

9-2011

Nonholonomic Feedback Control Among Moving Obstacles

Stephen Gregory Armstrong

Embry-Riddle Aeronautical University - Daytona Beach

Follow this and additional works at: <https://commons.erau.edu/edt>



Part of the [Automotive Engineering Commons](#)

Scholarly Commons Citation

Armstrong, Stephen Gregory, "Nonholonomic Feedback Control Among Moving Obstacles" (2011).
Dissertations and Theses. 17.

<https://commons.erau.edu/edt/17>

This Thesis - Open Access is brought to you for free and open access by Scholarly Commons. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of Scholarly Commons. For more information, please contact commons@erau.edu.

NONHOLONOMIC FEEDBACK CONTROL AMONG MOVING OBSTACLES

By
Stephen Gregory Armstrong

A Thesis Submitted to the
Physical Sciences Department
In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering Physics

Embry-Riddle Aeronautical University
Daytona Beach, Florida
September 2011

© Copyright by Stephen Gregory Armstrong 2011
All Rights Reserved

Nonholonomic Feedback Control Among Moving Obstacles

by

Stephen Gregory Armstrong

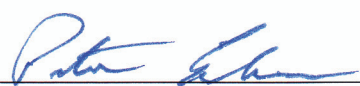
This thesis was prepared under the direction of the candidate's thesis committee chair, Dr. Sergey V. Drakunov, Department of Physical Sciences, and has been approved by the members of the thesis committee. It was submitted to the Department of Physical Sciences and was accepted in partial fulfillment of the requirements of the Degree of Master of Science in Engineering Physics

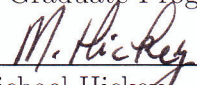
THESIS COMMITTEE:


Dr. Sergey V. Drakunov, Chair


Dr. Mahmut Reyhanoglu, Member


Dr. Bereket Berhane, Member


Dr. Peter Erdman
MSEP Graduate Program Coordinator


Dr. Michael Hickey
Department Chair, Physical Sciences


Dr. Robert Oxley
Associate Vice President for Academics


Date

Abstract

A feedback controller is developed for navigating a nonholonomic vehicle in an area with multiple stationary and possibly moving obstacles. Among other applications the developed algorithms can be used for automatic parking of a passenger car in a parking lot with complex configuration or a ground robot in cluttered environment. Several approaches are explored which combine nonholonomic systems control based on sliding modes and potential field methods.

Acknowledgments

I would like to thank my advisor, Dr. Drakunov for more than two years of help and guidance. I also have benefited from frequent advice from my other committee members, Dr. Berhane, and Dr. Reyhanoglu. I also would like to thank my committee members for their efforts in reviewing the draft copy of this thesis. Donna Fremont, and Susan Adams helped me schedule and reschedule my defence. I would like to thank the Physical Science Department for employing me while I did my studies and, in the last few months, while I have been writing this. To my parents, thank you so much for all of your support over the last seven years as I was a student. I will always be grateful. I used an edited version of a latex style file written by a previous graduate student, Nathan Haluska. I am grateful to my wonderful girlfriend, Trang Ta for keeping me focused and cooking nearly every meal that I have eaten over the last four months. I would also like to thank my friends for staying my friends, even though I have nearly ignored them since January.

Contents

List of Figures	v
1 Problem Description	1
1.1 Problem Statement	1
1.2 Research Scope	2
1.3 Possible Applications	3
2 Theory of Systems with Nonholonomic Constraints	4
2.1 Nonholonomic Constraints and Systems	4
2.1.1 Informal Introduction with Examples	4
2.1.2 Formal Definition	5
2.1.3 Mathematical Preliminaries	6
2.1.4 Degrees of Nonholonomy	9
2.1.5 Brockett's Theorem	10
2.2 Examples of Nonholonomic Systems	12
2.2.1 Nonholonomic Vehicle Model	12
2.2.2 Extension to N Trailers	14
3 Nonholonomic Control Systems	17
3.1 Nonholonomic Path Planning with Obstacles	17
3.2 Nonholonomic Feedback Control	20
3.2.1 The Heisenberg System	20
3.2.2 Bloch-Drakunov Controller	24
3.3 Feedback Control for Systems with Obstacles	38

