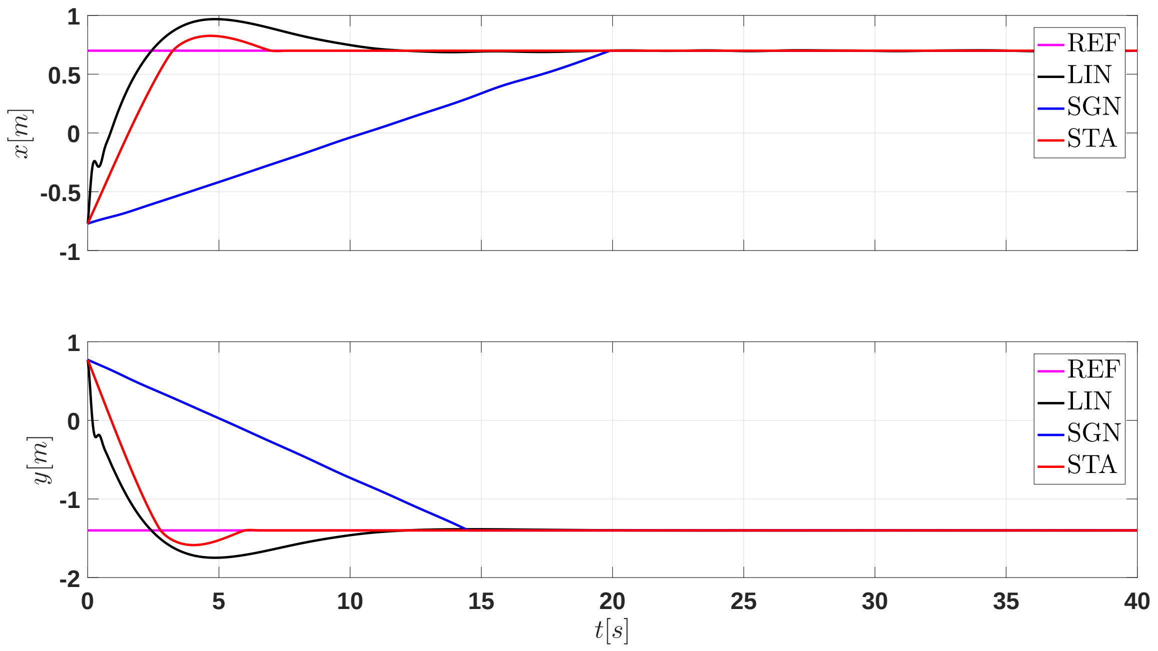


Figure 3.6: Simulation: Regulation. Controlled output signals

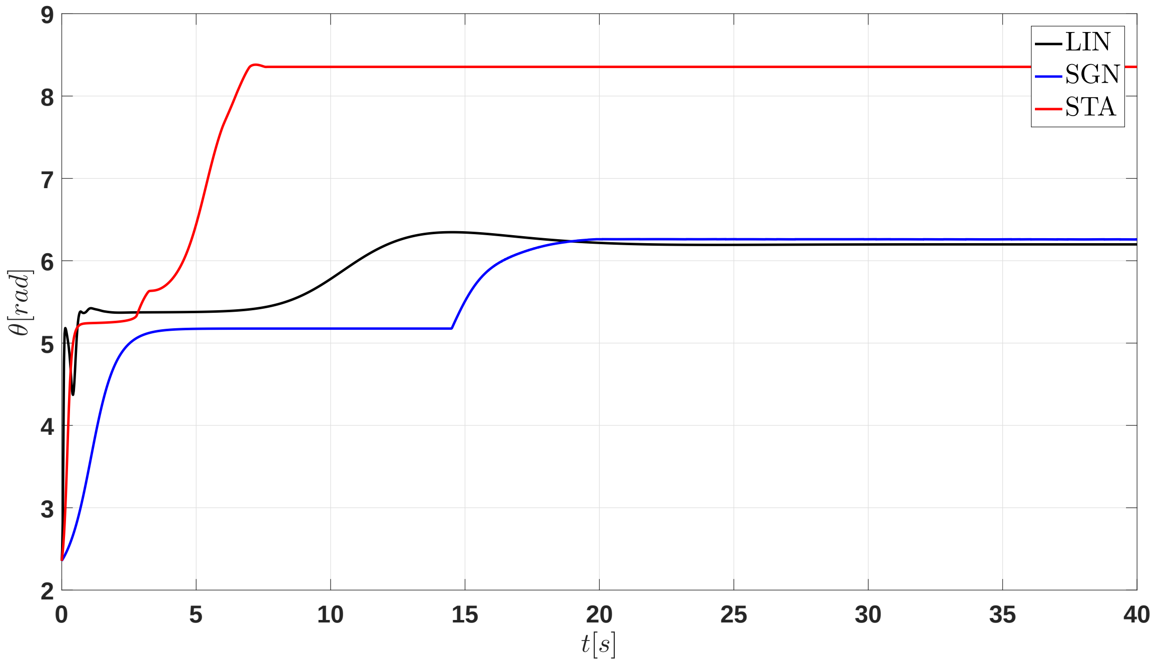


Figure 3.7: Simulation: Regulation. Measured output signal

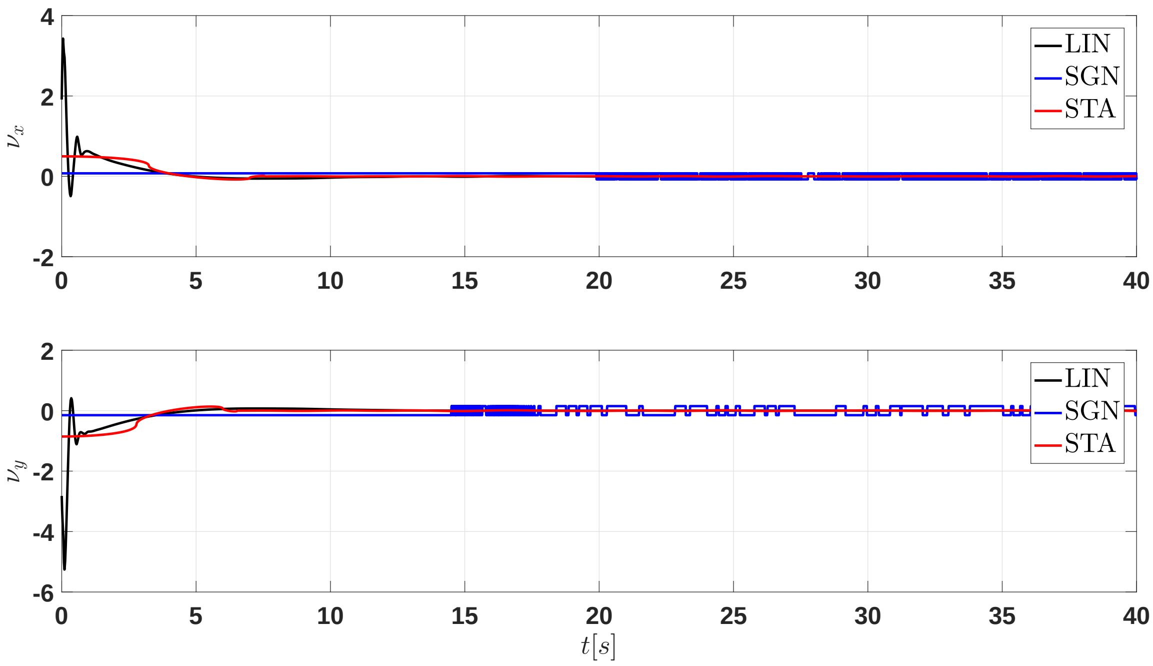


Figure 3.8: Simulation: Regulation. Control signals

Generated error signals are shown in Figure 3.9. A close-up to these signals after convergence is shown in Figure 3.10which shows that the linear algorithm cannot compensate exactly the proposed disturbances and that both sliding mode controllers do.

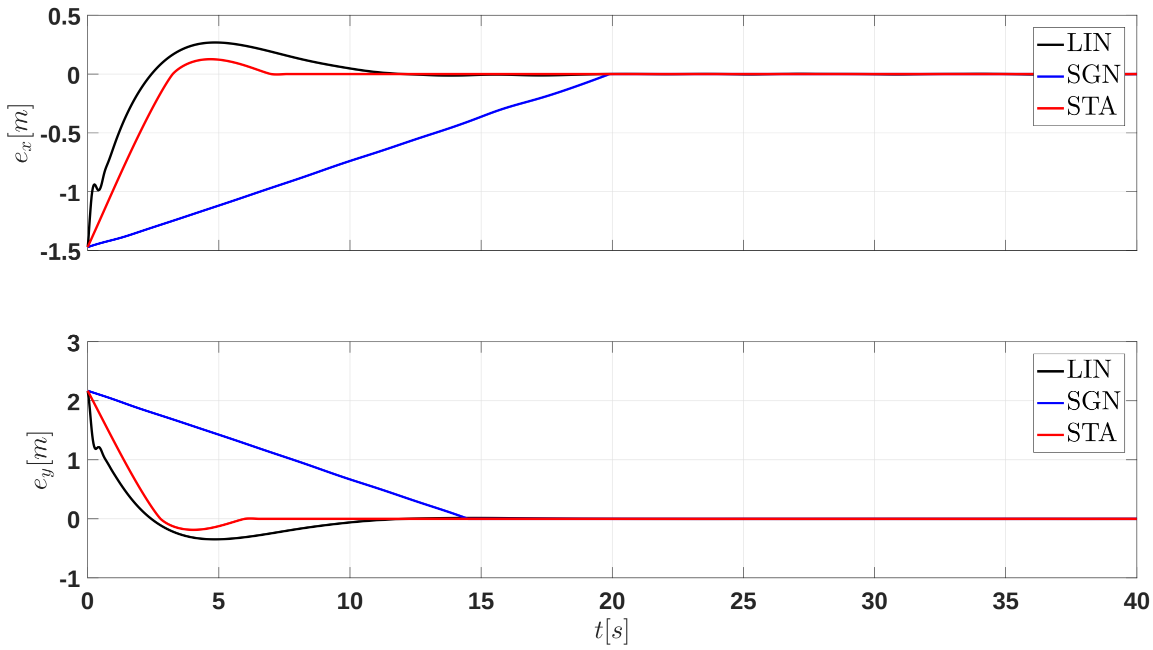


Figure 3.9: Simulation: Regulation. Error signals

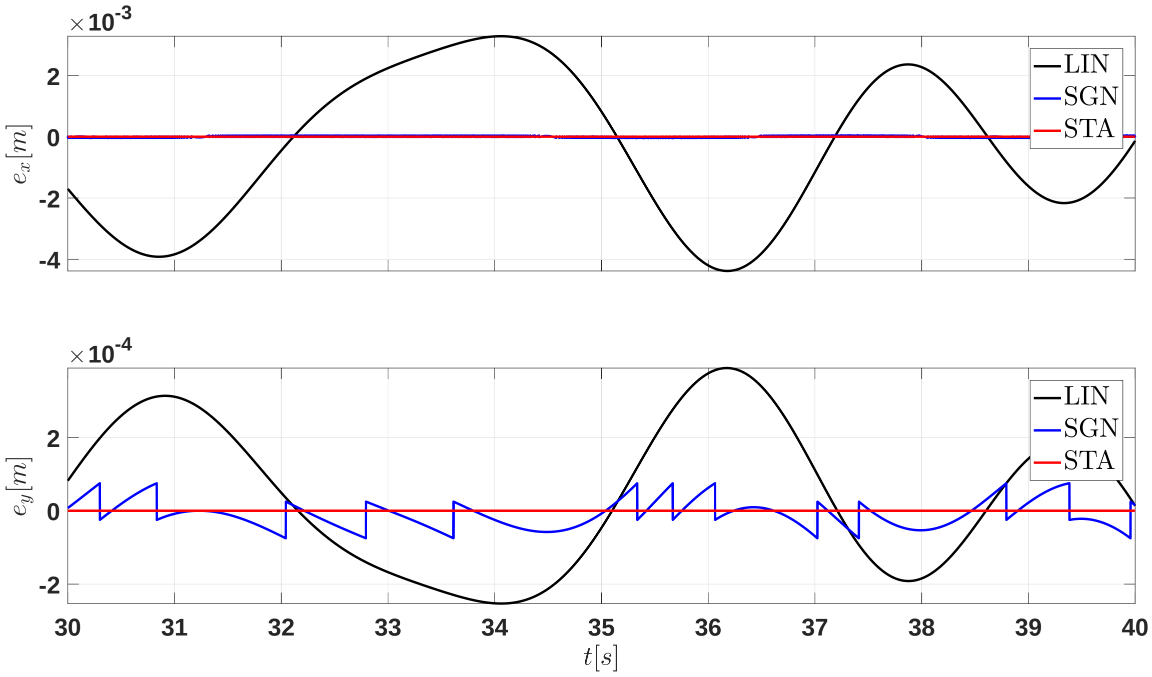
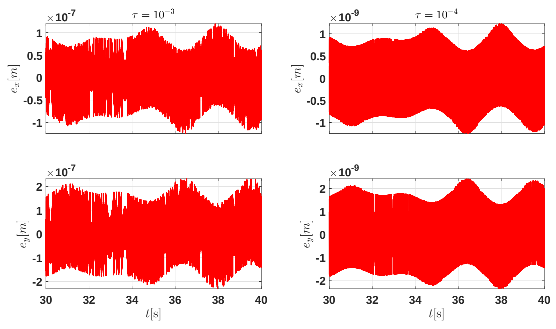
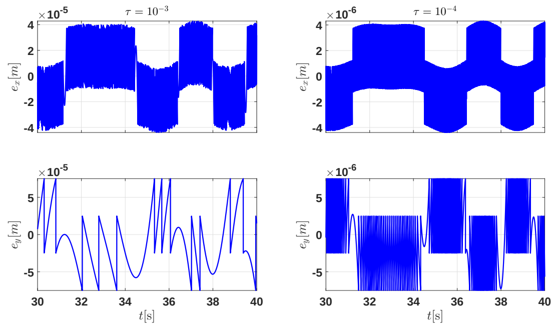


Figure 3.10: Simulation: Regulation. Error signals (After convergence)

However, SGN only guarantees a first-order sliding mode and STA guarantees a second-order sliding mode of the sliding variable. Since STA reaches a higher order sliding mode than SGN, STA can guarantee a better error precision. In order to show that the STA and SGN are actually reaching their respective sliding mode (SM), e_x and e_y are analyzed performing the same simulation but for sampling step $\tau = 10^{-3}$ and $\tau = 10^{-4}$. For STA, Figure 3.11a shows that the SM is present because the precision of the algorithm is increased by decreasing the sampling step (Levant [1998]; Moreno [2009]). The same result is obtained for SGN as shown in Figure 3.11b. In both sampling step cases STA has the best error precision compared to LIN and SGN performances.



(a) STA



(b) SGN

Figure 3.11: Simulation: Regulation. Comparison of precision.

Straight line

The reference trajectory is a straight line described by Equation (3.25) with $m_x = 0.01$, $m_y = 0.01$, $b_x = +0.7$m and $b_y = +0.7$m and initial conditions $x_0 = -0.7$m, $y_0 = -0.7$m and $\theta_0 = 5\pi/4$. Figure 3.12 shows the DDR's motion in the XY plane using the control signals shown in Figure 3.15. Evolution in time of controlled position outputs is shown in Figure 3.13 and the remaining measured orientation output is shown in Figure 3.14.

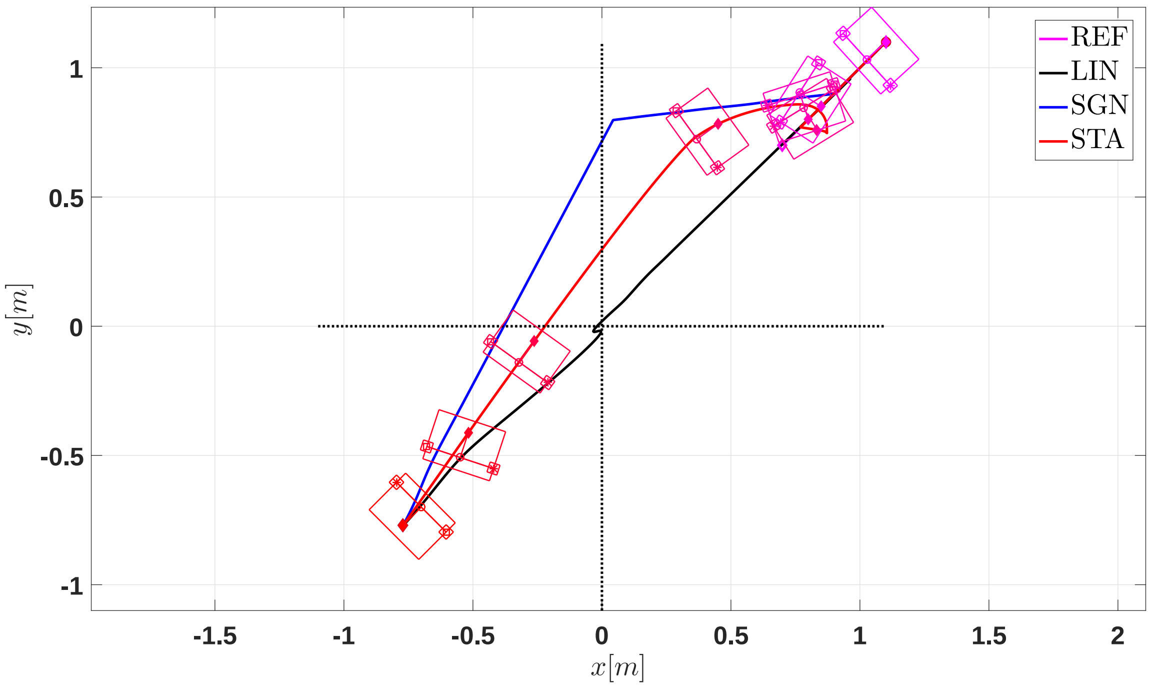


Figure 3.12: Simulation: Straight line. XY plane. Controlled output signals

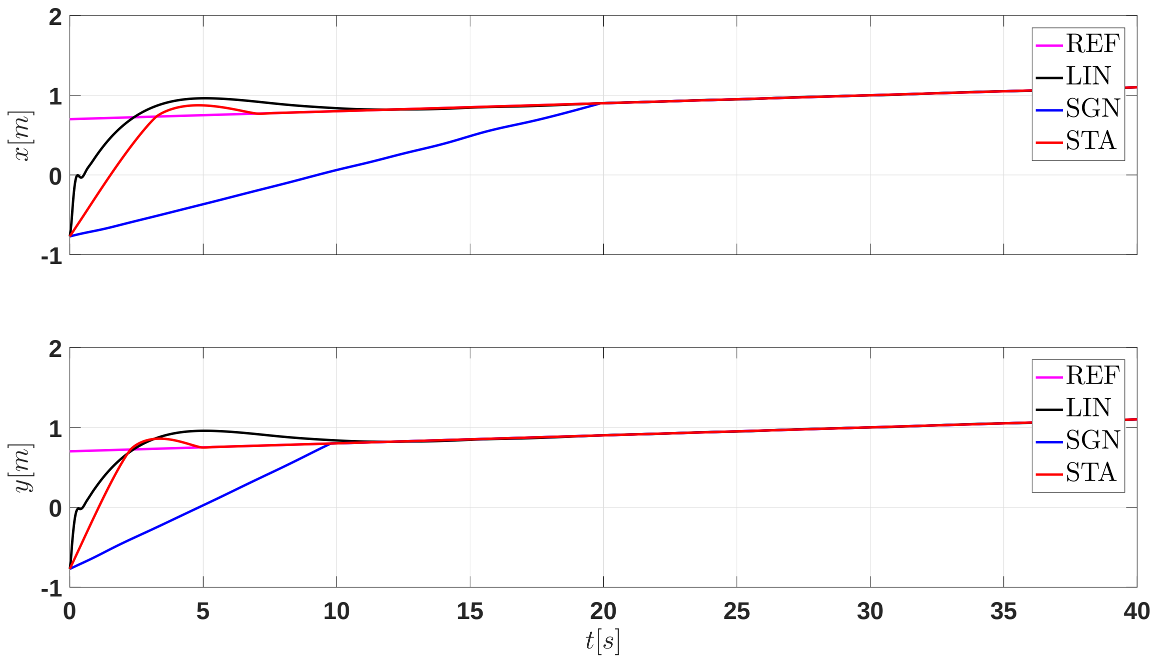


Figure 3.13: Simulation: Straight line. Controlled output signals

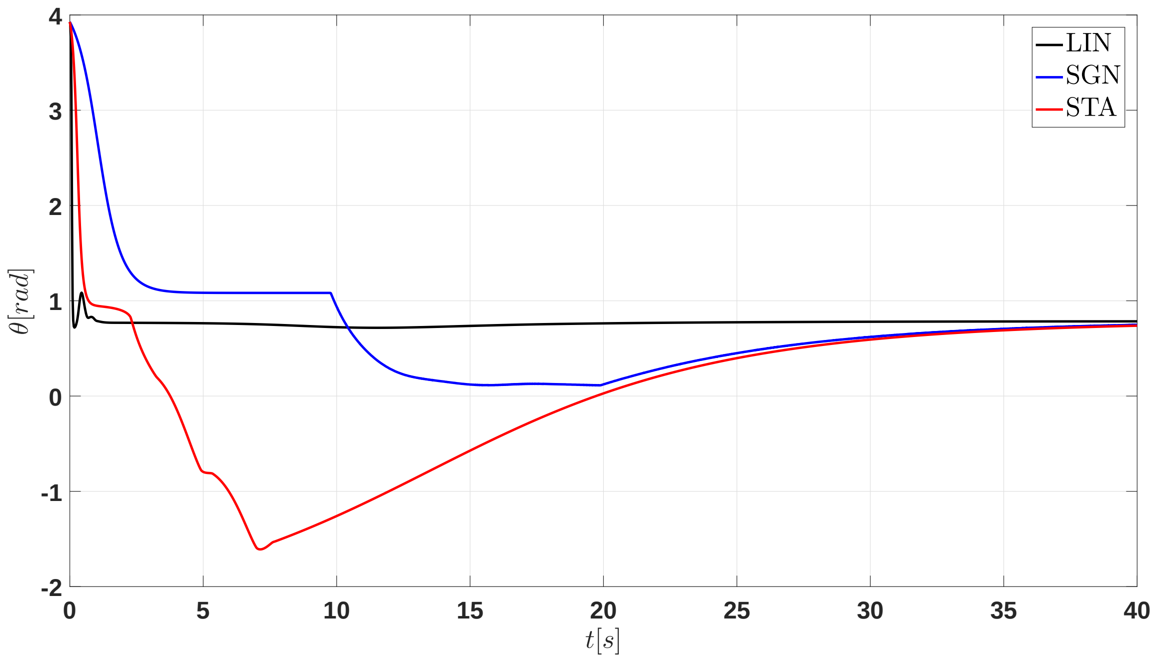


Figure 3.14: Simulation: Straight line. Measured output signal

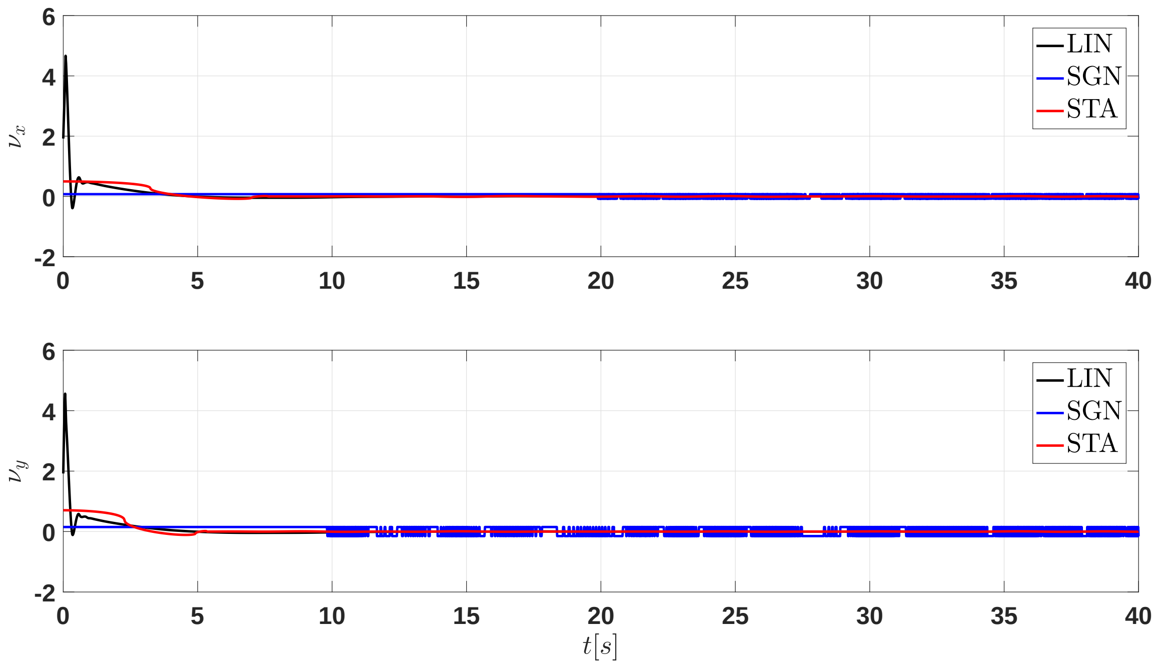


Figure 3.15: Simulation: Straight line. Control signals

Generated error signals are shown in Figure 3.16. A close-up to these signals after convergence is shown in Figure 3.17 which shows, as occurs in regulation too, that the linear algorithm cannot compensate exactly the proposed disturbances and that both sliding mode controllers do. Furthermore, the STA accomplishes the control goal with a better precision than SGN, as shown in Figure 3.18.