3.3.1	Tracking	38
3.3.2	Example: Tracking in the Heisenberg System	39
4	Optimal Control	43
4.1	Pontryagin's Maximum Principle	43
4.1.1	The General Case	43
4.1.2	The Time-Optimal Case	47
4.2	Applying Maximum Principle in Configuration Space	48
4.3	Applying Maximum Principle in Heisenberg Space	50
4.3.1	Control Derivation	50
4.3.2	Optimal Control as Partial Feedback	51
5	Variable Structure Feedback Control for Nonholonomic Systems	54
5.1	Chained-Form Controller, Example: The Car	55
5.2	Hybrid Chain-form and Bloch-Drakunov type Heisenberg System Controller	58
5.3	Simulink Model of Altered Controller	61
6	Potential Field Methods and Their Applications to Nonholonomic Systems	69
6.1	Potential Field Method for Obstacle Avoidance in Robotic Systems	69
6.2	Potential Field Method for Obstacle Avoidance in Systems Without Nonholonomic Constraints	70
6.2.1	Simulated Gradient of 2D Potential Field with Image Processing	71
6.3	Transforming Potentials into Heisenberg Space with Paraboloidal Sliding Surface	73
6.4	Wrapping a Potential Field on a Sliding Manifold Using Geodesic Distance	74
6.5	Including Potential in the Lyapunov Function for Obstacle Avoidance	78
6.6	Combined Potential Fields and Sliding Modes	81
6.7	Combined Potential Fields and Sliding Modes (A New Approach)	82

7	Conclusions	84
8	Matlab Code	85
8.1	The Main File	85
8.2	Animating the Car to Demonstrate Simulink Models	89
8.3	Image Processing and Potential Field Functions	91
8.3.1	Image Processing to Find Obstacles	91
8.3.2	Placing Charges on Obstacles and Goal Position	92
8.3.3	Creating the Potential Field	93
8.4	Transformation to Heisenberg Space	94
8.4.1	Putting Obstacles in Heisenberg Space	94
8.4.2	Adding in a Sample Paraboloid	96
	Bibliography	97

List of Figures

1.1	Nonholonomic vehicle in a configuration space with moving obstacles	2
1.2	Combining nonholonomic constraints and moving obstacles	3
2.1	Lie bracket motion	7
2.2	Single car	12
2.3	Car with N-trailers Murray and Sastry [1993]	15
3.1	The two possible shapes of Dubins Curves. LaValle [2006]	19
3.2	Classic nonholonomic car model	21
3.3	Bloch-Drakunov controller modeled with Simulink: main window . .	27
3.4	Bloch-Drakunov controller modeled with Simulink: state transformation	28
3.5	Bloch-Drakunov controller modeled with Simulink: variable structure control Algorithm	29
3.6	Bloch-Drakunov controller modeled with Simulink: switching condition	30
3.7	Bloch-Drakunov controller modeled with Simulink: controllers a & b .	31
3.8	Bloch-Drakunov controller modeled with Simulink: control transfor- mation	32
3.9	Vehicle model	33
3.10	Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (x and y convergence)	34
3.11	Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (ϕ convergence)	34

3.12	Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (x_1 , x_2 and x_3 convergence)	35
3.13	Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (u_1 and u_2 convergence)	35
3.14	Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (x and y convergence)	36
3.15	Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (ϕ convergence)	36
3.16	Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (x_1 , x_2 and x_3 convergence)	37
3.17	Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (u_1 and u_2 convergence)	37
3.18	Model of car stabilized by Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$	38
5.1	Geometric phase technique	55
5.2	Classic nonholonomic car model	56
5.3	Altered Bloch-Drakunov controller modeled with Simulink: main window	61
5.4	Altered Bloch-Drakunov controller modeled with Simulink: control algorithm	62
5.5	Altered Bloch-Drakunov controller modeled with Simulink: controllers a & b	63
5.6	Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (x and y convergence)	64
5.7	Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (ϕ convergence)	64
5.8	Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (x_1 , x_2 and x_3 convergence)	65
5.9	Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (u_1 and u_2 convergence)	65