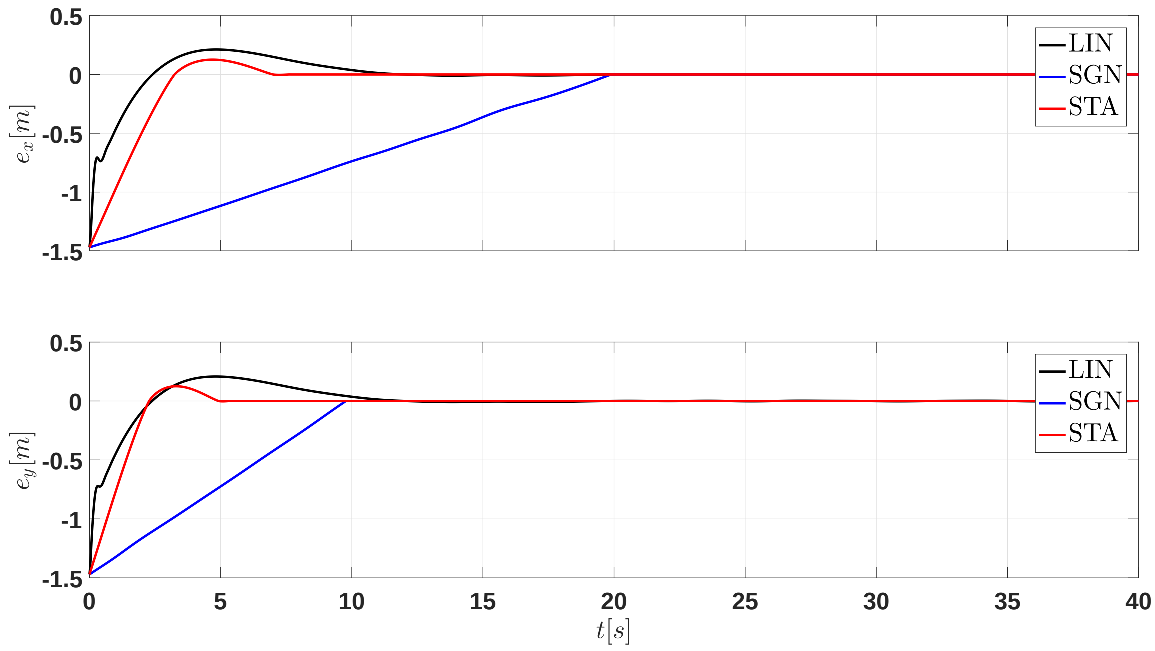


Figure 3.16: Simulation: Straight line. Error signals

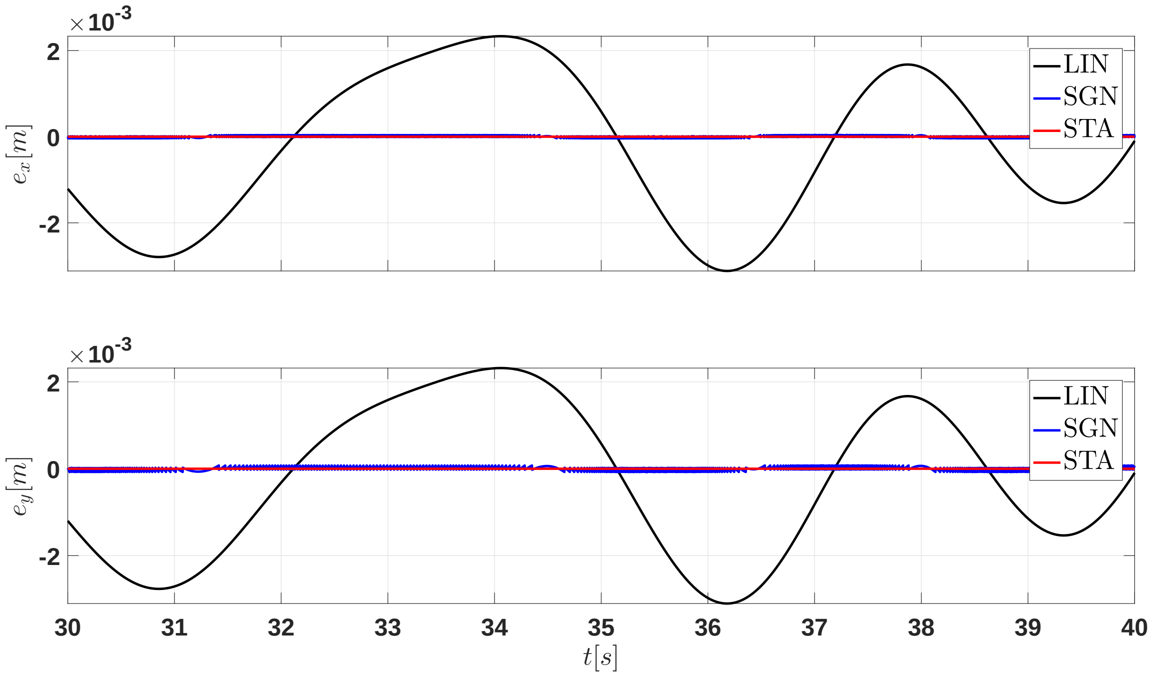
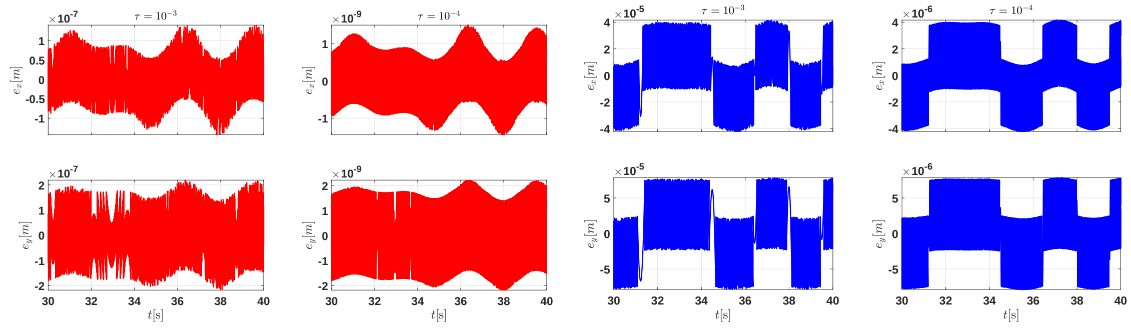


Figure 3.17: Simulation: Straight line. Error signals (After convergence)



(a) STA **(b) SGN**

Figure 3.18: Simulation: Straight line. Comparison of precision.

Tracking

The reference trajectory is a circle with radius $R = 0.7$m and center $(h, k) = (0, 0)$m described by the following

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} R\cos(\omega t) + h \\ R\sin(\omega t) + k \end{bmatrix} \text{m} \quad , \quad \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix} = \begin{bmatrix} -R\omega\sin(\omega t) \\ +R\omega\cos(\omega t) \end{bmatrix} \text{m/s} \tag{3.50}$$

where $\omega = 2\pi/T$ with $T = 60$s. Initial conditions are set to

$$\begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix} = \begin{bmatrix} -0.7\text{m} \\ +0.7\text{m} \\ 0\text{rad} \end{bmatrix}$$

Figure 3.19 shows the DDR's motion in the XY plane using the control signals shown in Figure 3.22. Evolution in time of controlled position outputs is shown in Figure 3.20 and the measured orientation output is shown in Figure 3.21.

Generated error signals are shown in Figure 3.23. A close-up to these signals after convergence is shown in Figure 3.24. For LIN, e_x and e_y are bounded because linear algorithms cannot compensate exactly matched disturbances and cannot take the state to zero in finite time; this is shown in Figure 3.24.

However, SGN only guarantees a first-order sliding mode and STA guarantees a second-order sliding mode of the sliding variable. This reflects on the precision that the algorithm can achieve. In order to show that the STA and SGN are actually reaching the sliding mode (SM), e_x and e_y are analyzed performing the same simulation but for sampling step $\tau = 10^{-3}$ and $\tau = 10^{-4}$. For STA, Figure 3.25a shows that the SM is present because the precision of the algorithm is increased by decreasing the sampling step (Levant [1998]; Moreno [2009]). The same result is obtained for SGN as shown in Figure 3.25b. Summarizing in Table 3.1, in both sampling step cases STA has the best error precision compared to LIN and SGN performances.

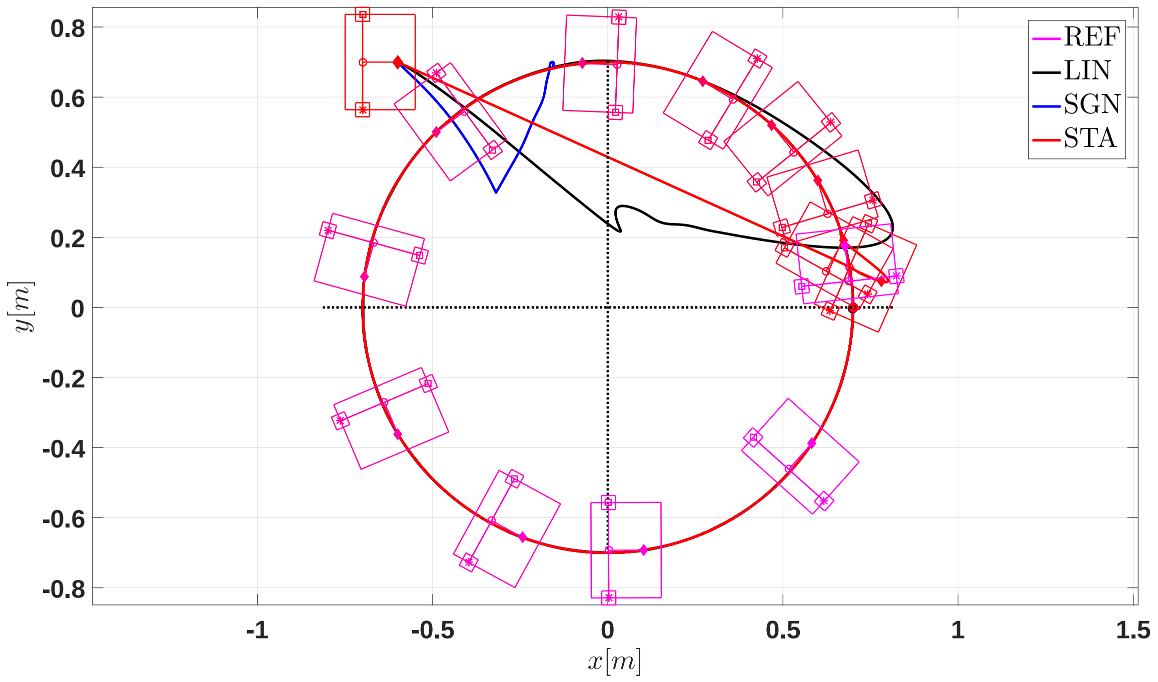


Figure 3.19: Simulation: Tracking. XY plane. Controlled output signals

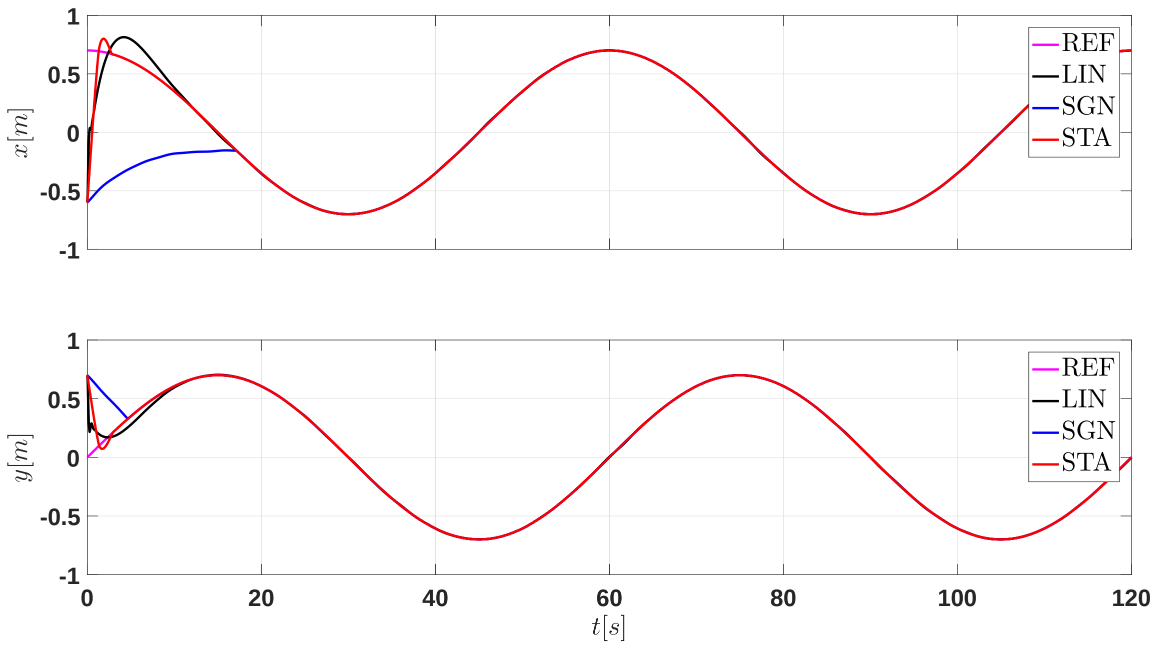


Figure 3.20: Simulation: Tracking. Controlled output signals

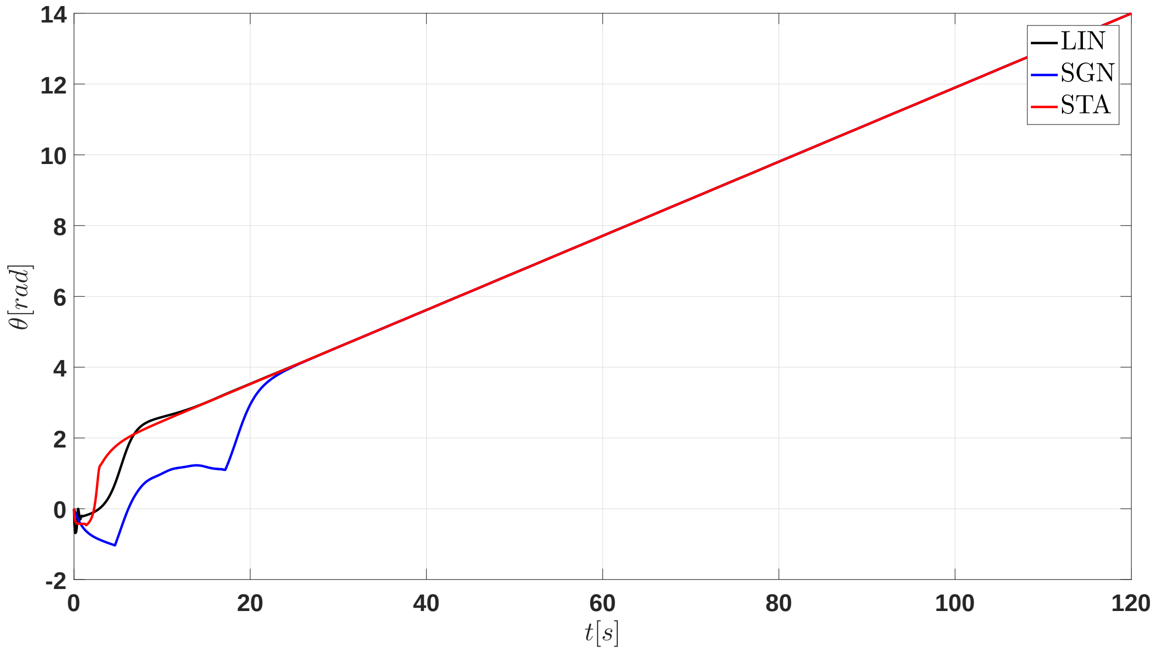


Figure 3.21: Simulation: Tracking. Measured output signal

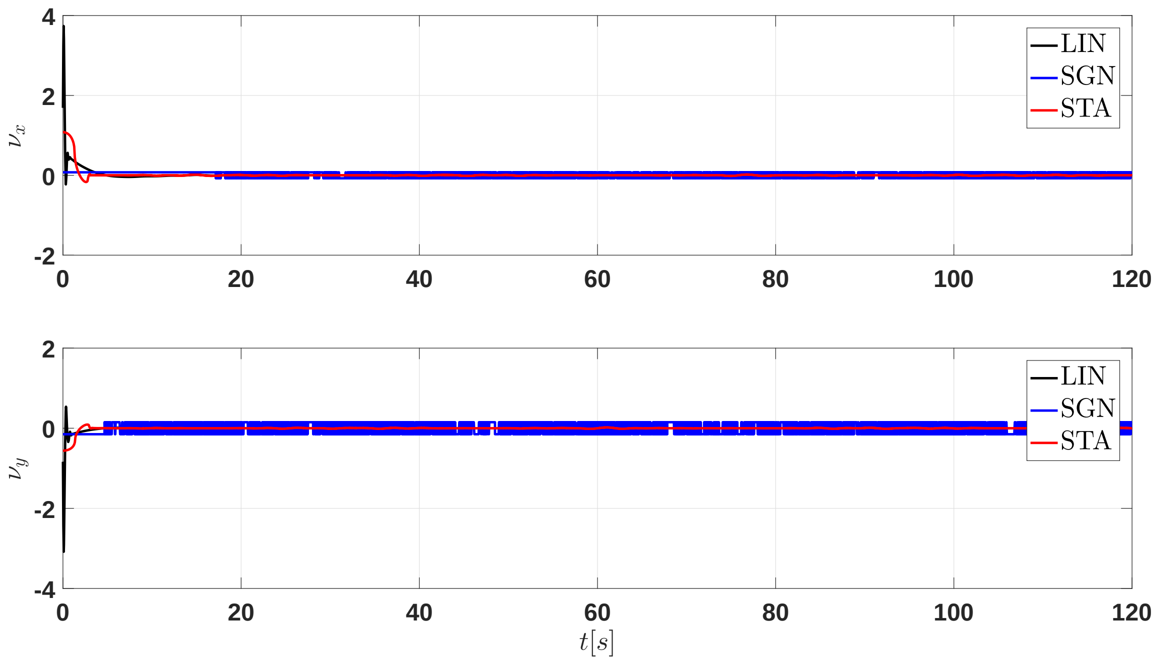


Figure 3.22: Simulation: Tracking. Control signals

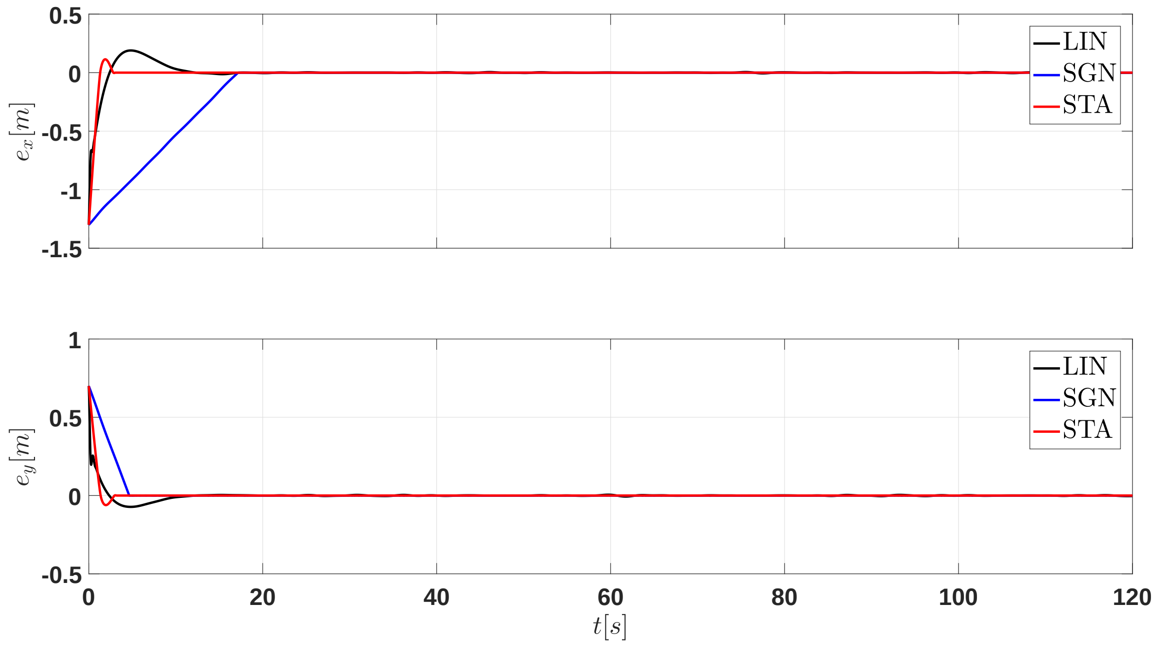


Figure 3.23: Simulation: Tracking. Error signals

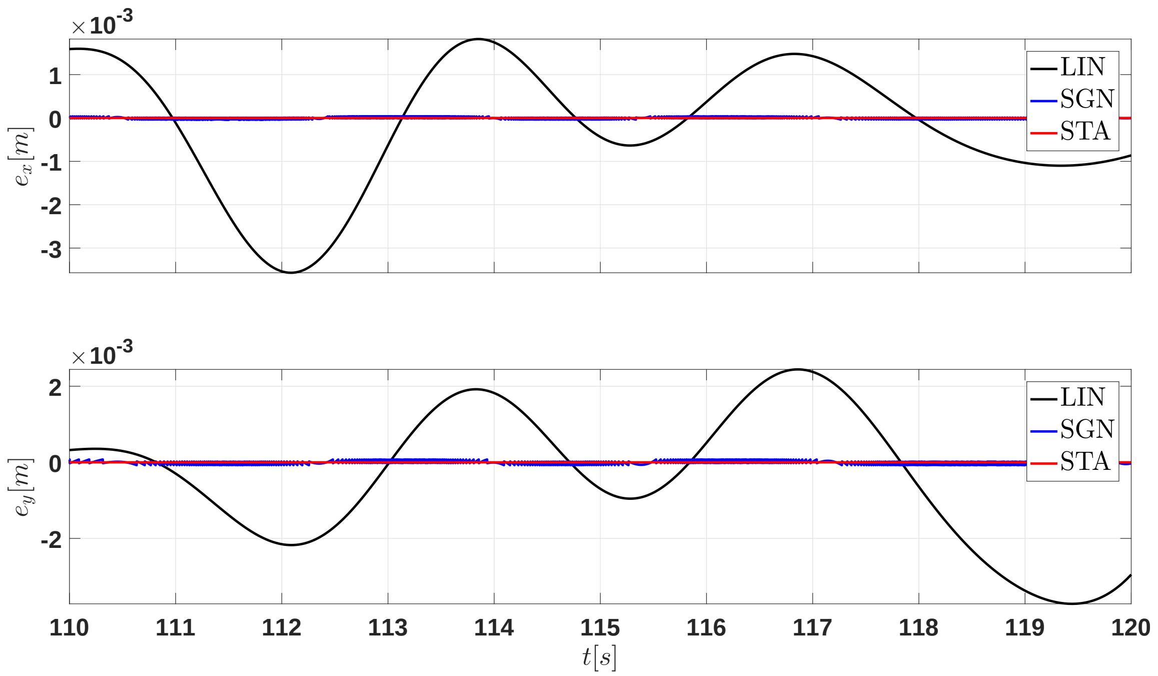
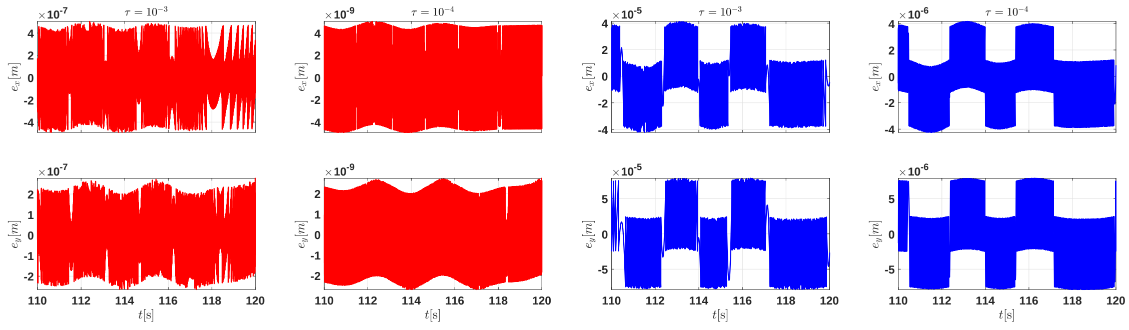


Figure 3.24: Simulation: Tracking. Error signals (After convergence)



(a) STA (b) SGN

Figure 3.25: Simulation: Tracking. Comparison of precision.

Table 3.1: Simulation: Tracking. Precision Table

	$\tau = 1 \times 10^{-3}$	$\tau = 1 \times 10^{-4}$
STA	$\tau^2 = 1 \times 10^{-6}$	$\tau^2 = 1 \times 10^{-8}$
SGN	$\tau^1 = 1 \times 10^{-3}$	$\tau^1 = 1 \times 10^{-4}$

3.5 Experimental Tests

The position and orientation signals $\boldsymbol{a}_{\text{raw}}$ and θ_{raw} received to perform the experimental tests have quantization and discretization phenomena. Computing $\boldsymbol{p}$ with $\boldsymbol{a}_{\text{raw}}$ and θ_{raw} inherits these phenomena for coordinate x signal, for simplicity, as shown in Figure 3.26. Particularly, these phenomena can cause unexpected behavior in the controller. For this reason there is a need to filter $\boldsymbol{a}_{\text{raw}}$ and θ_{raw} in order to obtain smoother signals $\boldsymbol{a}$ and θ and, consequently, $\boldsymbol{p}$ signal.

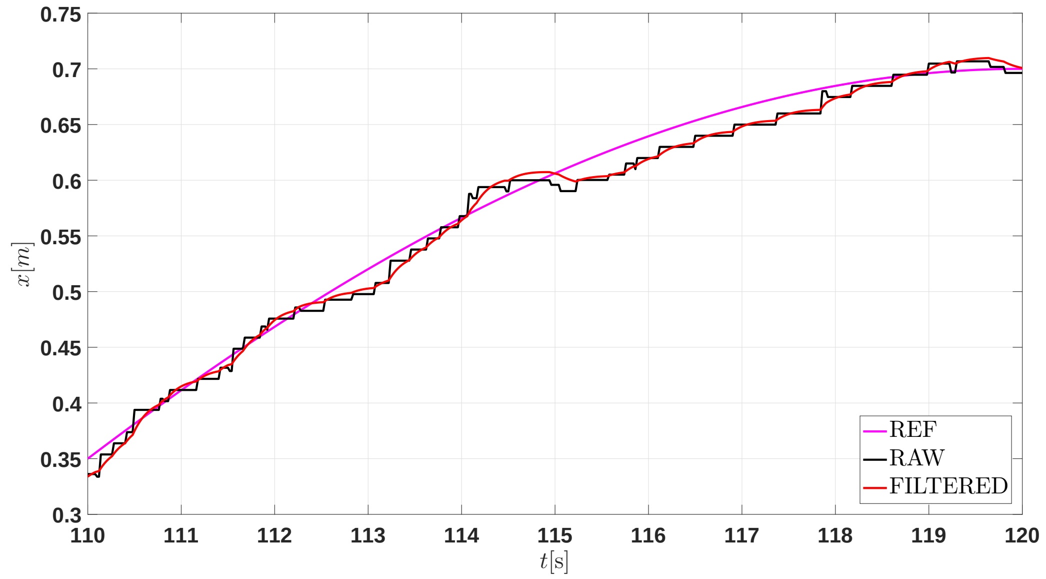


Figure 3.26: Quantization and discretization phenomena

Filtering

Filtering can be done with many methods. In this thesis, two method's performance are compared: a discrete Kalman Filter (KF) and second order Sliding Mode Differentiator (SMD) [Levant, 2003, p. 38],

$$\begin{aligned}\dot{z}_0 =& v_0; v_0 = -2L^{1/3}\lceil z_0 - f(t)\rfloor^{2/3} + z_1,\\ \dot{z}_1 =& v_1; v_1 = -1.5L^{1/2}\lceil z_1 - v_0\rfloor^{1/2} + z_2,\\ \dot{z}_2 =& v_2; v_2 = -1.1L\lceil z_2 - v_1\rfloor^{0}.\end{aligned} \tag{3.51}$$

which is in recursive form and L gain is the differentiator gain to tune.

Consider continuous KF as a pure double integrator for which dynamics are

$$\begin{aligned}\dot{\boldsymbol{\psi}}(t) &= \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\boldsymbol{\psi}(t) + \begin{bmatrix}0\\ 1\end{bmatrix}\boldsymbol{w}(t),\\ \boldsymbol{z}(t) &= \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{\psi}(t) + \boldsymbol{v}(t).\end{aligned} \tag{3.52}$$

Discrete KF [Grewal, 2011] is derived from continuous KF as follows. From [Reid, 2001], for a sample step Δt, integrating Equation (3.52) over one sample step and then shifting forward in time to match initial conditions yields

$$\begin{aligned}\boldsymbol{\psi}_{k+1} &= \begin{bmatrix}1 & \Delta t\\ 0 & 1\end{bmatrix}\boldsymbol{\psi}_k + \underbrace{\int_0^{\Delta t}\begin{bmatrix}\Delta t-\tau\\ 1\end{bmatrix}\boldsymbol{w}\left(t_k+\tau\right)d\tau}_{\mathbf{w}_k}, \\ \boldsymbol{z}_k &= \begin{bmatrix}1 & 0\end{bmatrix}\boldsymbol{\psi}_k + \boldsymbol{v}_k.\end{aligned} \tag{3.53}$$

After some mathematical manipulations, computing process noise covariance as $Q_k = E\left[\mathbf{w}_k\mathbf{w}_k^\top\right]$ with $E\left[\cdot\right]$ the mean of the process yields

$$Q_k = a\begin{bmatrix}(1/3)\Delta t^3 & (1/2)\Delta t^2\\ (1/2)\Delta t^2 & \Delta t\end{bmatrix}, \tag{3.54}$$

where $a > 0$ is the tunning parameter. Notice that Q_k is a semi-definite positive matrix as required.

An open loop sinusoidal motion in the $x-$coordinate is selected to tune the gains L and a for the SMD from Equation (3.51) and the discrete KF from Equation (3.53) - Equation (3.54), respectively, with $\omega_0 = 2\pi/T$ the following open loop command for both wheels

$$\omega_r = \left(\frac{60}{2\pi}\right)\left(\frac{R\omega_0}{r}\operatorname{sign}\left(\cos(\omega_0 t)\right)\,\cos(\omega_0 t)\right) = \omega_l.$$

In the next chapter voltages will be the control inputs w.r.t. positions as outputs and estimations of velocity will be needed in order to implement CTA algorithm. Therefore, it is advantegeous to analyze the performance of velocity estimations in this chapter.

Figure 3.27 - Figure 3.29 show that position and orientation filtering for both methods are very similar. Notice that since Figure 3.27 - Figure 3.29 are open loop experiments pure $x-$coordinate motion is not achieved since y-coordinate and θ orientation do not remain constant. Nevertheless, this fact will be helpful to tune the observer gains. After an exhaustive tunning procedure, SMD has a better estimation of velocity signals based on the critical points of these velocity signals but the resulting estimation is always a noisy signal; especially, when there are sudden changes in the signals as shown in Figure 3.28 and Figure 3.29. In contrast, eventhough KF estimated velocity signals are slightly delayed, they are considerably less noisy. It is preferrable that the DDR does not have abrupt behavior, therefore, KF is the velocity estimator selected for the following experimental tests; the same KF is applied to each signal during the experiments.

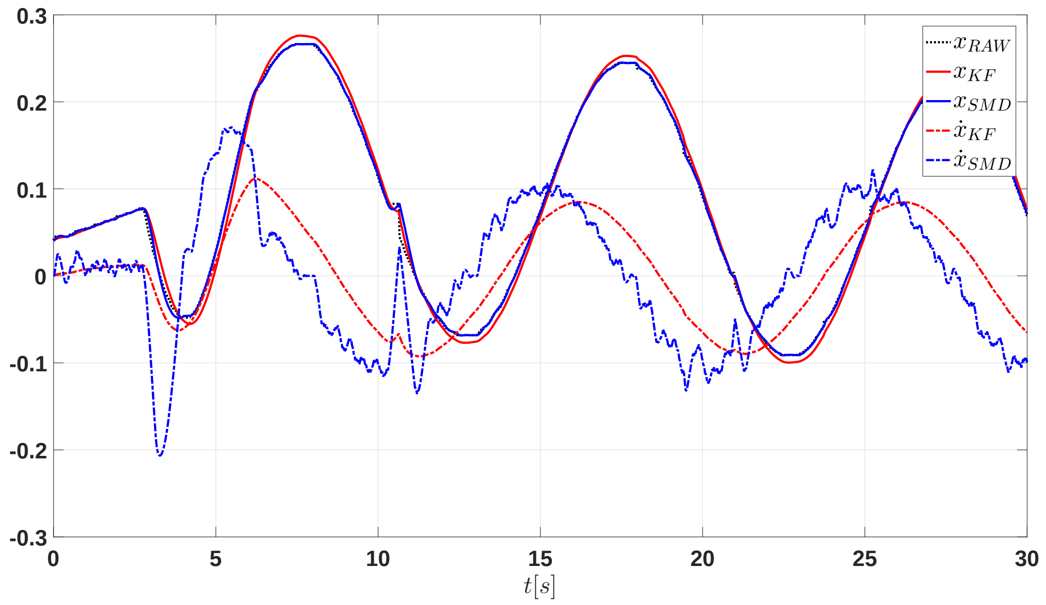


Figure 3.27: Filtering x signal

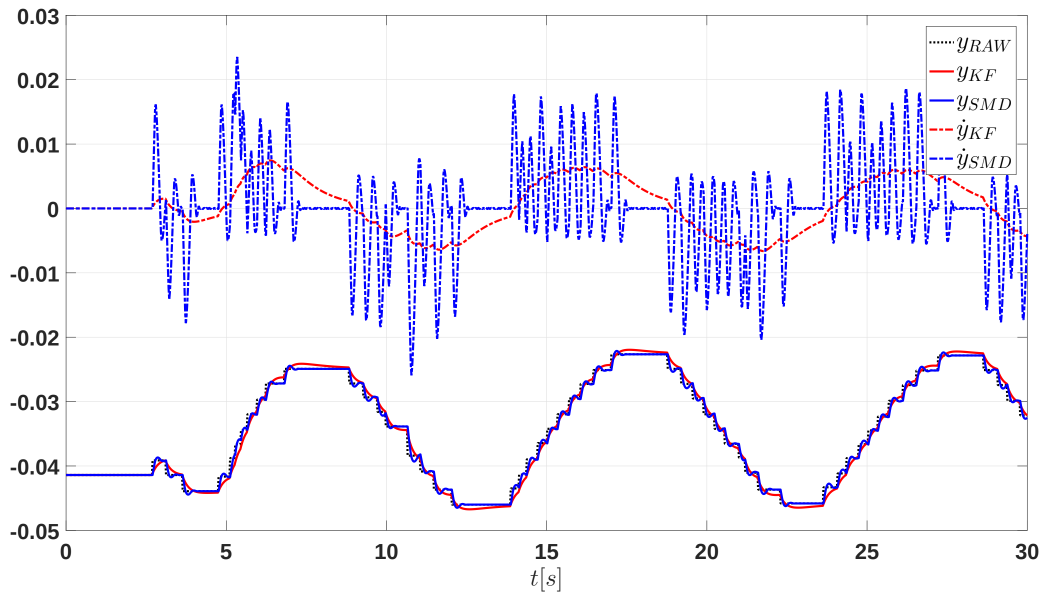


Figure 3.28: Filtering y signal

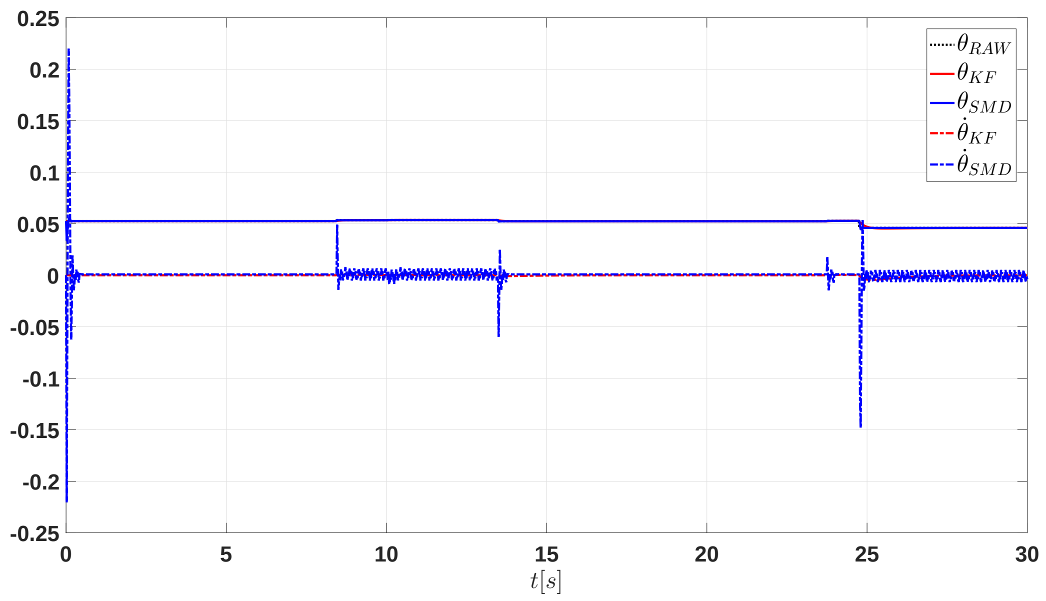


Figure 3.29: Filtering θ signal

Experimental tests

The reference trajectory is the one stated in Equation (3.50). An XY graph is presented to show the form of the trajectory in Figure 3.30. Trajectory tracking performance is shown in Figure 3.31 implementing control signals shown in Figure 3.33 and obtaining error signals shown in Figure 3.34.

For LIN, e_x and e_y are bounded between approximately $|0.08|$ because linear algorithms cannot compensate exactly matched disturbances and cannot take the state to zero in finite time; this is shown in Figure 3.24. From Figure 3.24 it would seem like SGN has smaller errors than LIN, especially on the y coordinate graph; however STA definitely has smaller errors on both coordinates compared to LIN and SGN.

As seen in Figure 3.36, for SGN algorithm the sign function looses its properties after transforming the control signal $\boldsymbol{\nu}$ to actuations for the wheels.