5.10	Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (x and y convergence)	66
5.11	Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (ϕ convergence)	66
5.12	Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (x_1, x_2 and x_3 convergence)	67
5.13	Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (u_1 and u_2 convergence)	67
6.1	Producing a 2-dimensional potential field; the image processing steps	72
6.2	Simulated 2-D. obstacles transformed into Heisenberg space	73
6.3	Original $\frac{\beta}{2\alpha} (x_1^2 + x_2^2) = x_3 $ paraboloid drawing from Bloch and Drakunov [1996]	75
6.4	Wrapping potential field on to sliding manifold for feedback switching conditions	76

Chapter 1

Problem Description

1.1 Problem Statement

To accurately model the movement of most systems in nature, such as human walking and the rolling of a seed, one has to consider non-integrable velocity constraints. These types of constraints (known as nonholonomic constraints) are also prevalent in many manmade mechanical systems, including wheeled vehicles. The control of these inherently nonlinear systems has received much attention over the past twenty or so years, because many other systems in nature, for example quantum mechanical systems, demonstrate similar behavior.

Practical control problems often have both nonholonomic constraints (described more thoroughly in the next chapter), and holonomic constraints, (constraints on the configuration space).

The goal of this thesis is to:

Investigate robust methods of feedback control for nonholonomic systems, and to find a state dependent controller for a nonholonomic system, where certain areas of the state-space must be avoided. These areas may be time-dependant and in this case, they are represented as moving obstacles

Additional effort was made towards optimality.

The following cartoon illustrates the problem statement:

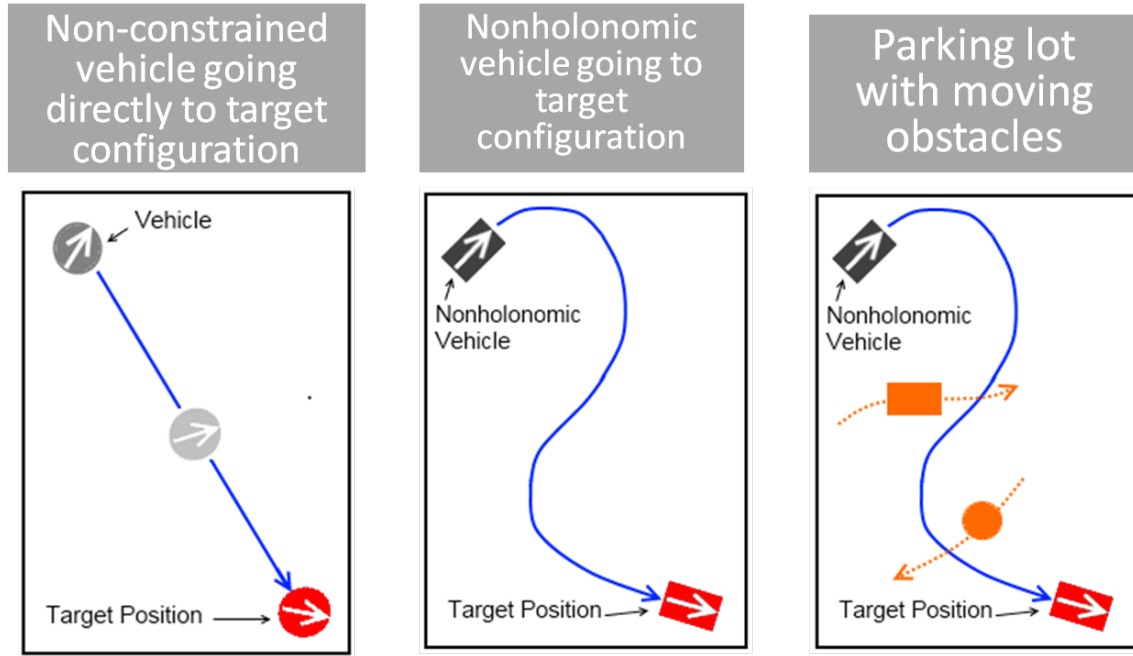


Figure 1.1: Nonholonomic vehicle in a configuration space with moving obstacles

1.2 Research Scope

A lot of research has gone into developing controls for nonholonomic systems, Non-holonomic systems in configurations spaces with obstacles and systems with moving obstacles. There has also been a lot of research into feedback nonholonomic controllers. The aim of this research, and what would be a new contribution, is a controller which combines all three: - Nonholonomic - Feedback - Obstacles

A diagram is presented to better illustrate the goal.

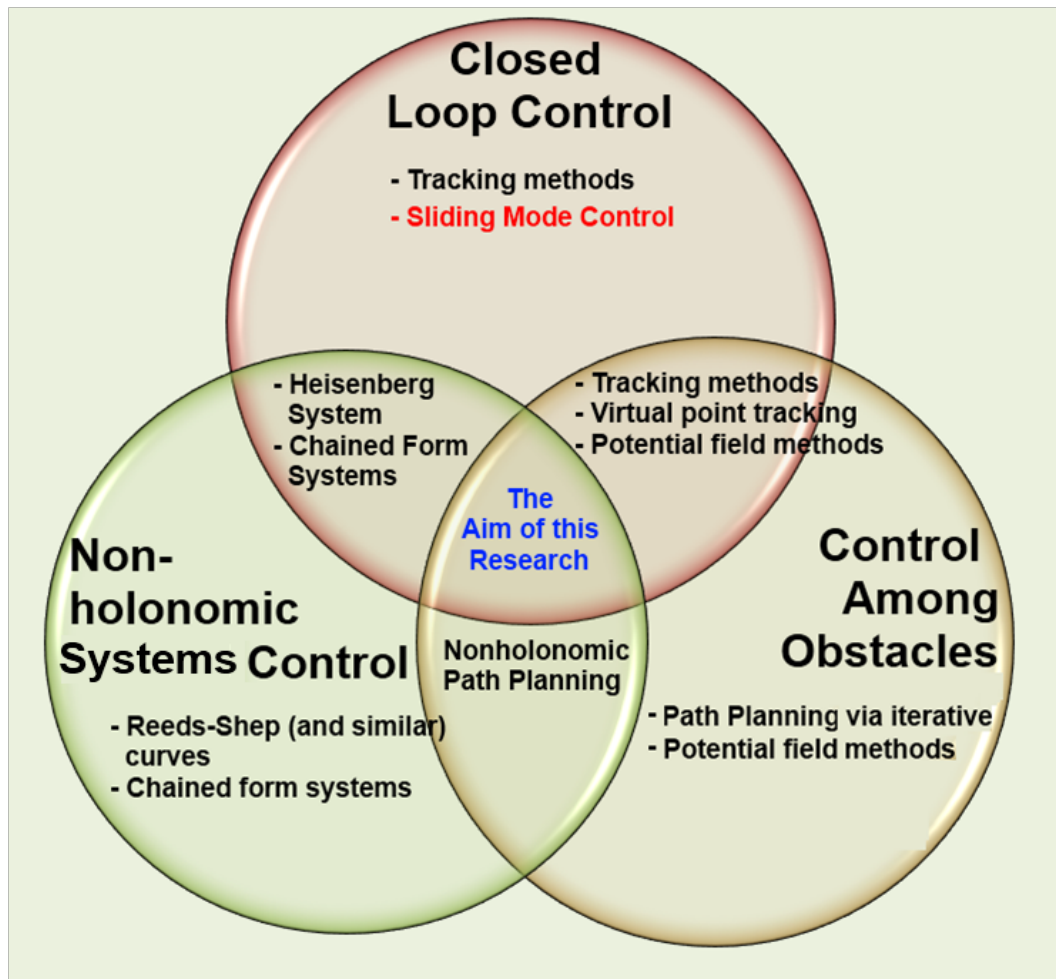


Figure 1.2: Combining nonholonomic constraints and moving obstacles

1.3 Possible Applications

A successful solution to this problem would have applications in many practical problems. Here are a couple of examples:

- I More robust satellite docking algorithms for satellites controlled by momentum wheels
- II A car automatically parking in a parking lot among other moving cars and people

Chapter 2

Theory of Systems with Nonholonomic Constraints

In this chapter we define nonholonomic systems and give some important theorems related to nonholonomic control. Some of the math involved is also reviewed. The goal of the chapter is to introduce the main concepts for the reader that does not have a background in nonholonomic systems control or differential geometry, to follow.

2.1 Nonholonomic Constraints and Systems

2.1.1 Informal Introduction with Examples

This thesis deals with systems that have nonholonomic constraints, which (for mechanical systems) can be intuitively interpreted as constraints on the ways an object can move, but not on where it can move. To be a little more precise, they constrain the types of movements allowed to get an object into a specific configuration (location and orientation), without constraining the allowed configurations. Consider for example a six-sided die, on a flat table, with the number two facing up. Now imagine that the die is allowed to role but not to slide, and it is desired to have the die two dice-lengths to the right with the number two facing up. If the die were to simply role twice to the right, it would be in the desired position, but the wrong orientation. (It

would have the number five on top.) In order to get the die into the desired position and orientation, a more complicated series of maneuvers would be required. In this example, the nonholonomic constraint is the constraint that the dice not be allowed to slide. Another example of a nonholonomic system (and one which is referred to frequently in this text) is that of a car in a parking lot. The car may move forwards or backwards, and may turn the front wheels, but it may not simply move sideways. These two examples involve an object which is allowed to roll on a plane without sliding. For a different example of a nonholonomic system, consider a falling cat. If a cat is dropped from sufficient height, it will reorient itself so that its legs are pointed towards the ground. The cat has no direct control on its orientation, but by swinging its legs it can effect its angular momentum around its midsection.

2.1.2 Formal Definition

A formal definition of a nonholonomic system is a system which has non-integrable velocity constraints.

The type of nonholonomic constraints that we are concerned with are of the form,

$$\omega_j(x)\dot{x} = 0 \quad j = 1, \dots, k, \quad (2.1)$$

where x represents the configuration of the system in phase space \mathcal{PS} , and each $\omega_j(x)$ is a row vector in \mathbb{R}^n .

Not all constraints of the form (2.1) are non-integrable. To demonstrate, we will integrate, and see what we get.

$$\int_0^{t^f} \omega(x) \frac{dx}{dt} dt = 0 \quad (2.2)$$

This becomes a path integral.

$$\int_{x^i}^{x^f} \omega(x) dx. \quad (2.3)$$

It is worth pointing out that ω is an $n \times k$ -matrix and x is an n -element vector. This type of integral is called a Pfaffian System Frobenius [1877].

If the path integral is path-independent i.e. it's equal to $h(x^f) - h(x^i)$ for some function $h(x)$, then the constraints are holonomic and can be expressed as $h(x) = \text{const}$. Otherwise, the constraint (2.1) is nonholonomic.

2.1.3 Mathematical Preliminaries

This section aims at familiarizing the reader with some vocabulary and theorems which are very useful when discussing nonholonomic control. Much of the content in this section is taken from (Murray and Sastry Murray and Sastry [1993]).

Consider a set of k non-integrable velocity constraints of the form:

$$\omega_j^T(x)\dot{x} = 0 \quad j = 1, \dots, k \quad (2.4)$$

$$x \in \mathcal{PS} \subset \mathbb{R}^n, \quad (2.5)$$

where the ω_j 's are linearly independent and smooth, and where x represents the configuration of the system in phase space \mathcal{PS} . Suppose we want to drive the system from initial configuration $x(0) = x^i$ to a final goal configuration x^f . A control system is then expressed in the form,

$$\Sigma: \quad \dot{x} = g_0(x) + g_1(x)u_1 + \dots + g_m(x)u_m \quad x \in \mathcal{PS} \quad u \in \mathcal{PS} \subset \mathbb{R}^m \quad (2.6)$$

$$g_i(x) \in \mathbb{R}^n, \quad i = 1, \dots, n - k \quad (2.7)$$

where the g_i 's are smooth and linearly independent vector fields, g_0 is a drift term, and the u_i 's are elements of the control space, $U \subset \mathbb{R}^m$.