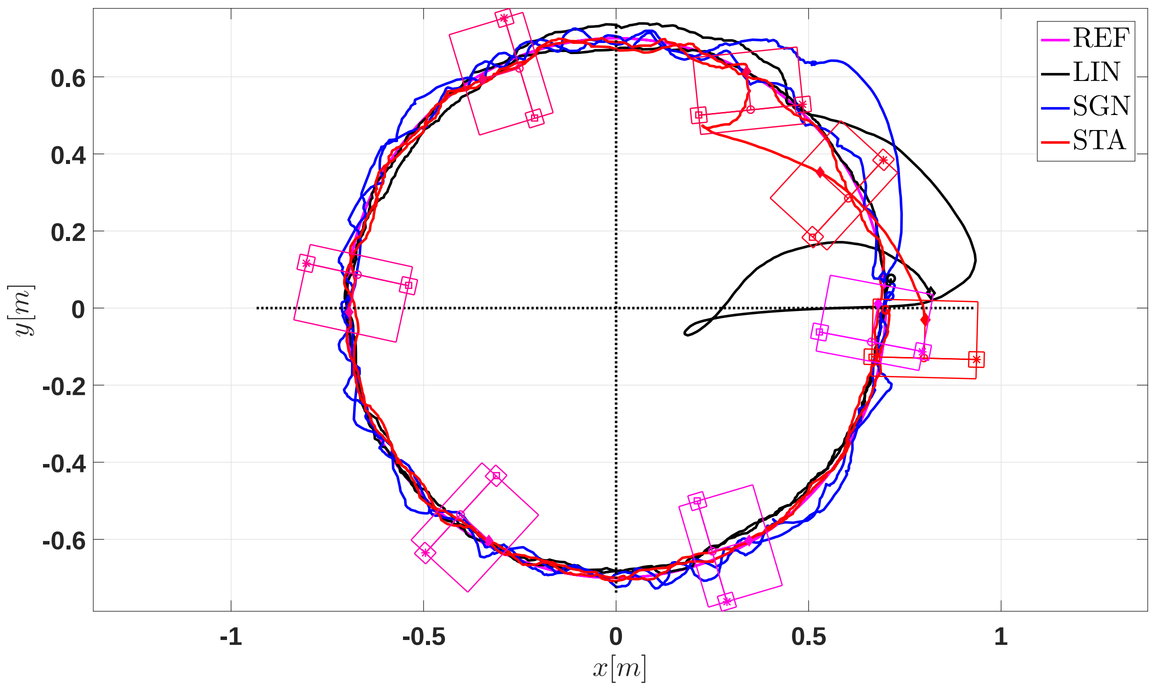


Figure 3.30: Experimental test: Tracking. XY plane. Controlled output signals

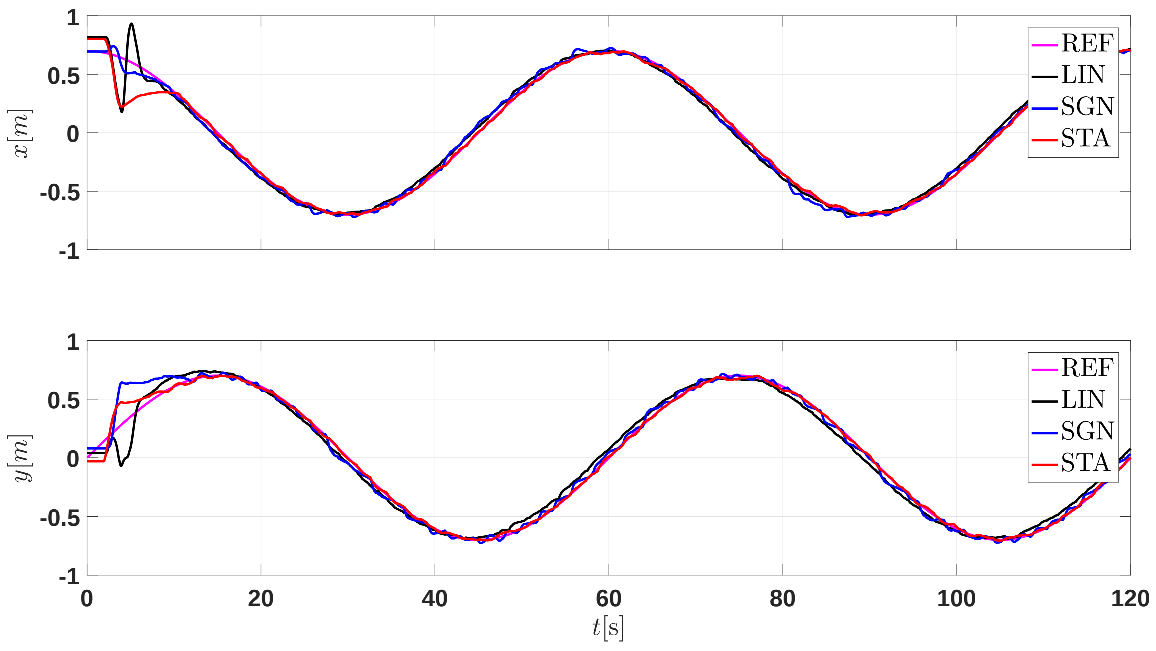


Figure 3.31: Experimental test: Tracking. Controlled output signals

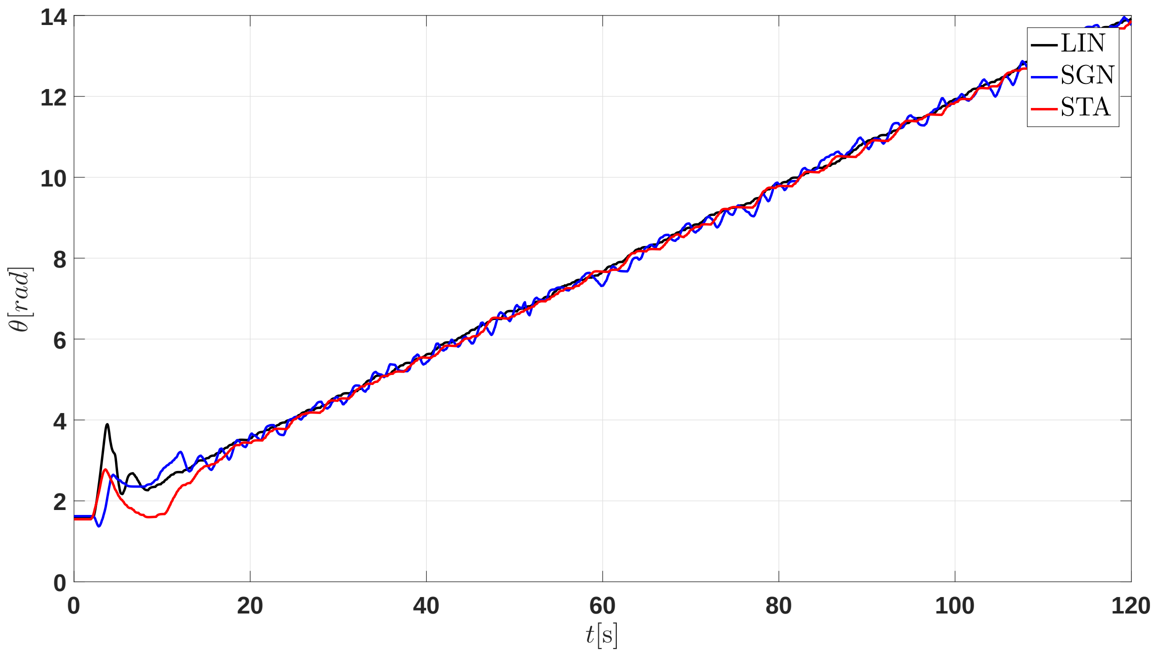


Figure 3.32: Experimental test: Tracking. Measured output signal

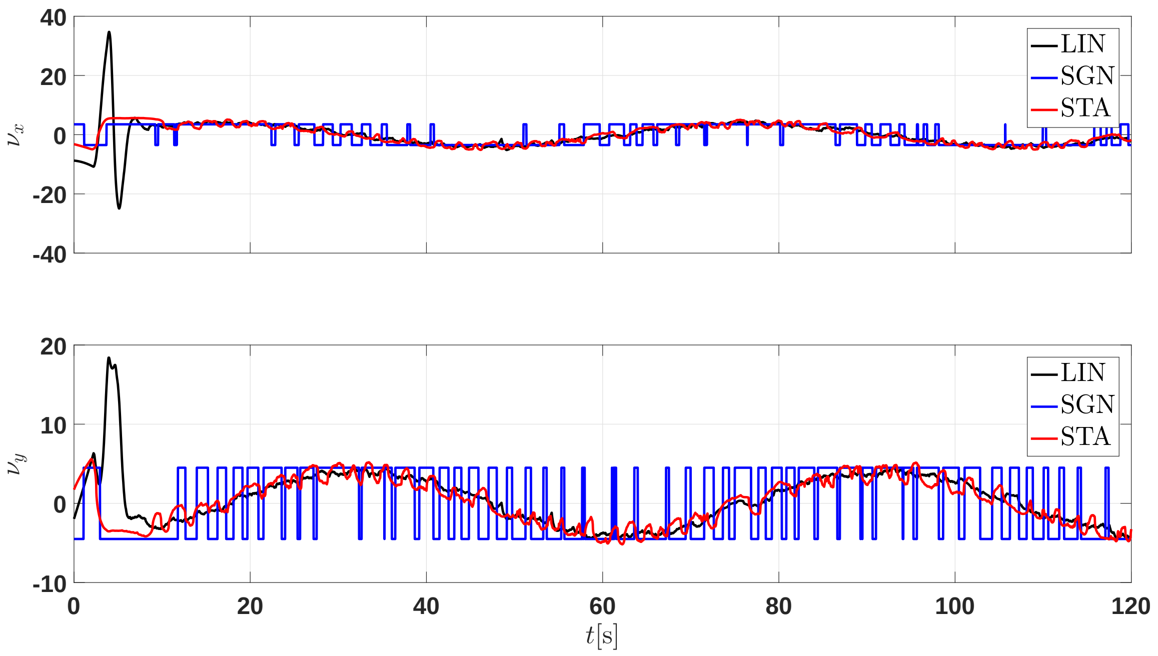


Figure 3.33: Experimental test: Tracking. Control signals

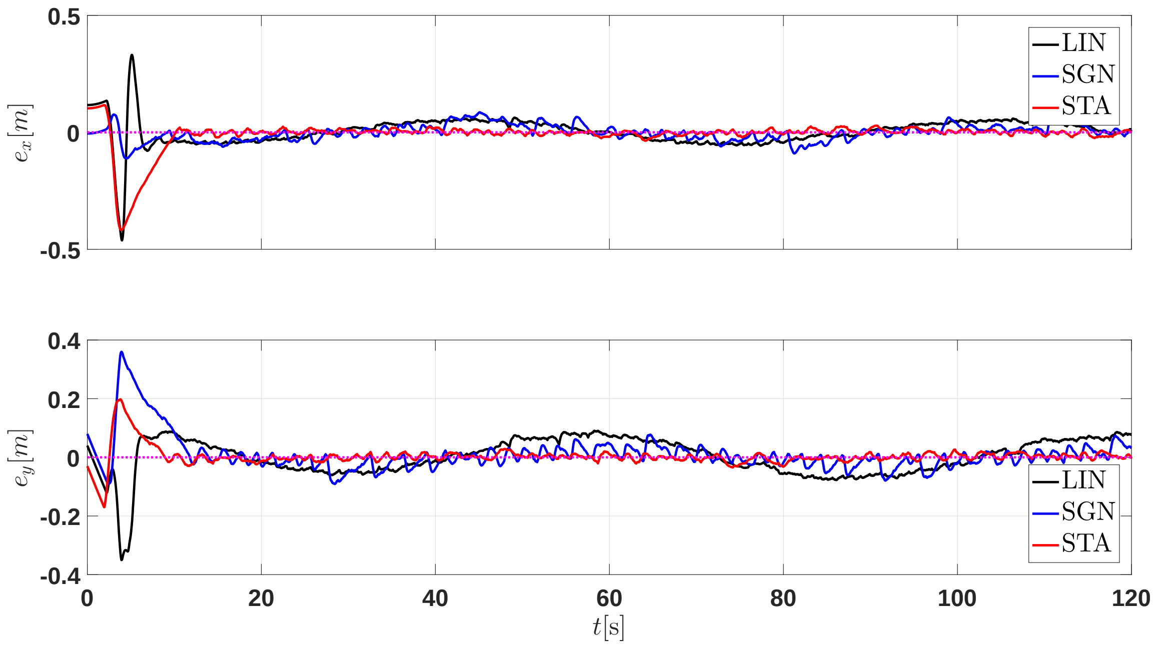


Figure 3.34: Experimental test: Tracking. Error signals

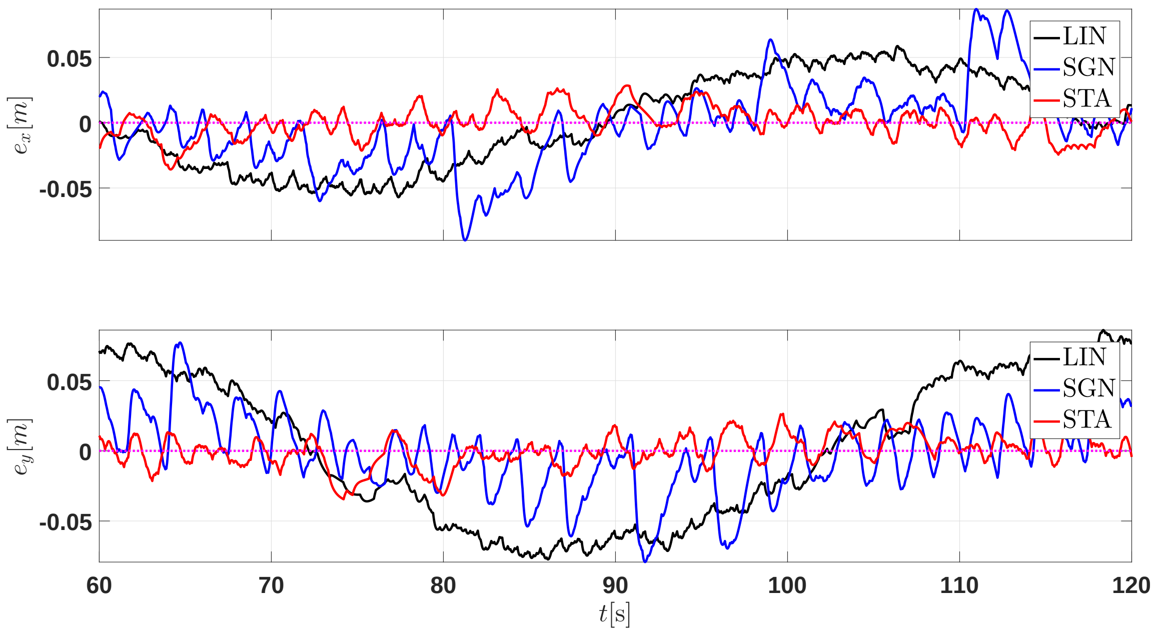


Figure 3.35: Experimental test: Tracking. Error signals (After convergence)

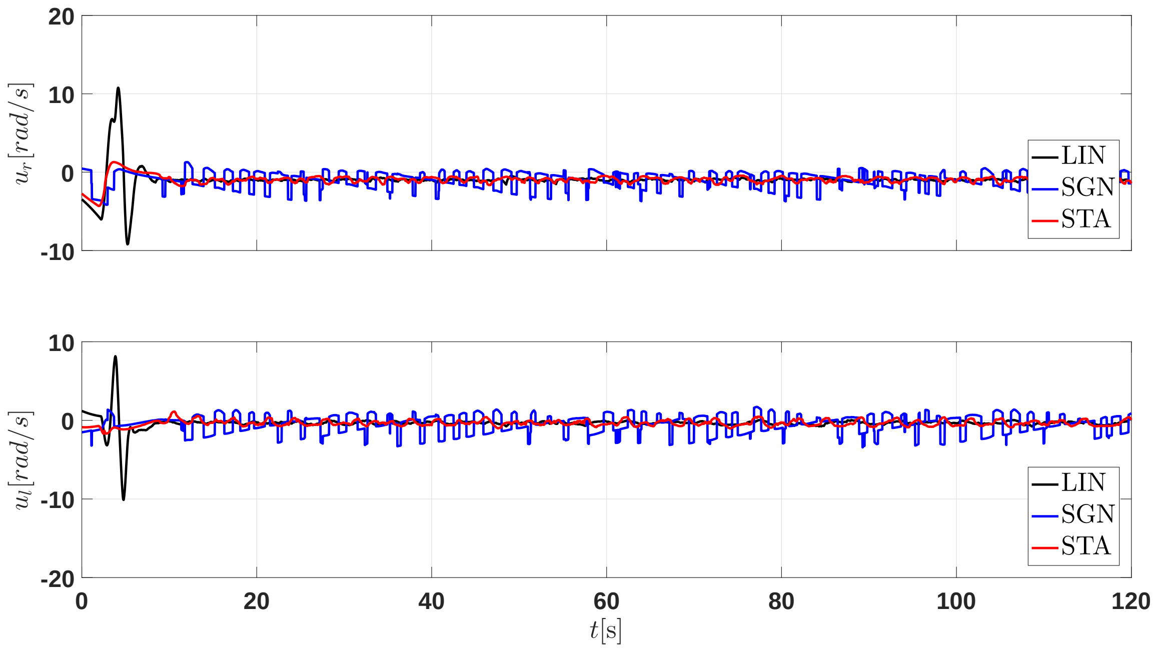


Figure 3.36: Experimental test: Tracking. Control signals. Wheels

Since STA showed the best error precision results in comparison with SGN and LIN algorithms, an experiment with another time-varying was performed using STA. The reference trajectory for this time-varying curve is given by the following equations

$$\begin{aligned} x_d(t) &= R\cos(\omega t) + h_c \qquad & \dot{x}_d(t) &= -R\omega\sin(\omega t) \\ y_d(t) &= 0.5R\sin(2\omega t) + k_c, \qquad & \dot{y}_d(t) &= +R\omega\cos(2\omega t) \end{aligned} \tag{3.55}$$

Figure 3.37 and Figure 3.38 show trajectory tracking performance on the XY plane and posture signals w.r.t. to time, respectively. The latter is accomplished with control signals shown in Figure 3.40 which yield the error signals shown in Figure 3.41.

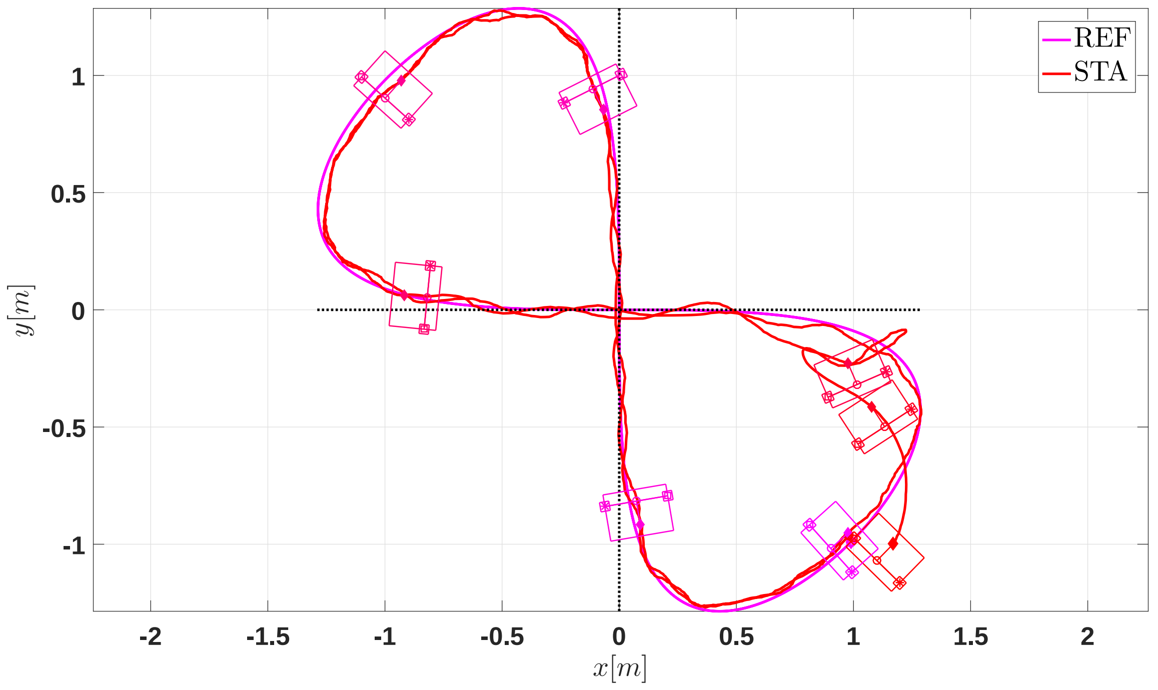


Figure 3.37: Experimental test: Tracking. Lemniscate. XY plane. Controlled output signals

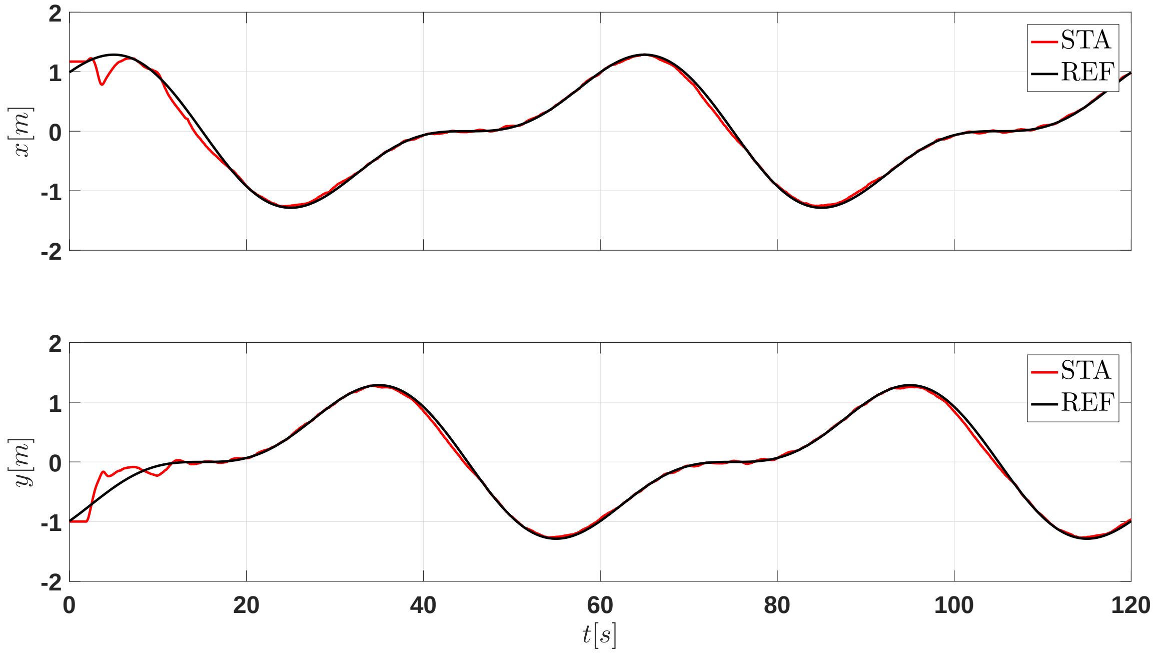


Figure 3.38: Experimental test: Tracking. Lemniscate. Controlled output signals

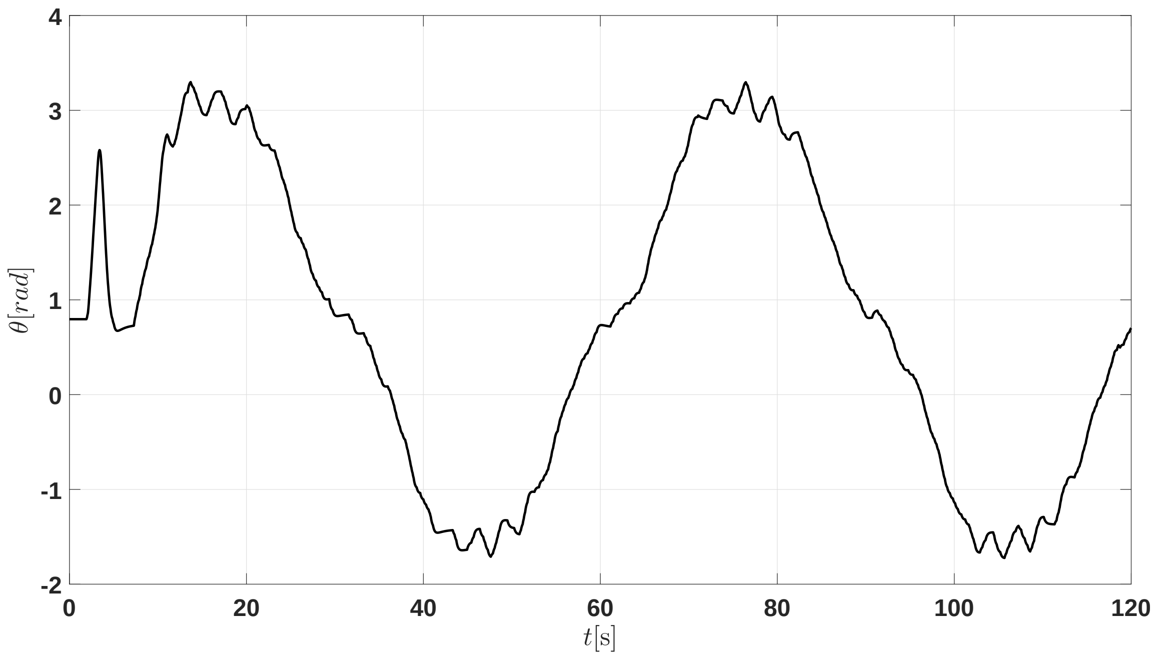


Figure 3.39: Experimental test: Tracking. Lemniscate. Measured output signal

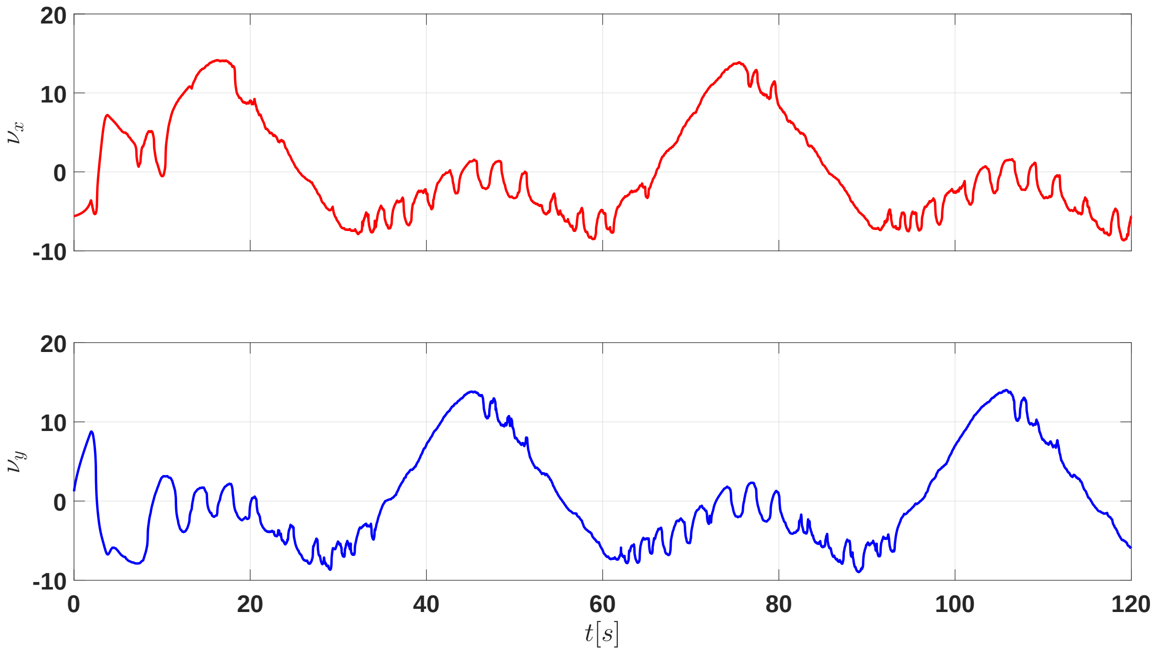


Figure 3.40: Experimental test: Tracking. Lemniscate. Control signals.

Figure 3.41 and Figure 3.42 show the error signals are, for most of the experiment's time, bounded between |0.06|m.

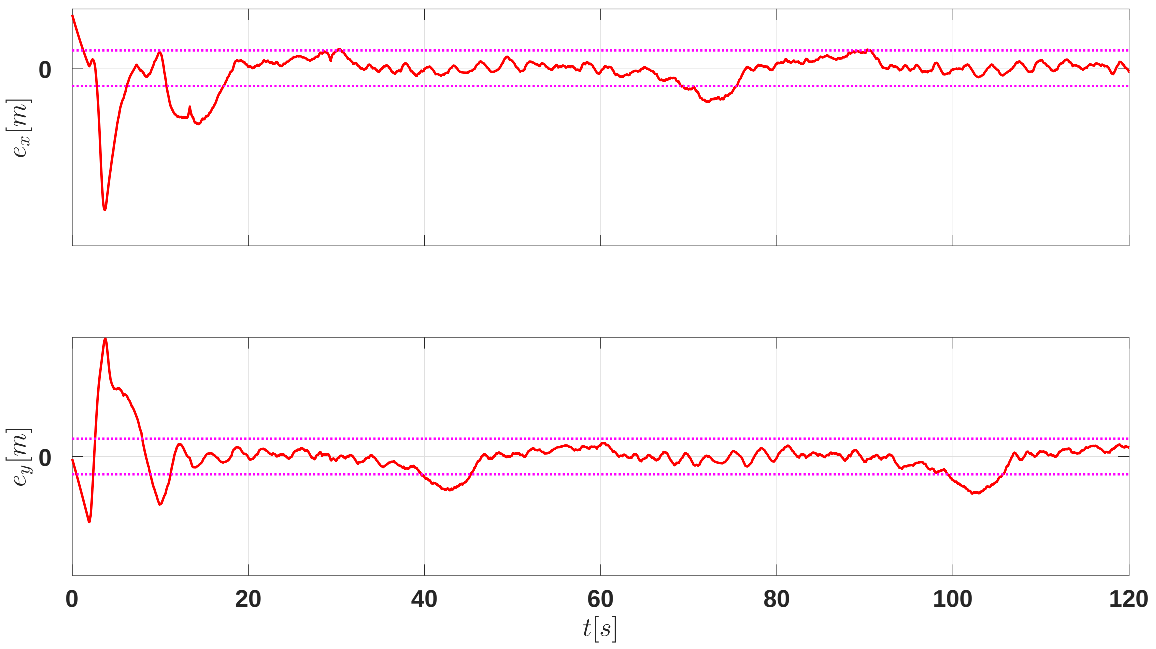


Figure 3.41: Experimental test: Tracking. Lemniscate. Error signals

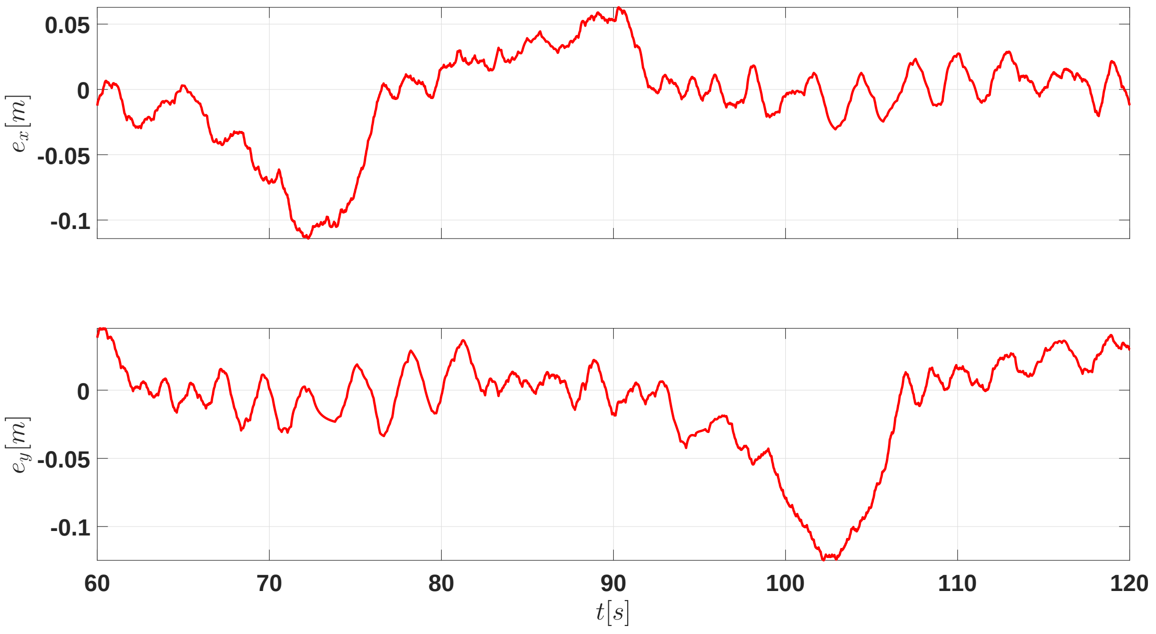


Figure 3.42: Experimental test: Tracking. Lemniscate. Error signals (After convergence)

Chapter 4

Voltages as control inputs

In this chapter the control objective is to solve trajectory tracking to drive posture errors and, also, velocity errors to zero in finite-time regardless of perturbations. For this, it is required to obtain a model of relative degree 2 ($r = 2$) w.r.t. the position as output and voltages as the control inputs. After obtaining this model, the control goal can be achieved using the "Continuous Twisting Algorithm" (CTA) [Torres-González et al., 2015; Kamal et al., 2016; Moreno et al., 2016].