Here's a quick summary of our notation to this point,

$$\mathcal{PS} \in \mathbb{R}^n, \quad \text{phase space}$$

$$U \in \mathbb{R}^m, \quad \text{control space}$$

$$\omega \in \mathbb{R}^{n \times k}, \quad \text{nonholonomic constraints}$$

$$k + m \leq n$$

Imagine a path along the gradient of a scalar field, this is called a flow ϕ_t^f . A more

formal definition is that a flow is an integration along a vector field. The cyclical series of infinitesimal motions $\phi_\epsilon^{g_i}$ along vector fields g_1 and g_2 Murray and Sastry [1993],

$$\phi_\epsilon^{-g_1} \circ \phi_\epsilon^{-g_2} \circ \phi_\epsilon^{g_1} \circ \phi_\epsilon^{g_2}(x_0) \quad (2.8)$$

is depicted in the following diagram from a paper by Murray and Sastry Murray and Sastry [1993].

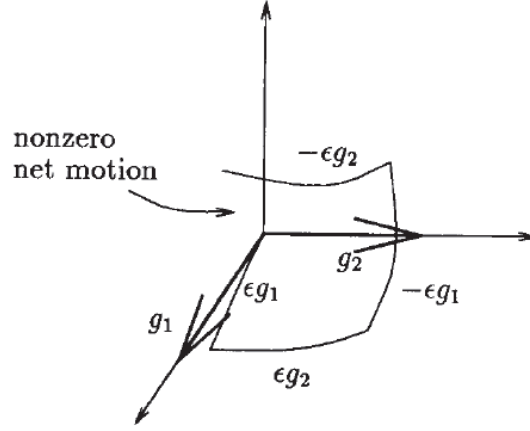


Figure 2.1: Lie bracket motion

Notice that the net motion is nonzero. This nonzero net motion is a physical interpretation of a non-commuting Lie bracket $[g_1, g_2]$ of two vector fields g_1 and g_2 . i.e.

$$[g_1, g_2] \neq 0 \quad (2.9)$$

This brings us to Lie brackets. A Lie bracket of two vector fields f and g is defined as

$$[f, g] = \frac{\partial g}{\partial x} f - \frac{\partial f}{\partial x} g \quad (2.10)$$

Some important properties of Lie brackets are the Jacobi identity,

$$[f, [g, h]] + [g, [h, f]] + [h, [f, g]] = 0, \quad (2.11)$$

and skew-symmetry,

$$[f, g] = -[g, f]. \quad (2.12)$$

Let us now define the distribution, Δ such that

$$\Delta = \text{span}\{g_1, \dots, g_m\}. \quad (2.13)$$

A distribution is said to be involutive if Lie brackets of all of the tangent vectors, as well as the tangent vectors themselves, are elements of the distribution. i.e. Murray and Sastry [1993],

$$\Delta \text{ involutive} \iff \forall f, g \in \Delta, \quad [f, g] \in \Delta. \quad (2.14)$$

Now imagine that a distribution Δ was not involutive, but a larger distribution was chosen, which contained $[f, g]$ as well as f and g , such that it was involutive. An involutive closure, $\bar{\Delta}$ is the smallest distribution, containing the original distribution which is involutive.

Definition:

A Lie algebra \mathcal{A} is a space, or distribution which is closed on $[f, g]$ for all $f, g \in \mathcal{A}$, which also satisfies the properties of Jacobi identity and skew-symmetry.

Definition:

A Lie group is a set of all transformations inside a Lie algebra.

The reason for all of this discussion of distributions is that there is a nice theorem which relates controllability of nonlinear systems to distributions.