4.1 Modelling

Differentiating Equation (3.11) once w.r.t. to time yields

$$\begin{bmatrix} \ddot{\boldsymbol{p}} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{D}(\theta,\dot{\theta}) \\ \mathbf{0}^{\top} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} + \begin{bmatrix} D(\theta) \\ \boldsymbol{\phi}^{\top} \end{bmatrix} \begin{bmatrix} \dot{\omega}_r \\ \dot{\omega}_l \end{bmatrix} + \underbrace{\begin{bmatrix} \dot{D}(\theta,\dot{\theta}) \\ \mathbf{0}^{\top} \end{bmatrix} \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} + \begin{bmatrix} D(\theta) \\ \boldsymbol{\phi}^{\top} \end{bmatrix} \begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix}}_{\rho}. \tag{4.1}$$

Model Equation (4.1) has three terms that need to be reformulated.

The first term in Equation (4.1) can be rewritten as a function of the states of the system as follows

$$\begin{aligned} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} &= M^{-1} \begin{bmatrix} v_a \\ \omega \end{bmatrix} \\ &= M^{-1} \begin{bmatrix} \sqrt{\dot{x}_a^2 + \dot{y}_a^2} \\ \dot{\theta} \end{bmatrix} \\ &= M^{-1} \begin{bmatrix} \sqrt{\left\{\dot{x} + h\dot{\theta}\sin(\theta)\right\}^2 + \left\{\dot{y} - h\dot{\theta}\cos(\theta)\right\}^2} \\ \dot{\theta} \end{bmatrix} \\ &= \boldsymbol{F}(\dot{\boldsymbol{p}}, \theta, \dot{\theta}) \in \mathbb{R}^2. \end{aligned} \tag{4.2}$$

And,

$$\dot{D}(\theta,\dot{\theta}) = \dot{\theta} \begin{bmatrix} -a_1\sin(\theta) - ha_2\cos(\theta) & -a_1\sin(\theta) + ha_2\cos(\theta) \\ +a_1\cos(\theta) - ha_2\sin(\theta) & +a_1\cos(\theta) + ha_2\sin(\theta) \end{bmatrix} \in \mathbb{R}^{2\times 2}.$$

Note that $\dot{D}(\theta, 0) = 0_{2\times 2}$ and $\boldsymbol{F}(\mathbf{0}, \theta, 0) = \mathbf{0}$.

The second term in Equation (4.1) has $u_r = \dot{\omega}_r$ and $u_l = \dot{\omega}_l$ accelerations as control inputs but the interest of this work is to consider the actuator dynamics and express these two control inputs in terms of motor voltages instead of accelerations.

$$\begin{aligned} J_m\dot{\omega} =& K_m I \\ L_m\dot{I} =& -R_m I - \phi_m(t,\omega) + V, \end{aligned} \tag{4.3}$$

where J_m $[N\,m\,s^2/rad]$, $K_m = k_m$ $[N\,m/A]$, L_m $[H]$ and R_m, $[\Omega]$. Disturbances $\phi_m(t,\omega)$ are assumed Lipschitz, i.e.,

$$|\phi_m(t,\omega)| \leq \phi_{m_{max}}, \quad |\dot{\phi}_m(t,\omega)| \leq L_m. \tag{4.4}$$

Using singular perturbation theory ($L_m = 0$) ([Khalil, 2002]) and after some mathematical manipulations, Equation (4.3) yields

$$\dot{\omega} = d_m V - \underbrace{b_m\phi_m(t,\omega)}_{\Delta_m}, \tag{4.5}$$

where $d_m = \frac{K_m}{J_m R_m} > 0$ and $b_m = \frac{K_m k_m}{J_m R_m} \geq 0$.

The latter analysis is made for each motor therefore it can be stated that

$$\begin{bmatrix} u_r \\ u_l \end{bmatrix} = \underbrace{\begin{bmatrix} d_r & 0 \\ 0 & d_l \end{bmatrix}}_{D_m} \begin{bmatrix} V_r \\ V_l \end{bmatrix} - \begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix}. \tag{4.6}$$

The last term in Equation (4.1) is conformed by perturbations that affect the system. All these terms can be gathered into one perturbation term denoted as $\bar{\boldsymbol{\rho}} = \begin{bmatrix} \bar{\rho}_1, & \bar{\rho}_2, & \bar{\rho}_3 \end{bmatrix} \in \mathbb{R}^3$ and $\boldsymbol{\rho}$ is assumed Lipschitz, i.e.,

$$|\bar{\rho}_i(t)| \leq \rho_{i_{max}}, \quad |\dot{\bar{\rho}}_i(t)| \leq L_i, \ i = 1,2,3 \tag{4.7}$$

Hence, Equation (4.1) is rewritten as

$$\begin{bmatrix} \ddot{\boldsymbol{p}} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{D} \\ \mathbf{0}^\top \end{bmatrix} \boldsymbol{F}(\dot{\boldsymbol{p}}, \theta, \dot{\theta}) + \begin{bmatrix} D \\ \boldsymbol{\phi}^\top \end{bmatrix} D_m \begin{bmatrix} V_r \\ V_l \end{bmatrix} \underbrace{- \begin{bmatrix} D \\ \boldsymbol{\phi}^\top \end{bmatrix} \begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix} + \begin{bmatrix} \dot{D} \\ \mathbf{0}^\top \end{bmatrix} \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} + \begin{bmatrix} D \\ \boldsymbol{\phi}^\top \end{bmatrix} \begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix}}_{\bar{\boldsymbol{\rho}}} \tag{4.8}$$

where $\dot{D} = \dot{D}(\theta, \dot{\theta})$ and $D = D(\theta)$ to avoid verbose notation.

Transforming Equation (4.8) to a state space representation using

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{4.9}$$

results in

$$\begin{aligned}\begin{bmatrix}\dot{x}_1\\ \dot{x}_3\end{bmatrix} &= \begin{bmatrix}x_2\\ x_4\end{bmatrix} = D\left(\boldsymbol{F}(\boldsymbol{x}) + \begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix}\right)\\ \begin{bmatrix}\dot{x}_2\\ \dot{x}_4\end{bmatrix} &= \boldsymbol{f}(\boldsymbol{x}) + \bar{\boldsymbol{u}} + \begin{bmatrix}\bar{\rho}_1\\ \bar{\rho}_2\end{bmatrix}\\ \dot{x}_5 &= x_6\\ \dot{x}_6 &= \boldsymbol{\phi}^\top D^{-1}\bar{\boldsymbol{u}} + \bar{\rho}_3\end{aligned} \tag{4.10}$$

where $\boldsymbol{f}(\boldsymbol{x}) = \dot{D}\boldsymbol{F}(\boldsymbol{x}) \in \mathbb{R}^2$ is a term of known dynamics,

$$\bar{\boldsymbol{u}} = \begin{bmatrix}u_x\\ u_y\end{bmatrix} = DD_m\begin{bmatrix}V_r\\ V_l\end{bmatrix}, \tag{4.11}$$

and

$$\begin{bmatrix}\bar{\rho}_1\\ \bar{\rho}_2\end{bmatrix} = -D\begin{bmatrix}\Delta_r\\ \Delta_l\end{bmatrix} + \dot{D}\begin{bmatrix}\rho_r\\ \rho_l\end{bmatrix} + D\begin{bmatrix}\dot{\rho}_r\\ \dot{\rho}_l\end{bmatrix}, \tag{4.12}$$

$$\bar{\rho}_3 = -\boldsymbol{\phi}^\top\begin{bmatrix}\Delta_r\\ \Delta_l\end{bmatrix} + \boldsymbol{\phi}^\top\begin{bmatrix}\dot{\rho}_r\\ \dot{\rho}_l\end{bmatrix}. \tag{4.13}$$

Remark 1 *When $x_2 = x_4 = x_6 = 0$, then vector function $f(\boldsymbol{x} = 0) = 0$.*

4.2 Control design

Trajectory tracking for a DDR based on System (4.10) implementing a continuous sliding mode algorithm is intended. In other words, control objective is to drive position and velocity tracking errors to zero in finite-time regardless of perturbations.

Suppose $\boldsymbol{p_d}(t) = \begin{bmatrix}x_d(t), & y_d(t)\end{bmatrix}^\top \in \mathbb{R}^2$ is continuous and twice differentiable such that $\dot{\boldsymbol{p}}_{\boldsymbol{d}}(t) = \begin{bmatrix}\dot{x}_d(t), & \dot{y}_d(t)\end{bmatrix}^\top \in \mathbb{R}^2$ and $\ddot{\boldsymbol{p}}_{\boldsymbol{d}}(t) = \begin{bmatrix}\ddot{x}_d(t), & \ddot{y}_d(t)\end{bmatrix}^\top \in \mathbb{R}^2$ exist. Also, $\boldsymbol{p_d}(t)$, $\dot{\boldsymbol{p}}_{\boldsymbol{d}}(t)$ and $\ddot{\boldsymbol{p}}_{\boldsymbol{d}}(t)$ are known. In the following (t) is omitted to avoid verbose notation. Attempting to do trajectory tracking for x and y coordinates forces orientation dynamics to be the zero dynamics of System (4.10) making θ a measured uncontrolled output.

Let error variables be defined as

$$\boldsymbol{e_1} = \begin{bmatrix}x_1\\ x_3\end{bmatrix} - \boldsymbol{p_d}, \quad \dot{\boldsymbol{e}}_{\boldsymbol{1}} = \underbrace{\begin{bmatrix}x_2\\ x_4\end{bmatrix} - \dot{\boldsymbol{p}}_{\boldsymbol{d}}}_{\boldsymbol{e_2}}. \tag{4.14}$$

Using Equation (4.11) and differentiating Equation (4.14) w.r.t. time, the error dynamics yields

$$\begin{aligned}\dot{\boldsymbol{e}}_{\boldsymbol{1}} &= \boldsymbol{e_2}\\ \dot{\boldsymbol{e}}_{\boldsymbol{2}} &= \begin{bmatrix}\dot{x}_2\\ \dot{x}_4\end{bmatrix} - \ddot{\boldsymbol{p}}_{\boldsymbol{d}} = \boldsymbol{f} + DD_m\begin{bmatrix}V_r\\ V_l\end{bmatrix} + \begin{bmatrix}\bar{\rho}_1\\ \bar{\rho}_2\end{bmatrix} - \ddot{\boldsymbol{p}}_{\boldsymbol{d}},\end{aligned} \tag{4.15}$$

where

$$\begin{bmatrix} V_r \\ V_l \end{bmatrix} = D_m^{-1} D^{-1} \boldsymbol{u} \in \mathbb{R}^2 \tag{4.16}$$

and

$$\boldsymbol{u} = -\boldsymbol{f} + \boldsymbol{\nu} + \ddot{\boldsymbol{p}}_{\boldsymbol{d}} \in \mathbb{R}^2 \tag{4.17}$$

with $\boldsymbol{\nu} = \begin{bmatrix} \nu_x, & \nu_y \end{bmatrix}^\top$ is a new control input. Hence, Equation (4.15) can be rewritten as

$$\dot{\boldsymbol{e_1}} = \boldsymbol{e_2}, \quad \dot{\boldsymbol{e_2}} = \boldsymbol{\nu} + \begin{bmatrix} \bar{\rho}_1 \\ \bar{\rho}_2 \end{bmatrix} \tag{4.18}$$

System Equation (4.18) has the form of two independently disturbed double integrators. Then the control goal can be achieved using the "Continuous Twisting Algorithm" (CTA) ([Torres-González et al., 2015; Moreno et al., 2016]) which has the following structure

$$\begin{aligned} \boldsymbol{\nu} &= -K_1 \lceil \boldsymbol{e_1} \rfloor^{1/3} - K_2 \lceil \boldsymbol{e_2} \rfloor^{1/2} + \boldsymbol{\eta}, \\ \dot{\boldsymbol{\eta}} &= -K_3 \lceil \boldsymbol{e_1} \rfloor^{0} - K_4 \lceil \boldsymbol{e_2} \rfloor^{0}, \end{aligned} \tag{4.19}$$

where

$$\lceil \boldsymbol{e_1} \rfloor^{\gamma} = \begin{bmatrix} |x - x_d|^{\gamma} \mathrm{sign}(x - x_d) \\ |y - y_d|^{\gamma} \mathrm{sign}(y - y_d) \end{bmatrix} \in \mathbb{R}^2, \quad \lceil \boldsymbol{e_2} \rfloor^{\gamma} = \begin{bmatrix} |\dot{x} - \dot{x}_d|^{\gamma} \mathrm{sign}(\dot{x} - \dot{x}_d) \\ |\dot{y} - \dot{y}_d|^{\gamma} \mathrm{sign}(\dot{y} - \dot{y}_d) \end{bmatrix} \in \mathbb{R}^2,$$

$$\boldsymbol{\eta} = \begin{bmatrix} \eta_x \\ \eta_y \end{bmatrix} \in \mathbb{R}^2,$$

and $K_i = \mathrm{diag}\left(\begin{bmatrix} k_{ix}, k_{iy} \end{bmatrix}\right)$ with $i = 1, 2, 3, 4$ such that finite-time convergence is achieved. Notice that Equation (4.19) is also a two-independent control law but CTA takes into account position and, also, velocity tracking errors.

The closed loop Equation (4.18)-Equation (4.19) can be rewritten as

$$\begin{aligned} \dot{\boldsymbol{e_1}} &= \boldsymbol{e_2} \\ \dot{\boldsymbol{e_2}} &= -K_1 \lceil \boldsymbol{e_1} \rfloor^{1/3} - K_2 \lceil \boldsymbol{e_2} \rfloor^{1/2} + \boldsymbol{e_3} \\ \dot{\boldsymbol{e_3}} &= -K_3 \lceil \boldsymbol{e_1} \rfloor^{0} - K_4 \lceil \boldsymbol{e_2} \rfloor^{0} + \begin{bmatrix} \dot{\bar{\rho}}_1 \\ \dot{\bar{\rho}}_2 \end{bmatrix} \end{aligned} \tag{4.20}$$

CTA guarantees (Torres-González et al. [2015]) a third order sliding mode for closed loop dynamics Equation (4.20) where

$$\boldsymbol{e_1} = \boldsymbol{0}, \ \boldsymbol{e_2} = \ \boldsymbol{0}, \text{ and } \boldsymbol{e_3} = \boldsymbol{0} \rightarrow \boldsymbol{\eta} = -\begin{bmatrix} \bar{\rho}_1 \\ \bar{\rho}_2 \end{bmatrix} \tag{4.21}$$

The block diagram of the closed loop is shown in Figure 4.1.

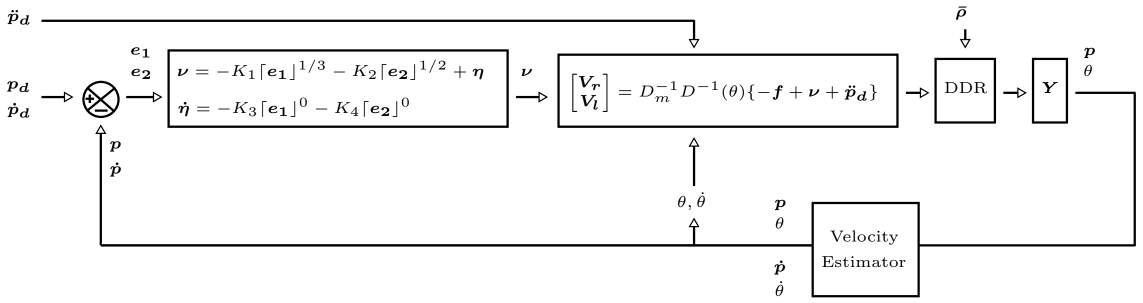

$\ddot{\boldsymbol{p}}_d$
$\boldsymbol{p_d}$
$\dot{\boldsymbol{p}}_d$
$\boldsymbol{e_1}$
$\boldsymbol{e_2}$
$\boldsymbol{\nu} = -K_1\lceil \boldsymbol{e_1}\rfloor^{1/3} - K_2\lceil \boldsymbol{e_2}\rfloor^{1/2} + \boldsymbol{\eta}$
$\dot{\boldsymbol{\eta}} = -K_3\lceil \boldsymbol{e_1}\rfloor^{0} - K_4\lceil \boldsymbol{e_2}\rfloor^{0}$
$\boldsymbol{\nu}$
$\begin{bmatrix}\boldsymbol{V_r}\\ \boldsymbol{V_l}\end{bmatrix} = D_m^{-1}D^{-1}(\theta)\{-\boldsymbol{f}+\boldsymbol{\nu}+\ddot{\boldsymbol{p}}_d\}$
$\bar{\boldsymbol{\rho}}$
DDR
$\boldsymbol{Y}$
$\boldsymbol{p}$
θ
$\boldsymbol{p}$
$\dot{\boldsymbol{p}}$
$\theta, \dot{\theta}$
$\boldsymbol{p}$
θ
Velocity
Estimator
$\dot{\boldsymbol{p}}$
$\dot{\theta}$

Figure 4.1: Relative degree 2: Block Diagram

Control gains must be tunned homogeneously according to paremeters Ł$_1$ and L_2 which are the Lipschitz bounds of the disturbances for channel x and y respectively. The next procedure can be done for each channel analogously making $L = L_1$ or $L = L_2$ and transforming with their corresponding error variables. Using the following change of variables

$$z_1 = Le_1,\ z_2 = \ \ Le_2,\ z_3 = Le_3 \quad \Rightarrow\ e_1 = \frac{z_1}{L},\ e_2 = \frac{z_2}{L},\ e_3 = \frac{z_3}{L}$$

yields the closed loop

$$\begin{aligned}\dot{z}_1 &= z_2\\ \dot{z}_2 &= -\bar{k}_1\lceil z_1\rfloor^{1/3} - \bar{k}_2\lceil z_2\rfloor^{1/2} + z_3\\ \dot{z}_3 &= -\bar{k}_3\lceil z_1\rfloor^{0} - \bar{k}_4\lceil z_2\rfloor^{0}\end{aligned}$$

which has the same structure as Equation (4.20). The only difference is that control gains are now given by $\bar{k}_1 = k_1L^{2/3}$, $\bar{k}_2 = k_2L^{1/2}$, $\bar{k}_3 = k_3L$ and $\bar{k}_4 = k_4L$.

Zero Dynamics

States x_5 and x_6 represent the zero dynamics of the system and are given by

$$\begin{aligned}\dot{x}_5 &= x_6\\ \dot{x}_6 &= \boldsymbol{\phi}^\top D^{-1}\bar{\boldsymbol{u}} + \bar{\rho}_3\end{aligned} \tag{4.22}$$

In closed loop, during transient response states x_5 and x_6 are given by

$$\begin{aligned}\dot{x}_5 &= x_6\\ \dot{x}_6 &= \boldsymbol{\phi}^\top D^{-1}\bar{\boldsymbol{u}} + \bar{\rho}_3\\ &= \boldsymbol{\phi}^\top D^{-1}\left(-\boldsymbol{f} - K_1\lceil \boldsymbol{e_1}\rfloor^{1/3} - K_2\lceil \boldsymbol{e_2}\rfloor^{1/2} + \boldsymbol{\eta} + \ddot{\boldsymbol{q}}_d\right) + \bar{\rho}_3\\ \dot{\boldsymbol{\eta}} &= -K_3\lceil \boldsymbol{e_1}\rfloor^{0} - K_4\lceil \boldsymbol{e_2}\rfloor^{0}.\end{aligned} \tag{4.23}$$

Subsystem Equation (4.23) shows that the dynamics for states x_5 and x_6 are affected by exogenous signals. During transient response little can be said about x_5 dynamics, however, depending on the desired trajectory some observations can be made about x_5 dynamics when the sliding mode is reached. From Equation (4.10), when Equation (4.21) occurs, it implies

$$\begin{aligned}\boldsymbol{e_1} =& 0 \Leftrightarrow \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \boldsymbol{p_d}, \\ \boldsymbol{e_2} =& 0 \Leftrightarrow \begin{bmatrix} x_2 \\ x_4 \end{bmatrix} = D\left(\boldsymbol{F}(\boldsymbol{x}) + \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix}\right) = \dot{\boldsymbol{p}}_{\boldsymbol{d}} \Leftrightarrow \left(\boldsymbol{F}(\boldsymbol{x}) + \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix}\right) = D^{-1}\dot{\boldsymbol{p}}_{\boldsymbol{d}},\end{aligned} \tag{4.24}$$

and recalling Equation (4.12) the following also occurs

$$\boldsymbol{\nu} = -\begin{bmatrix} \bar{\rho}_1 \\ \rho_2 \end{bmatrix} = D\begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix} - \dot{D}\begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} - D\begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix}. \tag{4.25}$$

Transforming Equation (4.25) using Equation (4.17) yields

$$\boldsymbol{u} = -\boldsymbol{f} + D\begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix} - \dot{D}\begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} - D\begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix} + \ddot{\boldsymbol{p}}_{\boldsymbol{d}}. \tag{4.26}$$

Performing two transformations, first, using Equation (4.17), and, second, using Equation (4.16) yields

$$\bar{\boldsymbol{u}} = -\boldsymbol{f} + D\begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix} - \dot{D}\begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} - D\begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix} + \ddot{\boldsymbol{p}}_{\boldsymbol{d}}. \tag{4.27}$$

Substituting Equation (4.27) and Equation (4.13) in Equation (4.22) yields

$$\begin{aligned}\dot{x_5} =& x_6 \\ \dot{x_6} =& -\boldsymbol{\phi}^\top D^{-1}\boldsymbol{f} + \cancel{\boldsymbol{\phi}^\top \begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix}} - \boldsymbol{\phi}^\top D^{-1}\dot{D}\begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix} - \cancel{\boldsymbol{\phi}^\top \begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix}} + \boldsymbol{\phi}^\top D^{-1}\ddot{\boldsymbol{p}}_{\boldsymbol{d}} \\ & \cancel{-\boldsymbol{\phi}^\top \begin{bmatrix} \Delta_r \\ \Delta_l \end{bmatrix}} \cancel{+\boldsymbol{\phi}^\top \begin{bmatrix} \dot{\rho}_r \\ \dot{\rho}_l \end{bmatrix}}.\end{aligned} \tag{4.28}$$

After rewritting $\boldsymbol{f}$ from Equation (4.28) and substituting Equation (4.24) in it yields

$$\begin{aligned}\dot{x_5} =& x_6 \\ \dot{x_6} =& -\boldsymbol{\phi}^\top D^{-1}\dot{D}\left(\boldsymbol{F} + \begin{bmatrix} \rho_r \\ \rho_l \end{bmatrix}\right) + \boldsymbol{\phi}^\top D^{-1}\ddot{\boldsymbol{p}}_{\boldsymbol{d}} \\ =& -\boldsymbol{\phi}^\top D^{-1}\dot{D}D^{-1}\dot{\boldsymbol{p}}_{\boldsymbol{d}} + \boldsymbol{\phi}^\top D^{-1}\ddot{\boldsymbol{p}}_{\boldsymbol{d}}\end{aligned} \tag{4.29}$$

Now, computing the matrix operations of Equation (4.29) yields

$$\begin{aligned}\dot{x_5} =& x_6 \\ \dot{x_6} =& -\left[x_6\frac{\cos(x_5)}{h}, \quad x_6\frac{\sin(x_5)}{h}\right]\dot{\boldsymbol{p}}_{\boldsymbol{d}} + \left[\frac{-\sin(x_5)}{h}, \quad \frac{\cos(x_5)}{h}\right]\ddot{\boldsymbol{p}}_{\boldsymbol{d}}\end{aligned} \tag{4.30}$$

Remark 2 *Differentiating Equation* (3.29) *w.r.t. time yields Equation* (4.31) *which shows Equation* (4.31) *is only a dynamic extension of Equation* (3.29)*. Therefore,*

$$x_6 = \left[\frac{-\sin(x_5)}{h}, \quad \frac{\cos(x_5)}{h}\right]\dot{\boldsymbol{p}}_{\boldsymbol{d}}. \tag{4.31}$$

4.3 Simulations

The parameters of the robot are $h = 0.1\text{m}$, $r = 0.05\text{m}$ and $d = 0.272\text{m}$. To show CTA strengths, this control law is compared with two other control algorithms. First, a homogeneous control law without sliding modes (BER) which has the following structure

$$\boldsymbol{\nu} = -K_1 \lceil \boldsymbol{e_1} \rfloor^{1/3} - K_2 \lceil \boldsymbol{e_2} \rfloor^{1/2} \tag{4.32}$$

where $K_1 = \text{diag}(k_{1x}, k_{1y})$ and $K_2 = \text{diag}(k_{2x}, k_{2y})$.

Second, the Twisting algorithm (TWA) which has the following structure

$$\boldsymbol{\nu} = -K_1 \lceil \boldsymbol{e_1} \rfloor^{0} - K_2 \lceil \boldsymbol{e_2} \rfloor^{0} \tag{4.33}$$

where $K_1 = \text{diag}(k_{1x}, k_{1y})$ and $K_2 = \text{diag}(k_{2x}, k_{2y})$. The reference trajectory is a circle with radius $R = 0.7\text{m}$ and center $(h, k) = (0, 0)\text{m}$ described by the following

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} R\cos(\omega t) + h \\ R\sin(\omega t) + k \end{bmatrix} \text{m}, \quad \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix} = \begin{bmatrix} -R\omega\sin(\omega t) \\ +R\omega\cos(\omega t) \end{bmatrix} \frac{\text{m}}{\text{s}}, \quad \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} = \begin{bmatrix} -R\omega^2\cos(\omega t) \\ -R\omega^2\sin(\omega t) \end{bmatrix} \frac{\text{m}}{\text{s}^2} \tag{4.34}$$

where $\omega = 2\pi/T$ with $T = 60\text{s}$. Initial conditions are set to

$$\begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix} = \begin{bmatrix} -0.7\text{m} \\ +0.7\text{m} \\ 0\text{rad} \end{bmatrix}$$

Figure 4.2 shows the DDR's motion in the XY plane using the control signals shown in Figure 4.5. Evolution in time of controlled position outputs is shown in Figure 4.3 and the measured orientation output is shown in Figure 4.4. Figure 4.6 shows the result of the estimation using the SMD from Equation (3.51).

Generated error signals are shown in Figure 4.7. A close-up to these signals after convergence is shown in Figure 4.8. For BER, e_x and e_y are bounded because linear algorithms cannot compensate exactly matched disturbances and cannot take the state to zero in finite time; this is shown in Figure 4.8.

However, TWA only guarantees a second-order sliding mode and CTA guarantees a third-order sliding mode. Since, CTA achieves a higher order sliding mode than TWA, CTA guarantees better error precision. In order to show that the CTA and TWA are actually reaching the sliding mode (SM), e_x and e_y are analyzed performing the same simulation but for sampling step $\tau = 10^{-3}$ and $\tau = 10^{-4}$. For CTA, Figure 4.9a and Figure 4.10a show that the SM is present because the precision of the algorithm is increased by decreasing the sampling step (Levant [1998]; Moreno [2009]). The same result is obtained for TWA as shown in Figure 4.9b and Figure 4.10b and as expected control signal is discontinuous. Summarizing in Table 4.1, in both sampling step cases CTA has the best error precision compared to BER and TWA performances.

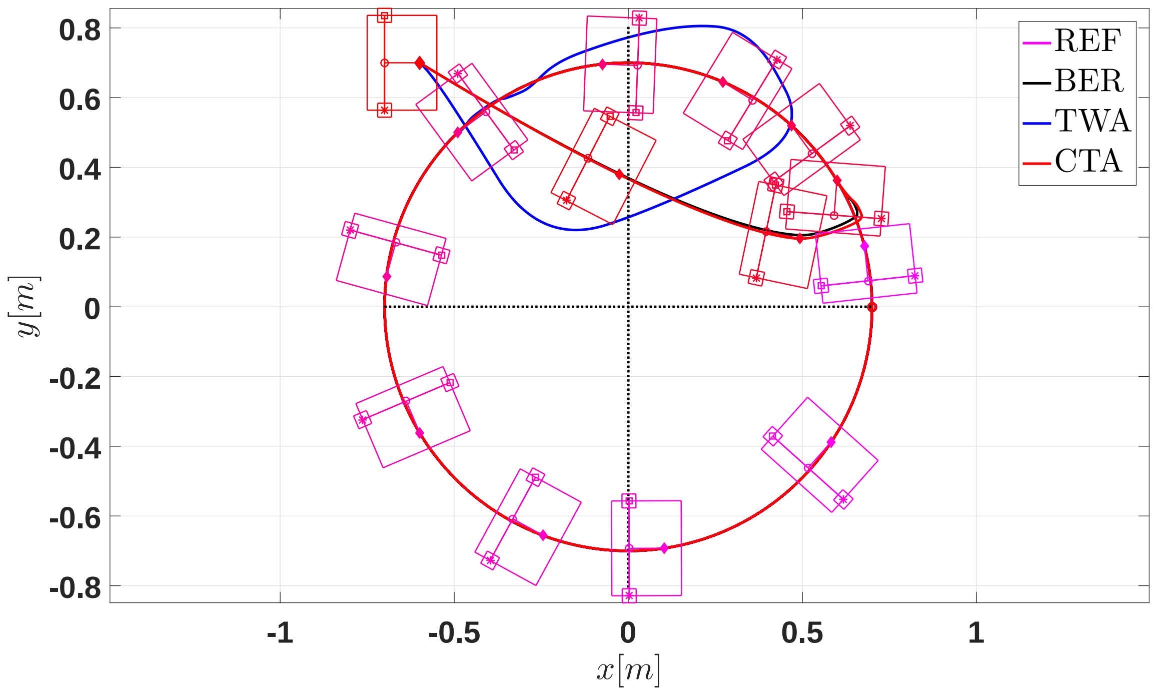


Figure 4.2: Simulation: Tracking. XY plane. Controlled output signals

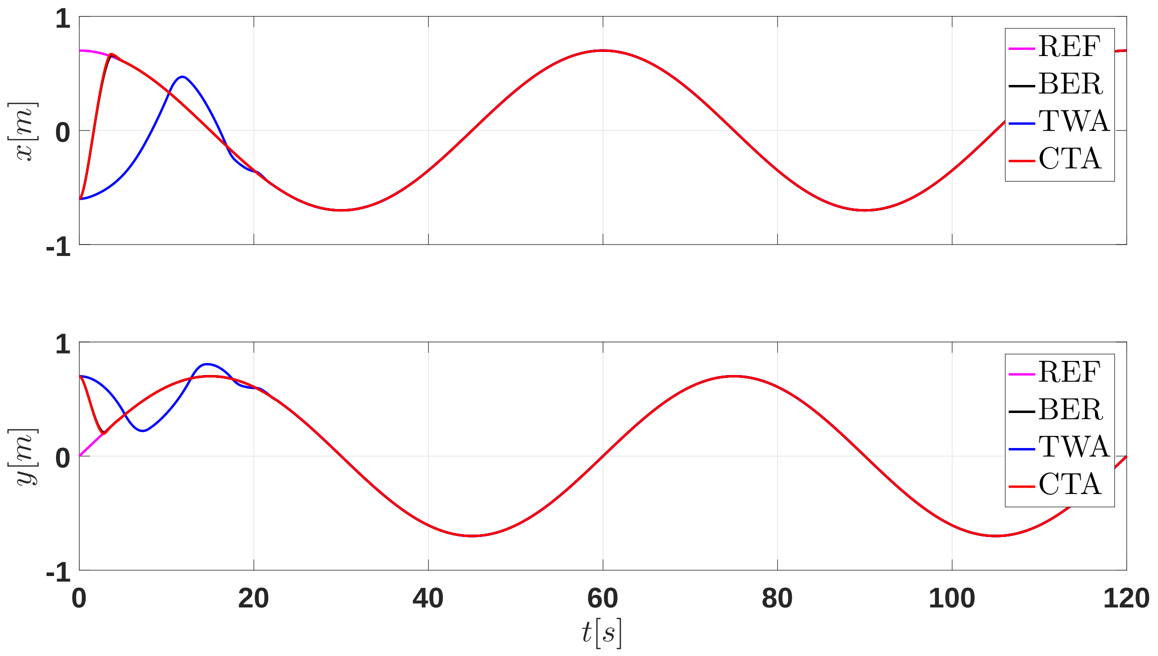


Figure 4.3: Simulation: Tracking. Controlled output signals

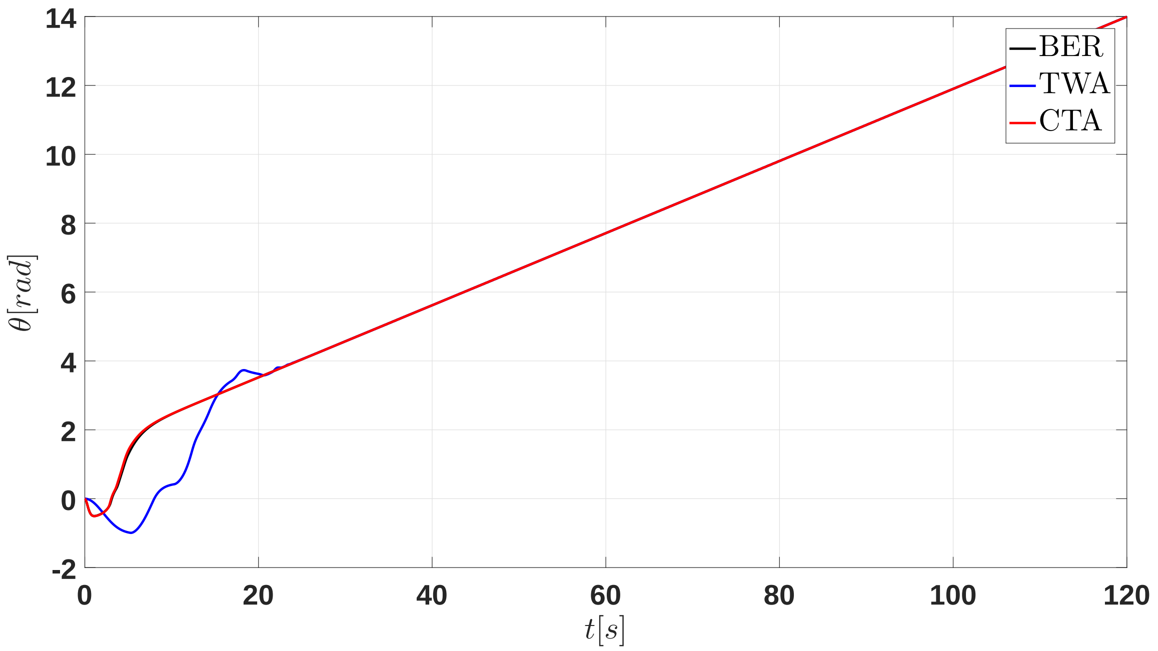


Figure 4.4: Simulation: Tracking. Measured output signal

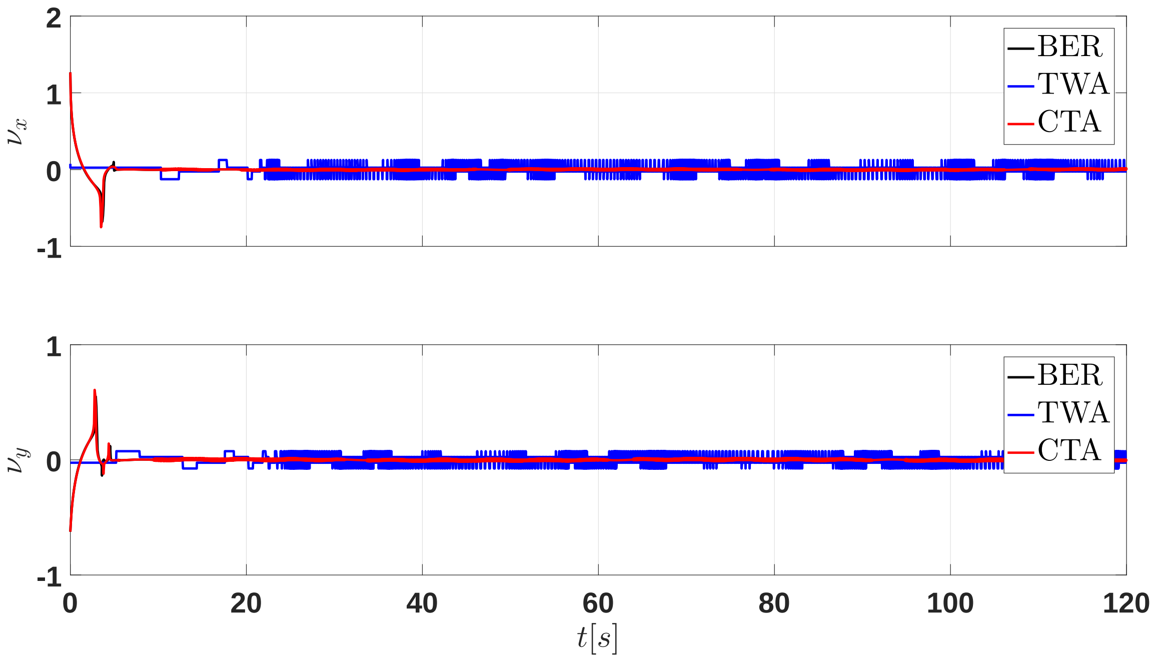


Figure 4.5: Simulation: Tracking. Control signals

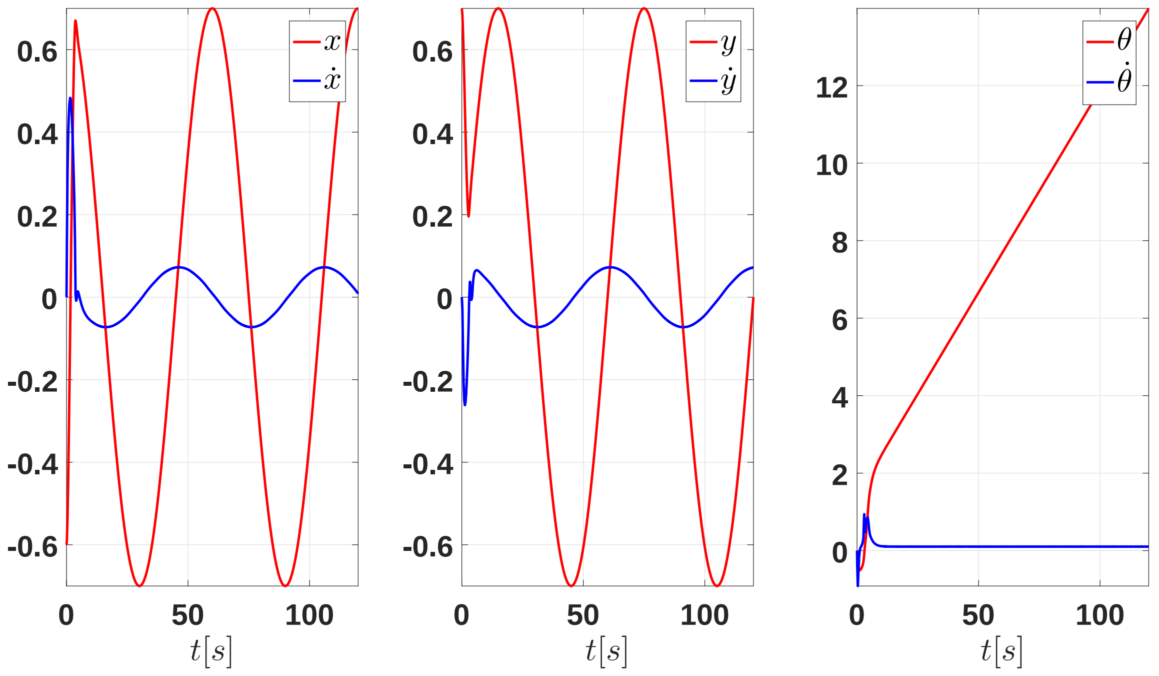


Figure 4.6: Simulation: Tracking. Estimated velocities signals

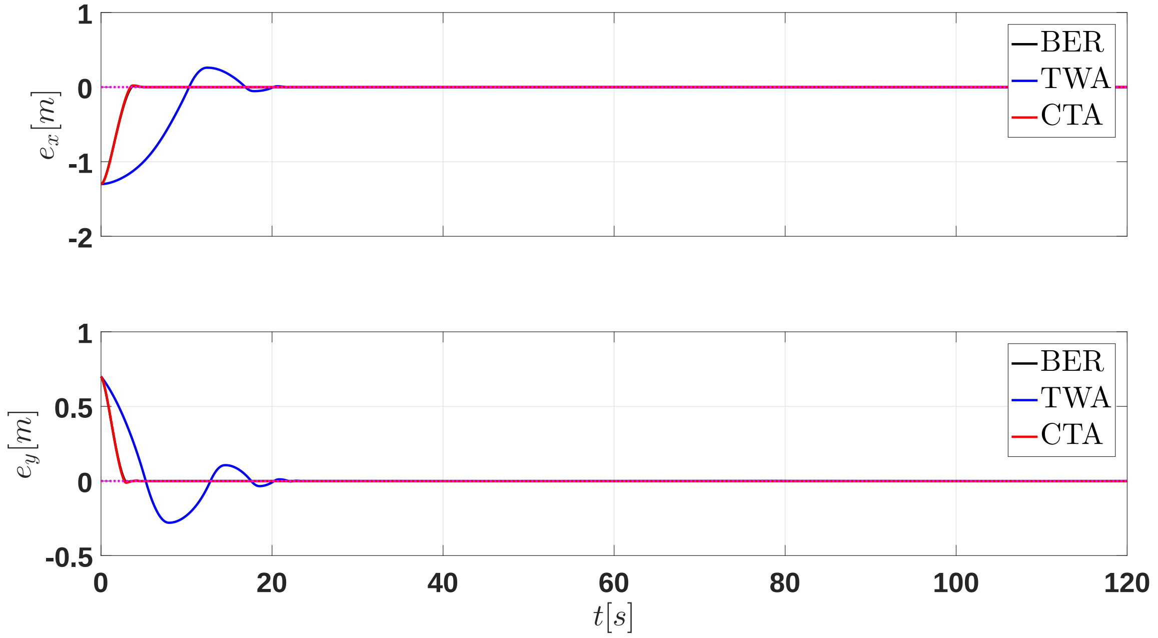


Figure 4.7: Simulation: Tracking. Error signals

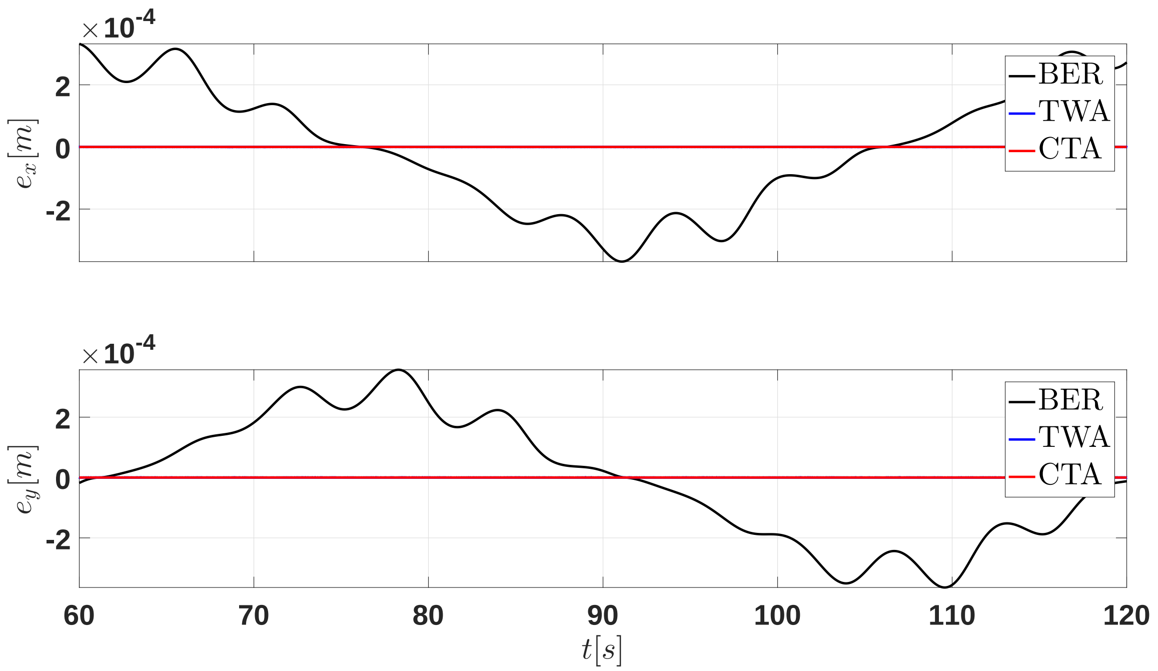
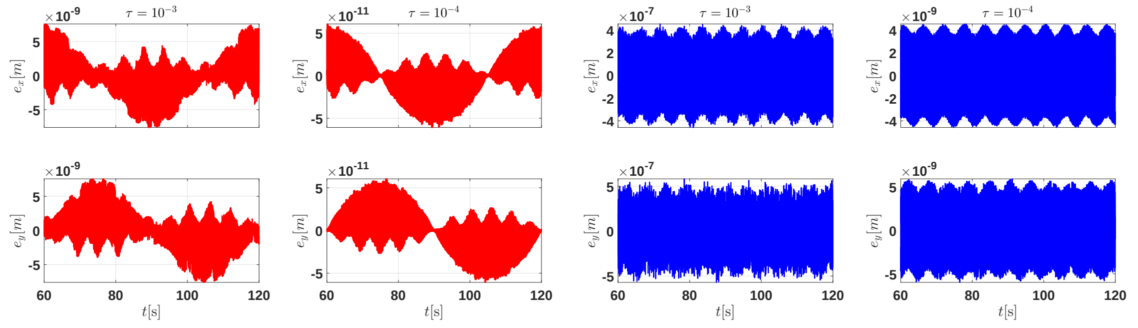


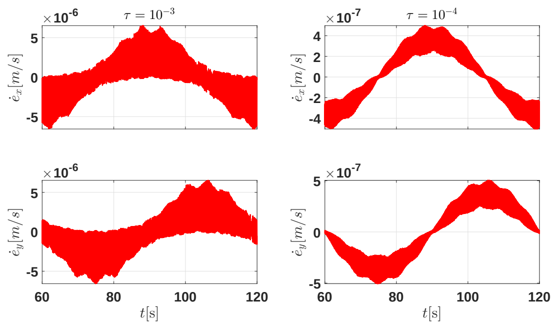
Figure 4.8: Simulation: Tracking. Error signals (After convergence)



(a) CTA

(b) TWA

Figure 4.9: Simulation: Tracking. Comparison of precision. Position errors.


(a) CTA

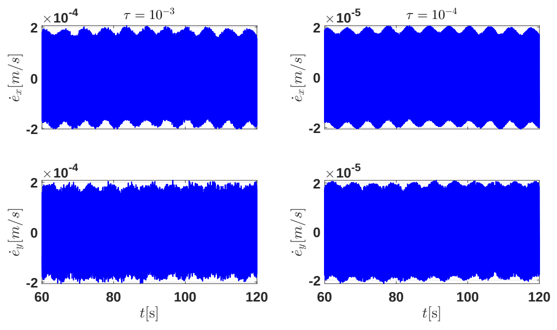

(b) TWA

Figure 4.10: Simulation: Tracking. Comparison of precision. Velocity errors.

Table 4.1: Simulation: Tracking. Precision Table

		$\tau = 1 \times 10^{-3}$	$\tau = 1 \times 10^{-4}$
CTA	e_1	$\tau^3 = 1 \times 10^{-9}$	$\tau^3 = 1 \times 10^{-12}$
	e_2	$\tau^2 = 1 \times 10^{-6}$	$\tau^2 = 1 \times 10^{-8}$
TWA	e_1	$\tau^2 = 1 \times 10^{-6}$	$\tau^2 = 1 \times 10^{-8}$
	e_2	$\tau^1 = 1 \times 10^{-3}$	$\tau^1 = 1 \times 10^{-4}$

4.4 Experiments

The reference trajectory is the one stated in Equation (4.34). An XY graph is presented to show the form of the trajectory in Figure 4.11. Trajectory tracking performance is shown in Figure 4.12 implementing control signals shown in Figure 4.17 and obtaining error signals shown in Figure 4.15. Figure 4.14 shows the result of the estimation using the discrete Kalman Filter from Equation (3.53).

For BER, e_x and e_y are bounded between approximately $|8|$cm because homogeneous algorithms without sliding modes cannot compensate exactly matched disturbances and cannot take the state to zero in finite time; this is shown in Figure 4.8. From Figure 4.8, CTA tends to have smaller errors on both coordinates compared to BER and TWA.

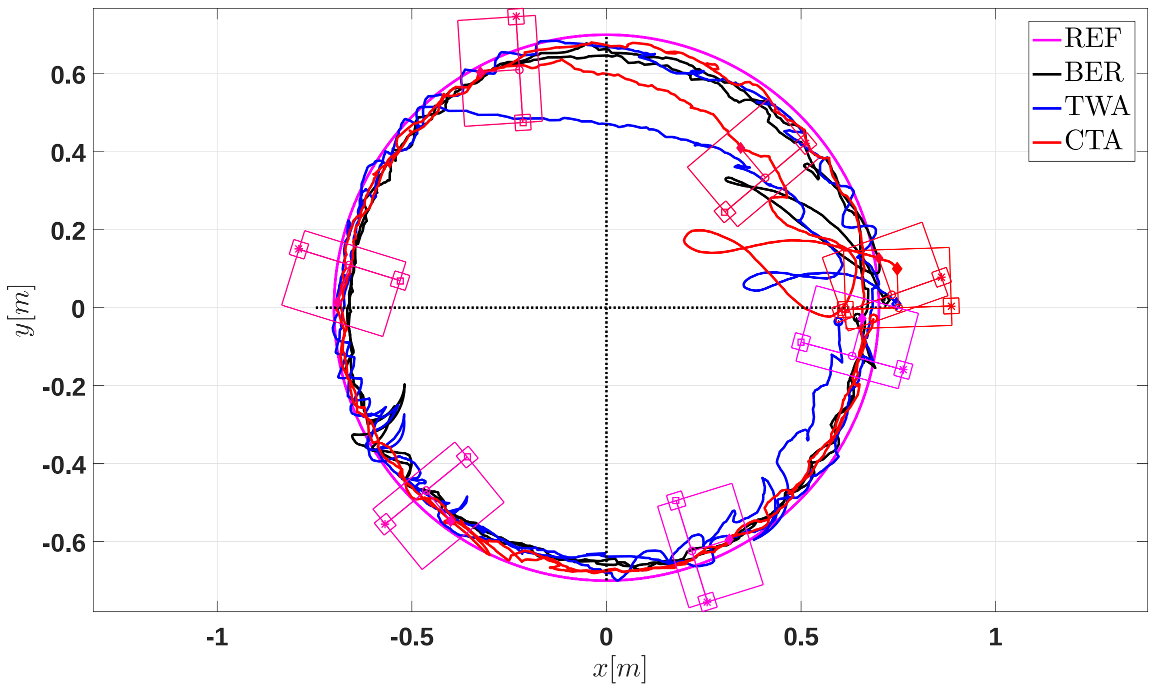


Figure 4.11: Experimental test. Tracking. XY plane. Controlled output signals

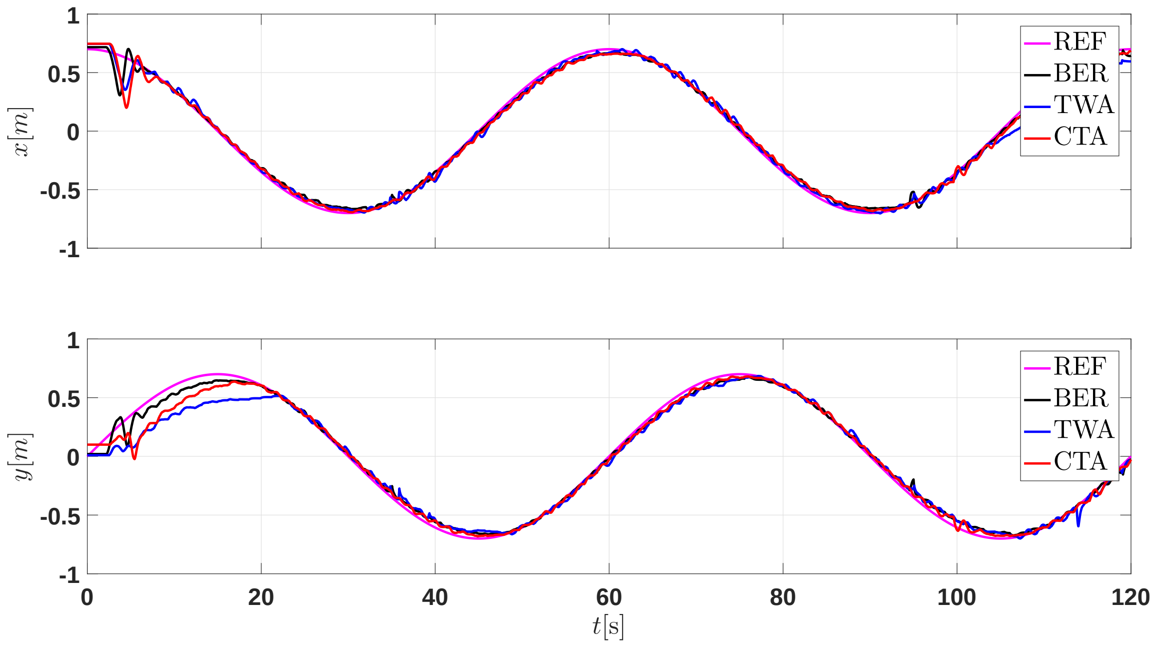


Figure 4.12: Experimental test. Tracking. Controlled output signals

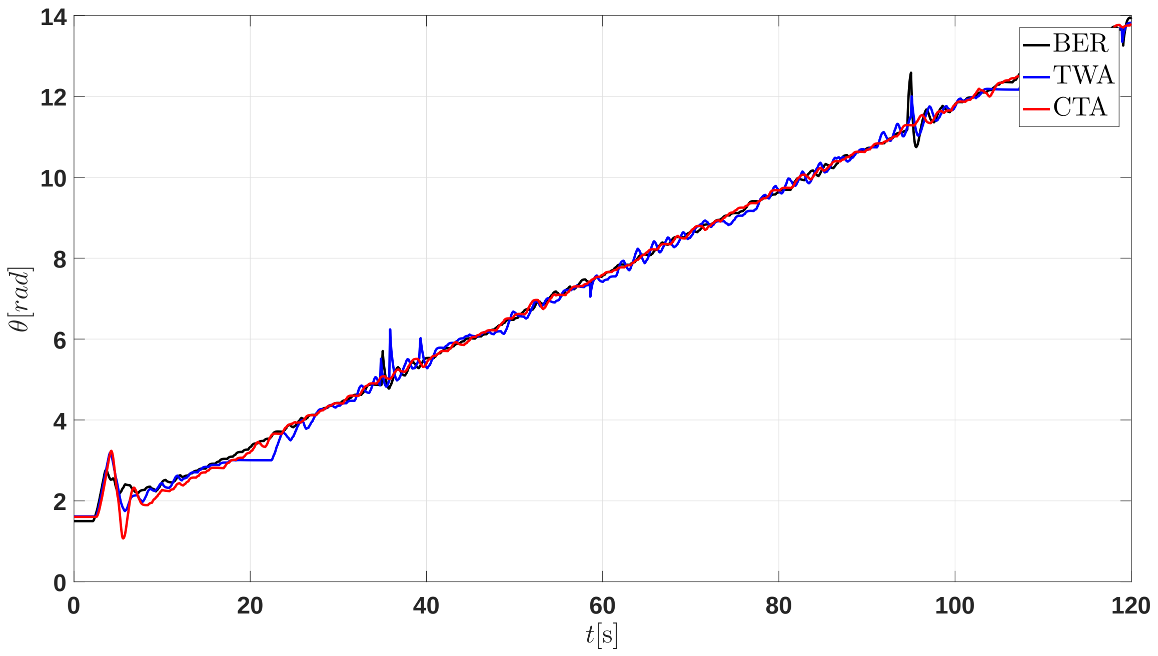


Figure 4.13: Experimental test. Tracking. Measured output signal

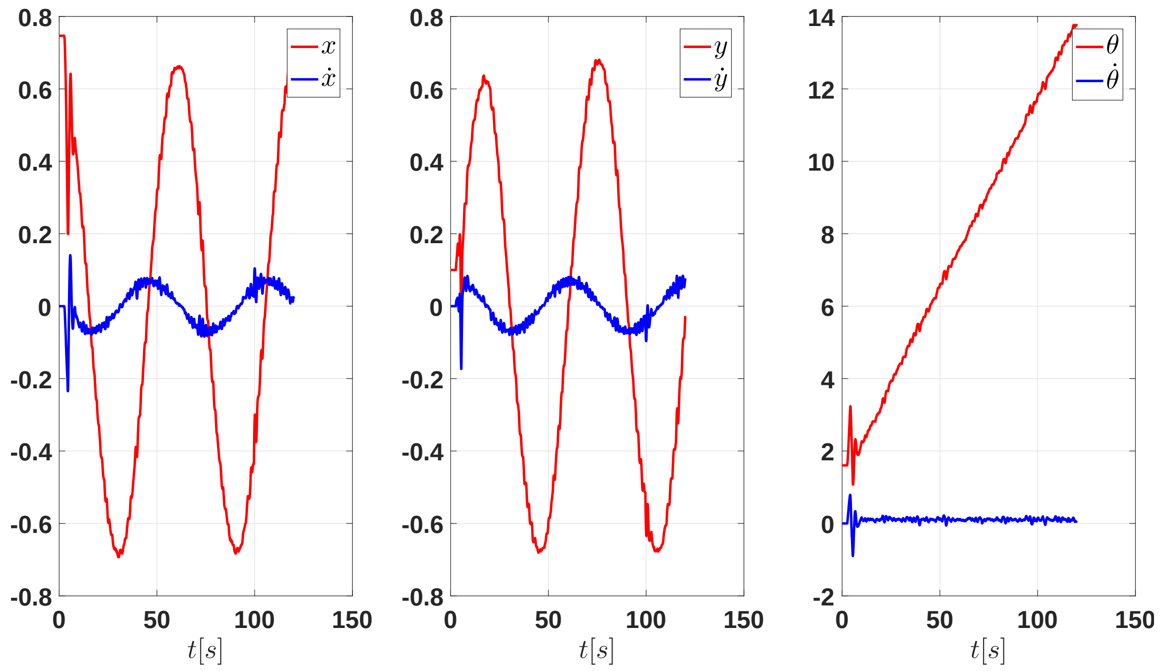


Figure 4.14: Experimental test. Tracking. Estimated velocity signals

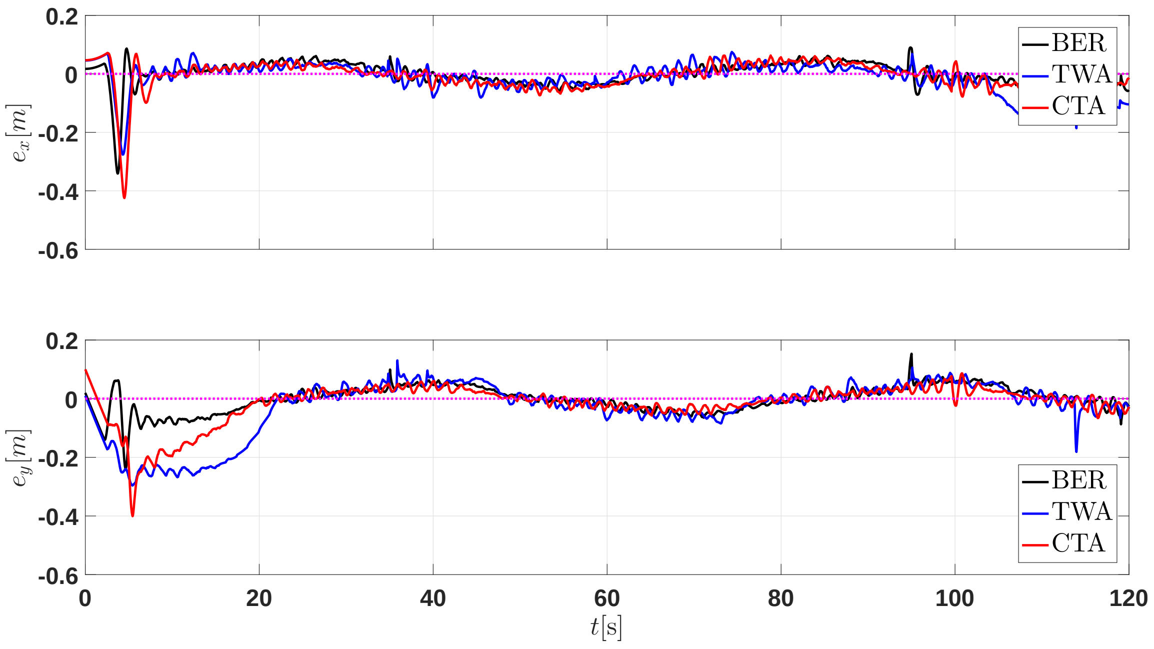


Figure 4.15: Experimental test. Tracking. Error signals

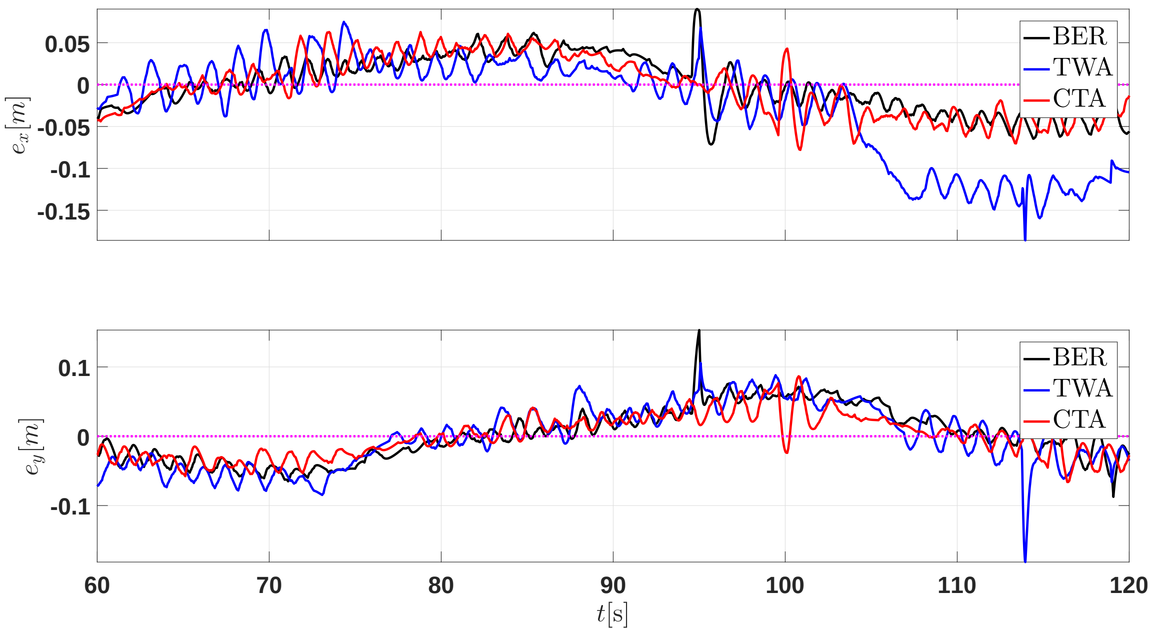


Figure 4.16: Experimental test. Tracking. Error signals (After convergence)

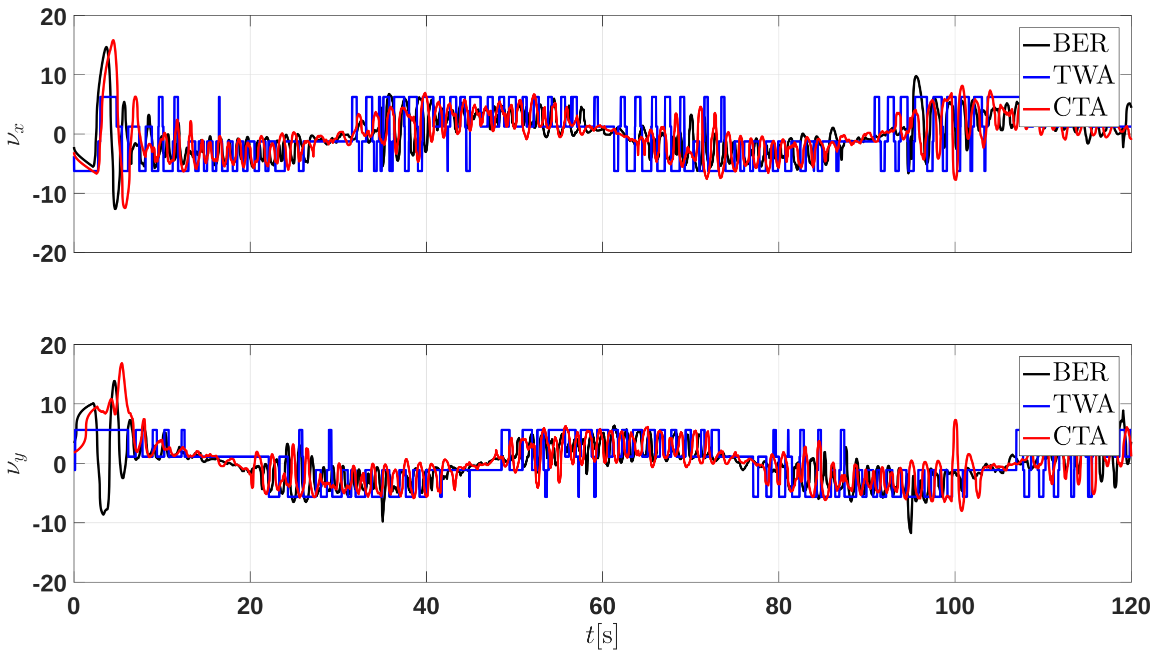


Figure 4.17: Experimental test. Tracking. Control signals

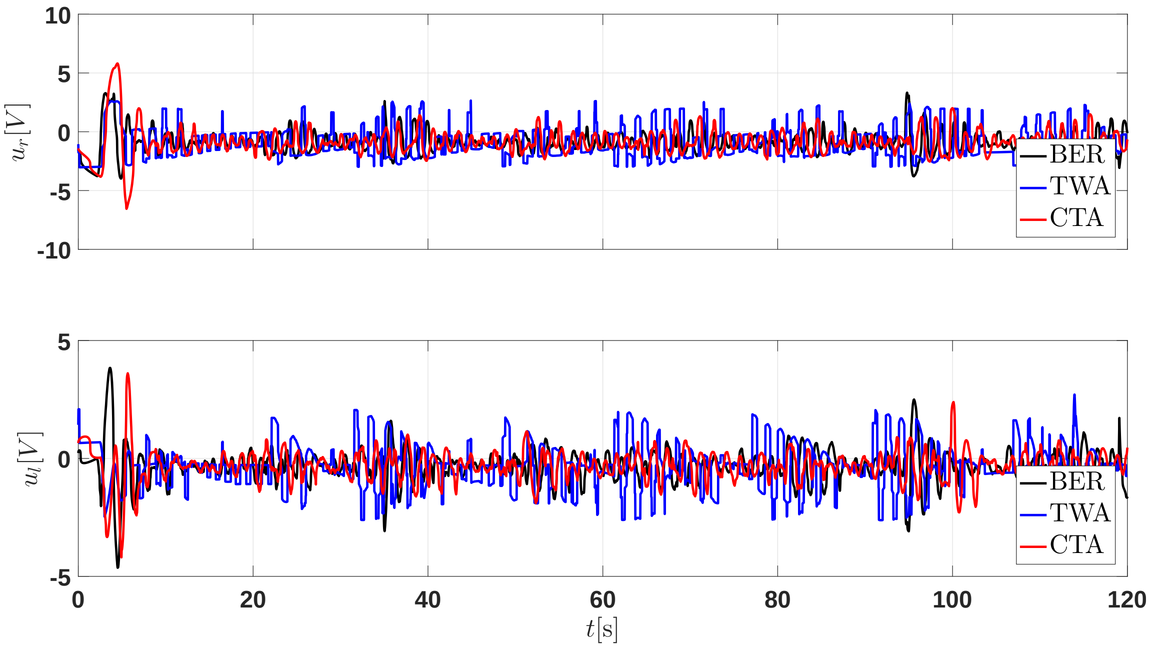


Figure 4.18: Experimental test. Tracking. Wheel control signals

Chapter 5

Conclusions and future work

5.1 Discussion

About the simulation and theoretical results there are some points to discuss.

First, simulations yield what is obtained from theory, as the sampling step decreases the precision of the sliding mode controller with respect to the error variables increases for both cases: relative degree 1 and 2.

Second, simulations yield that STA has the best performance because it has the smaller error measurements thanks to its capability of achieving a second-order sliding mode when using velocities as control inputs. Simulations yield that CTA has the best performance because it has the smaller error measurements thanks to its capability of achieving a third-order sliding mode when using voltages as control inputs.

Third, the discussion that arises is the comparison of STA and CTA. Simulations Figure 3.25a, Figure 4.9a and Figure 4.10a yield that extending the relative degree of Figure 3.11 does help improve the precision of the error measurements, as can be seen from comparing Table 3.1 with Table 4.1. Consequently, in simulations, CTA algorithm yields the best results of all the implemented algorithms in this thesis.

Fourth, comparison of simulation and experimental results yield interesting results. For relative degree 1, simulations from Figure 3.20 - Figure 3.25b compared to experimental tests from Figure 3.30 - Figure 3.36 yield very similar results with the exception of the bounds of the error measurements; for the experiments the bounds are bigger than the simulations. This is expected because in reality the dynamics of point $\boldsymbol{p}$ are far from a pure disturbed single integrator and, also, implemented control gains had to be scaled and adjusted for the experiments. However, control objective in practical terms is achieved satisfactorily. Similarly, for relative degree 2, simulations from Figure 4.3 - Figure 4.10 compared to experimental tests from Figure 4.11 - Figure 4.18 yield similar results with the exception of the bounds of the error measurements; for the experiments the bounds are bigger than the simulations.

A similar argument is made here because in reality the dynamics of point $\boldsymbol{p}$ are far from a pure disturbed double integrator and, also, implemented control gains had to be scaled and adjusted for the experiments. Moreover, velocity estimations are not as smooth as computed in the simulations and this fact affects directly the computed errors. However, control objective in practical terms is achieved satisfactorily.

Fifth, the last question that arises is the comparison of STA and CTA in the experimental tests. Comparing STA performances from Figure 3.34 and Figure 3.35 and CTA performances from Figure 4.15 and Figure 4.16 the error measurements are bigger for the CTA than for the STA. The veredict is that, in experiments, STA has a better performance than CTA in terms of error precision. This is directly related with the sampling rate used for the experiments. In order to match the simulation results it is required to adjust the sensor's sampling rate.

About the practical implementation there are some points to discuss.

First, about quantization and discretization phenomena that emerge from the tracking and detection process of the patterns that deliver posture information. After filtering Continuous Sliding Mode algorithms are able to achieve the control objective. When convergence is reached the error from the position measurements keep the robot from reaching a theoretical zero in the position errors. However, this behavior is within a tolerance for all the implemented algorithms and can be considered as a practical zero for sliding mode controllers.

Second, about the process to estimate velocity measurements from the position measurements in the controllers for relative degree 2. Thanks to the presence of the error position measurements, the velocity observer inherits this erroneous information and feeds the control algorithm with non accurate velocity information. Two velocity observers were considered: a Kalman Filter and a second order Sliding Mode differentiator. The best result of control design when the case of voltages as control inputs is considered was obtained using a Kalman Filter. In other words, the best velocity estimation resulted from using a Kalman Filter. This because eventhough an intensive tunning process was performed for the second-order SMD, the computed velocity signals were always considerably noisy w.r.t. the estimations of the position.

The last remark is about the sensor used, in this case the camera. Control decisions are based on the readings of the computer vision software, if the readings are not accurate control decisions cannot be accurate. This thesis shows that the methodology implemented works for real applications; however, it also shows that improving the readings precision will yield better performance results for the controllers.

5.2 Conclusions

Two models were considered to implement continuous Sliding Mode algorithms. First, a system with velocities as control inputs is considered. Hence, a system with position as outputs that has relative degree 1 is used. For this approach, Super Twisting Algorithm was implented and compared with a linear algorithm and a first-order sliding mode controller. Second, a system with voltages as control inputs is considered. Hence, a system with position as outputs that has relative degree 2 is used. For this approach, Continuous Twisting Algorithm was implemented and compared with an homogeneous algorithm without sliding modes and a Twisting controller. Simulations and experimental tests were shown to analyze the controllers performances.

5.3 Future Work

The main drawback from controlling x and y coordinates for a differential drive robot, in either of the two approaches, is that it is difficult to determine what happens with the transitory response of the zero dynamics of the system. Depending on the trajectory, when convergence of the continuous Sliding Mode algorithm is reached some observations can be done for that specific trajectory. This fact generates the question of whether x and y coordinates are the best choice of variables to be controlled in this system. Perhaps the choice of one of the XY plane coordinates and the orientation can overcome this drawback.

Nevertheless, controlling XY plane coordinates make possible to implement consensus and formation continuous Sliding Mode algorithms directly to differential drive robots. This because most of the consensus and formation problems in literature are designed for holonomic masses that are described only by x and y coordinates.

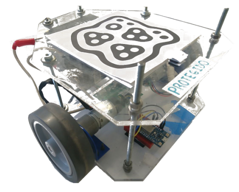
Another important assumption made in this thesis was that the area in which the robot moves is an obstacle-free environment and, therefore, one can establish predefined reference trajectories. This in general is not true, especially when humans could potentially interact with the robot. In industrial applications, the safety of the people present should always have priority over the integrity of the robot and this case is not addressed using the presented methodology. Adding reactive behaviors to the robot can solve this problem and still guarantee the fullfillment of its task.

Appendix A

Experimental setup

A.1 Overview

The experimental setup consists of a DDR, shown in Figure A.1a, a Microsoft Lifecam Studio camera, shown in Figure A.1b, and a computer with MATLAB/Simulink.



(a) Differential Drive Robot

(b) Camera

Figure A.1: Hardware elements

Reactvision is an application that performs detection and tracking of patterns called Fiducials. It is used to obtain point $\boldsymbol{a}$ and θ of the DDR. This information is recovered using Reactivision libraries in an auxiliary C++ program and then sent to MATLAB/Simulink using UDP protocol.

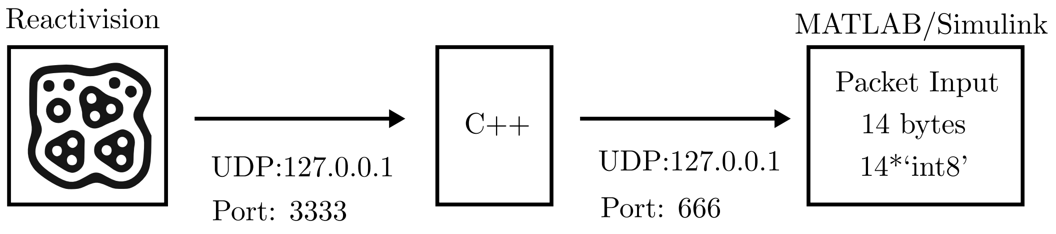


Figure A.2: Data transmission

Once the control algorithm is computed in MATLAB/Simulink, the two wheel actions are sent to the DDR using UDP protocol.

A.2 Hardware

The DDR's parameters are shown in Table A.1 and its hardware components are shown in Figure A.3.

Table A.1: DDR parameters

r	d	h
5cm	27.5cm	10cm

(a) Actuators

(b) Battery

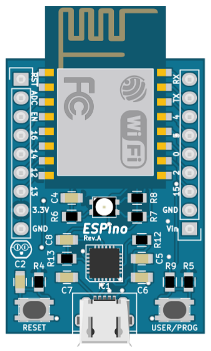

(c) Processor

Figure A.3: Hardware in the DDR

A.2.1 Actuators and power stage

The microcontroller communicates via I2C with a motor drive called MD25 which can drive two EMG30 motors independently. Each EMG30 motor is coupled to a 5 cm radius wheel with the help of a wheel-hub, as shown in Figure A.3a.
The MD25 has three turn modes that allow the user to define the format values of the wheel commands. In this thesis, the default configuration is used which is mode 0. In this mode only allows commands with format of unsigned 8-bit integers, i.e., only values in the range of 0 to 255; half of the spectrum is one turn direction and the other is the counter direction. This mode is illustrated in Figure A.4.

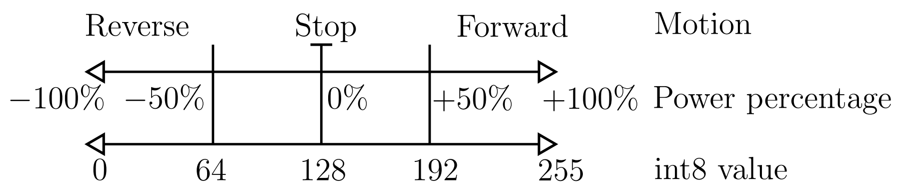


Figure A.4: MD25 turn wheel commands

For Section 3.5, the process to tranform from control inputs u_m to wheel commands W_m is shown in Figure A.5. First, control inputs u_m in rad/s, with m referring to left or right motor, are converted to revolutions per minute W_m as follows

$$W_m = \left[u_m\right]\frac{60}{2\pi}.$$

Then, if it is necessary W_m is saturated to $\left[-W_{\mathrm{SAT}}, +W_{\mathrm{SAT}}\right]$ with $W_{\mathrm{SAT}} = 100$. This means that for the experiments only half of the total capacity of the motors is used. Finally, to convert wheel control actions W_m to wheel commands PWM_m the following is implemented

$$\mathrm{PWM}_m = \begin{cases} 128 - \frac{128\,W_m}{200} & \text{if} \quad 0 \geq W_m \leq +W_{m\mathrm{SAT}}, \\ 128 + \frac{127\,W_m}{200} & \text{if} \; -W_{m\mathrm{SAT}} \geq W_m \leq 0. \end{cases} \tag{A.1}$$

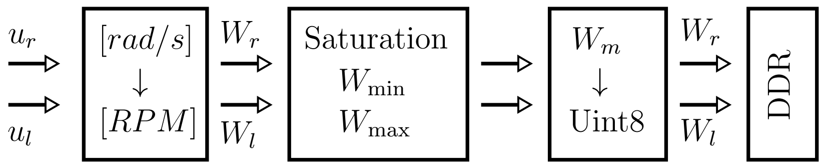


Figure A.5: Conversion to actuators for STA

For Section 4.4, control inputs u_m are in V, hence, after a rescaling only Equation (A.1) is needed.

A.2.2 Energy

Energy is provided to the DDR by a sealed lead-acid battery that delivers 12 V DC and has 4 Ah capacity, shown in Figure A.3b. The 12V battery energizes the MD25 and the latter energizes the microcontroller with its Vcc and GND pins.

A.2.3 Processor

ESPino, shown in Figure A.3c, is a 32-bit microcontroller based on ESP8266 chip. As many other microcontrollers it has I/O digital and an analog input pin. It also has SPI, serial and I2C communication pins. However, its main advantage is that ESP8266 has WI-fI communication integrated which is a wireless communication standard.

The ESPino connects to an specific WI-FI hotspot created by the computer with a SSID and a password that are predefined in the microcontroller code. A nonreserved port in the range of 0 to 2^{16} is also predefined in the ESPino code which allows bidirectional communication between a computer and the ESPino. In this thesis, communication from the computer to the ESPino is the only direction needed and the predifined port is 8888. Once the connection is made succesfully, a dynamic IP is assigned to the microcontroller. The generated IP and the predefined port form the address to which UDP packets will be sent from MATLAB/Simulink to make the DDR move its wheels.

The WI-FI hotspot to which the ESPino connects is created with the help of a program that allows to create a virtual router with the computer. In this thesis, the WI-FI hotstop SSID is “RoboticaMovil1” and password is “1234567890”. This program also displays the assigned IP of the ESPino.

A.3 Software

A.3.1 Reactivision

Reactivision is an open source standalone application that performs detection and tracking of fiducial markers like the one shown in Figure A.2. Attaching these markers to physical objects makes tracking of one or more objects possible. Reactivision sends TUIO messages using UDP protocol to IP address 127.0.0.1 with port 3333. These TUIO messages contain the detection and tracking information of the fiducial markers seen by the camera.

Cameras distorsion is generated by its oval lense. For this reason, a calibration process is needed before performing experiments. This process is done by adjusting the width and height calibration grid from Reactivision. For Reactivision 1.5.1 win64 version, first, show the calibration grid by pressing ‘c’ key in the camera view from Reactivision. Next, adjust the calibration grid to the work area by using a,d,w,x keys to move within the grid and arrow keys to modify the grid’s width and height.

A.3.2 C++ Library

Reactivision provides libraries coded in C++, Java and other programming languages to interact with the TUIO messages sent to port 3333. In this thesis, C++ library is chosen to optimize the speed of the transmission process. In addition, the C++ demo program “TuioDemo” provided by Reactivision’s support webpage automatically corrects the distorsion of the camera if the calibration process was performed beforehand.

For this thesis the information needed is the position and orientation of the fiducial marker which will be placed in point $\boldsymbol{a}$, i.e., the middle point of axis formed by the actuated wheels. A standard socket library is implemented to send via UDP protocol these 3 variables to local IP adresss 127.0.0.1 with port 666. The packet’s format is defined as follows

Table A.2: Transmision frame format

Data 1	Data 2	Data 3	Data 4	Data 5
‘M’	XPOS	YPOS	THETA	NUM
1 byte	4 bytes	4 bytes	4 bytes	1 byte

where ‘M’ is the identifier of the start of frame; XPOS, YPOS and THETA are float variables each one converted to a 4 byte unsigned integer. The whole frame data type is an array of unsigned 8-bit integers; NUM is an integer variable that stores the number of fiducials detected by Reactivision and can be used to detect reading errors whenever the camera fails to detect the fiducial marker. Therefore, the UDP packages sent to MATLAB/Simulink will be nBYTES = 14 bytes long.

A.3.3 MATLAB/Simulink

Receiving data

The data accquisition process in MATLAB/Simulink starts with receiving the packet sent from the socket in C++ to port 666 using Simulink’s Packet input block. Since UDP protocol is used to transmit the data the content of the package may arrive in disorder. An example of this is shown in Figure A.6. Therefore, a sort operation needs to preprocess the incoming byte array using the start of frame identifier.

The start of frame identifier is chosen to be character ‘M’ according to Table A.2 which in ASCII code corresponds to 77DEC. First, the start of frame identifier index AUX is found within the incoming byte array. Second, the following (nBYTES − AUX) bytes of the incoming array are actually the first part of the new sorted array and corresponds to Part I in Figure A.6. Third, the first AUX bytes of the incoming array are the rest of the new sorted array are and they correspond to Part II in Figure A.6.

After the ordering process, the 3 variables packed in 4 byte variables are converted back to 32-bit single precision floating-point format. To be able to use these three variables in Simulink, they have to be converted to double-precision floating-point format. Here, reformatting process is completed and the three double-type variables ($\boldsymbol{a}_{\text{raw}nc}$ and $\theta_{\text{raw}pd}$) are ready to perform computations on them.

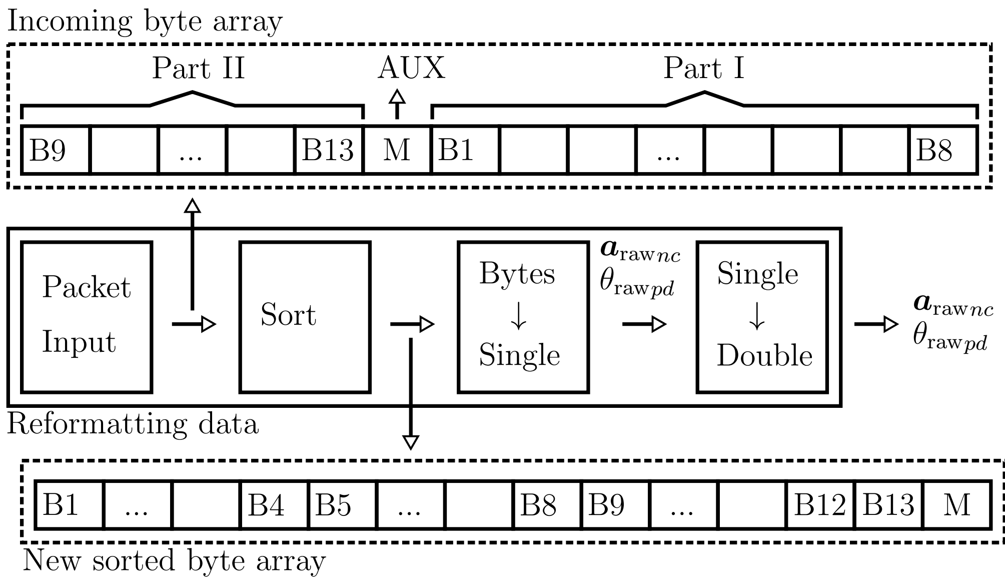


Figure A.6: DAQ: Reformatting

Point $\boldsymbol{a}_{\text{raw}nc}$ needs to be centered using the following translation w.r.t. to the working area

$$\boldsymbol{a}_{\text{raw}} = \boldsymbol{a}_{\text{raw}nc} - \begin{bmatrix} h_a \\ k_a \end{bmatrix},$$

where h_a and k_a will vary depending on the calibration grid. For this thesis, $h_a = 318$ and $k_a = 225$.

$\theta_{\text{raw}pd}$ signal has a range of 0 to 2π. Figure A.7 illustrates the behavior of $\theta_{\text{raw}pd}$ when a sequence of rotations is performed staring from 0 radians. First, the DDR rotates 2π radians counterclockwise. Second, rotation resets and the DDR rotates in the same direction π radians more. Third, the DDR changes rotation direction and turns $-\pi$ radians. Fourth, orientation resets and the DDR keeps the previous rotation direction turning -2π radians. Finally, orientation resets and the DDR turns another -2π radians maintaining the previous direction. For this thesis, there is a need to avoid the mentioned resets and keep track of the rotation direction obtaining an orientation signal θ_{raw} without phase discontinuities. In other words, $\theta_{\text{raw}pd}$ needs to expand its range to $\mathbb{R}$. The solution is to keep track of the turning direction and detect phase discontinuities larger than π radians. The latter operation can be performed by applying Simulink's Unwrap block with π tolerance to signal $\theta_{\text{raw}pd}$ which results in the expected signal θ_{raw} which is shown in Figure A.7.

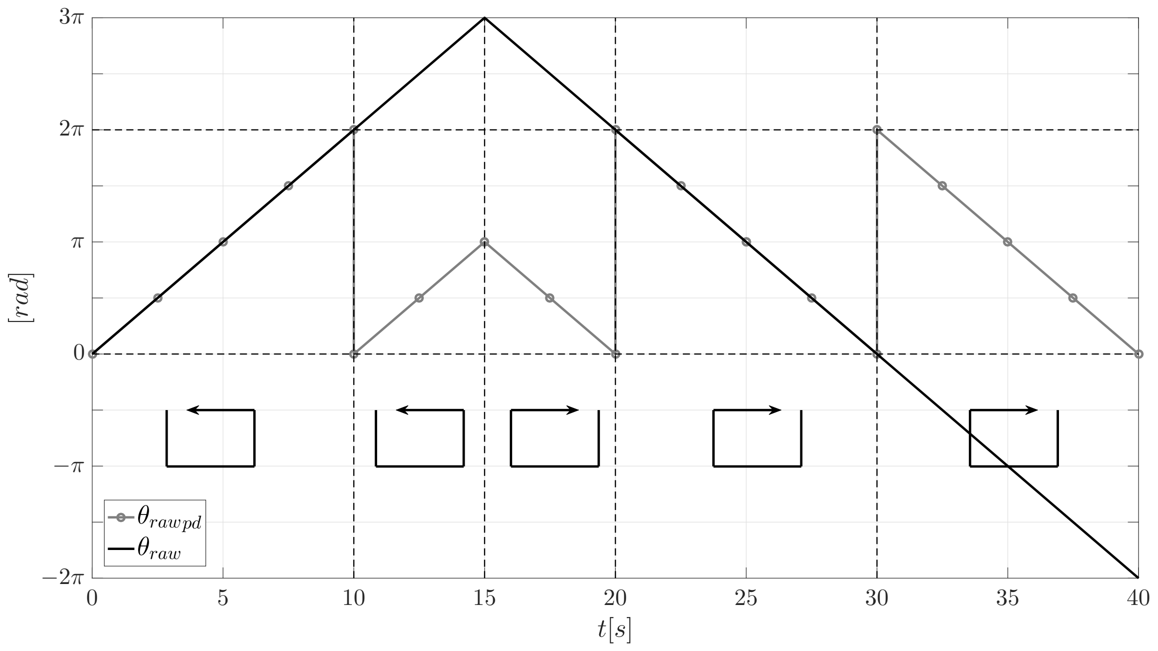
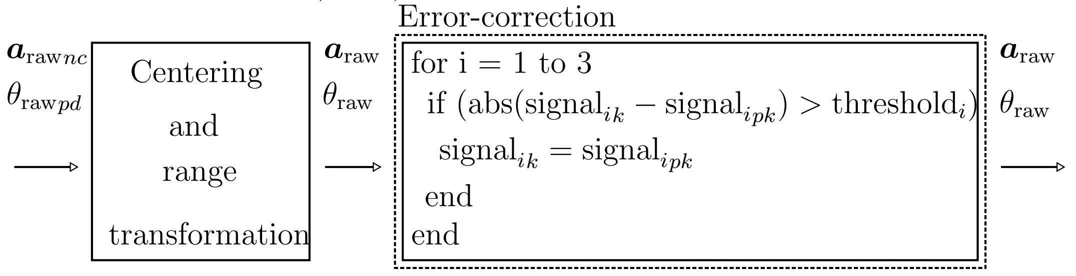


Figure A.7: Transforming θ_{raw}

Due to the use of UDP protocol and its inherent loss of information it is strongly suggested that at least one error-correction method to signals $\boldsymbol{a}_{\text{raw}}$ and θ_{raw} is implemented before computing control algorithms. When position and orientation of the fiducial are lost, the values of the signals from the previous sample step differ greatly from the values of the signals from the current sample step. This difference can be detected by using a threshold variable for each signal. If the threshold condition is broken then for some reason bad measurements are being received and, therefore, the previous value of the signal should replace the current value of the signal. If good measurements are received the threshold condition is not broken and they can be used for the control algorithms computations. This method will fill the gap derived from the loss of information and also remove outliers from the signals as long as the DDR does not behave erraticly in the sense that constant signals are generated due to this erratic behavior. Another failure phenomenon that could occur for this method is the generation of constant signals due to bad measurements in many consecutive sample steps. Error-correction pseudocode is shown in Figure A.8 and whith this method data acquistion (DAQ) is finished.


Error-correction
$\boldsymbol{a}_{\text{raw}nc}$
$\theta_{\text{raw}pd}$
Centering and range transformation
$\boldsymbol{a}_{\text{raw}}$
θ_{raw}
$\boldsymbol{a}_{\text{raw}}$
θ_{raw}

```
for i = 1 to 3
  if (abs(signal$_{ik}$ − signal$_{ipk}$) > threshold$_i$)
    signal$_{ik}$ = signal$_{ipk}$
  end
end
```

Figure A.8: DAQ: Error-Correction

As mentioned in Section 3.5, $\boldsymbol{a}_{\text{raw}}$ and θ_{raw} has quantization and discretization phenomena, hence, these signals need to go through a filtering process. For Section 3.5, filtered signals $\boldsymbol{a}$ and θ are used to compute the information for point $\boldsymbol{p}$. Finally, $\boldsymbol{p}$ and θ are used to compute STA. This process is illustrated in Figure A.9.

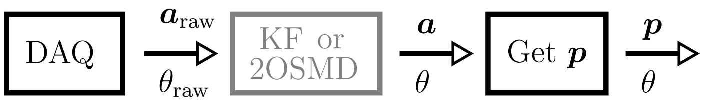


Figure A.9: DAQ and filtering for STA

For Section 4.4, filtered and velocity estimations signals, $\boldsymbol{a}$, θ and $\dot{\boldsymbol{a}}$, $\dot{\theta}$, respectively, are used to compute point $\boldsymbol{p}$ and $\dot{\boldsymbol{p}}$. Finally, $\boldsymbol{p}$, θ and $\dot{\boldsymbol{p}}$, $\dot{\theta}$ are used to compute CTA. This process is illustrated in Figure A.10.

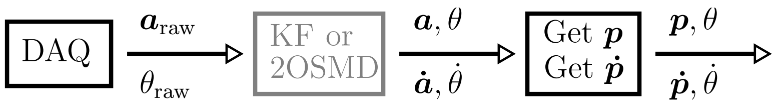

DAQ
$\boldsymbol{a}_{\text{raw}}$
θ_{raw}
KF or 2OSMD
$\boldsymbol{a}, \theta$
$\dot{\boldsymbol{a}}, \dot{\theta}$
Get $\boldsymbol{p}$
Get $\dot{\boldsymbol{p}}$
$\boldsymbol{p}, \theta$
$\dot{\boldsymbol{p}}, \dot{\theta}$

Figure A.10: DAQ, filtering and velocity estimations for CTA

Sending data

After transforming the control wheel actions to the MD25 wheel commands, the wheel commands are sent via Simulink's Packet Output Block. The packet is formed with the two wheel commands which are 2 8-bit unsigned integers. Hence, the packet sent to the DDR's IP, shown by the WI-FI hotspot program, and predefined port 8888 is 2 bytes long.

References

Bhat, S. P. and Bernstein, D. S. (2000). Finite – time stability of continuous autonomous systems. *SIAM Journal on Control and Optimization*, 38(3):751–766.

Boiko, I., Fridman, L., and Castellanos, M. (2004). Analysis of second-order sliding-mode algorithms in the frequency domain. *IEEE Transactions on Automatic Control*, 49(6):946–950.

Carlisle, B. (2000). Robot mechanisms. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 701–708. IEEE.

d'Andrea Novel, B., Bastin, G., and Campion, G. (1992). Dynamic feedback linearization of nonholonomic wheeled mobile robots. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2527–2532 vol.3.

Davila, J. (2013). Exact tracking using backstepping control design and high-order sliding modes. *IEEE Transactions on Automatic Control*, 58(8):2077–2081.

de Wit, C. C., Siciliano, B., and Bastin, G. (2012). *Theory of robot control.* Springer Science & Business Media.

Diaz, D. and Kelly, R. (2016). On modeling and position tracking control of the generalized differential driven wheeled mobile robot. In *2016 IEEE International Conference on Automatica (ICA-ACCA)*, pages 1–6.

Efimov, D. and Perruquetti, W. (2016). On conditions of oscillations and multi–homogeneity. *Mathematics of Control, Signals, and Systems*, 28(1):3.

Grewal, M. S. (2011). *Kalman Filtering*, pages 705–708. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hirschorn, R. M. (2002). Output tracking through singularities. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, volume 4, pages 3843–3848 vol.4.

Isidori, A. (2013). *Nonlinear control systems.* Springer Science & Business Media.

Kamal, S., Moreno, J. A., Chalanga, A., Bandyopadhyay, B., and Fridman, L. M. (2016). Continuous terminal sliding-mode controller. *Automatica*, 69:308–314.

Khalil, H. (2002). *Nonlinear Systems*. Pearson Education. Prentice Hall.

Levant, A. (1998). Robust exact differentiation via sliding mode technique. *automatica*, 34(3):379–384.

Levant, A. (2003). Introduction to high-order sliding modes. *School of Mathematical Sciences, Israel*, 58(6):1.

Moreno, J. A. (2009). A linear framework for the robust stability analysis of a generalized super-twisting algorithm. In *2009 6th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pages 1–6.

Moreno, J. A., Negrete, D. Y., Torres-González, V., and Fridman, L. (2016). Adaptive continuous twisting algorithm. *International Journal of Control*, 89(9):1798–1806.

Mu, J., Yan, X. G., Spurgeon, S. K., and Mao, Z. (2015). Trajectory tracking control of a two-wheeled mobile robot using sliding mode techniques. In *2015 34th Chinese Control Conference (CCC)*, pages 3307–3312.

Oriolo, G., Luca, A. D., and Vendittelli, M. (2002). Wmr control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6):835–852.

Orlov, Y., Aoustin, Y., and Chevallereau, C. (2011). Finite time stabilization of a perturbed double integrator – part i: Continuous sliding mode-based output feedback synthesis. *IEEE Transactions on Automatic Control*, 56(3):614–618.

Reid, I. (2001). Estimation ii. `www.robots.ox.ac.uk/~ian/Teaching/Estimation/LectureNotes2.pdf`.

Ríos, H., Efimov, D., Polyakov, A., and Perruquetti, W. (2016). Homogeneous time-varying systems: Robustness analysis. *IEEE Transactions on Automatic Control*, 61(12):4075–4080.

Rooks, B. (2001). Agvs find their way to greater flexibility. *Assembly Automation*, 21(1):38–43.

Sánchez, T. and Moreno, J. A. (2014). A constructive lyapunov function design method for a class of homogeneous systems. In *53rd IEEE Conference on Decision and Control*, pages 5500–5505.

Sastry, S. and Bodson, M. (2011). *Adaptive control: stability, convergence and robustness*. Courier Corporation.

Shtessel, Y., Edwards, C., Fridman, L., and Levant, A. (2014). *Sliding mode control and observation*. Springer.

Solea, R. and Cernega, D. (2015). Super twisting sliding mode controller applied to a nonholonomic mobile robot. In *2015 19th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 87–92.

Torres-González, V., Fridman, L. M., and Moreno, J. A. (2015). Continuous twisting algorithm. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 5397–5401. IEEE.