Chow's theorem states that if, $\bar{\Delta}_x = \mathbb{R}^n \forall x \in \mathcal{PS}$, then the system is controllable on \mathcal{PS} , where $\bar{\Delta}_x$ is $\bar{\Delta}$ evaluated at x .

Chow's Theorem is given in a useful format in Murray and Sastry [1993]:

$$\phi_\epsilon^{-f} \circ \phi_\epsilon^{-g} \circ \phi_\epsilon^f \circ \phi_\epsilon^g(x_0) = \epsilon^2[f, g](x_0) + o(\epsilon^2) \quad (2.15)$$

Chow's Theorem is similar to Frobenius' Theorem, which states that a distribution

is integrable if and only if it is involutive,

$$\Delta \text{ involutive} \iff \Delta \text{ integrable.} \quad (2.16)$$

2.1.4 Degrees of Nonholonomy

Suppose we have a nonholonomic system such that $\Delta = \text{span}\{g_1, \dots, g_m\}$ and $\Delta \neq \bar{\Delta}$. Let us define the following distributions,

$$G_1 = \Delta, \quad (2.17)$$

$$G_2 = G_1 + \text{span}\{[X, Y] | X \in G_1, Y \in G_1\}, \quad (2.18)$$

$$G_3 = G_2 + \text{span}\{[X, Y] | X \in G_1, Y \in G_2\}. \quad (2.19)$$

G_2 will not equal G_1 since the system was nonholonomic, i.e. G_1 was involutive. Therefore,

$$\dim(G_2) = \dim(G_1) + 1 = m + 1, \quad (2.20)$$

Depending on the degree of nonholonomy, G_3 may or may not equal G_2 . If we continue in this manor until we find some distribution $G_{k^*} = G_{k-1}$, then $G_{k^*} = \bar{\Delta}$. That is to say, Murray and Sastry [1993]

$$G_1 = \Delta, \quad (2.21)$$

$$G_k = G_{k-1} + \text{span}\{[X, Y] | X \in G_1, Y \in G_{k-1}\}, \quad (2.22)$$

$$G_{k^*} = G_{k-1}, \quad (2.23)$$

$$G_{k^*} = \bar{\Delta}. \quad (2.24)$$

The set of all of these distributions $(G_1, G_2, \dots, G_{k^*})$ is called the filtration. As long as the rank of the distributions does not depend on x , i.e. $\text{rank}(G_i(x)) = \text{rank}(G_i(x_0)) \quad \forall x \in \mathcal{PS}$, then the filtration is said to be regular, and $k^* - 1$ is the degree of nonholonomy.

2.1.5 Brockett's Theorem

Brockett's necessary stability conditions, Brockett [1983], state that for state- dependant nonholonomic control we must use discontinuous or time-varying smooth control.

To summarize his this part of his paper:

There exists no continuous control law $(u, v) = (u(x, y, z), v(x, y, z))$, which makes the origin asymptotically stable for

$$\dot{x} = u \tag{2.25}$$

$$\dot{y} = v \tag{2.26}$$

$$\dot{z} = xv - yu \tag{2.27}$$

Brockett showed this condition for the specific example above, (a third order nonholonomic system called the Heisenberg system), but this same condition can be extended to a generalized Heisenberg system. This is especially useful, since in his previous work, Brockett [1981], Brockett showed that any system of the form,

$$\dot{x} = B(x)u, \tag{2.28}$$

$$u \in \mathbb{R}^m, \quad x \in \mathbb{R}^n, \quad n = m(m+1)/2 \tag{2.29}$$

can be transformed to the generalized Heisenberg system,

$$\dot{x} = u \tag{2.30}$$

$$\dot{Y} = xu^T - ux^T. \tag{2.31}$$

Here, $x, u \in \mathbb{R}^m$, $Y \in o(m)$, where $o(m)$ is the set of $m \times m$, skew symmetric matrices. In terms of indices, the generalized Heisenberg system can be expressed as,

$$\dot{x}_i = u_i, \quad i = 1, 2, \dots, m \tag{2.32}$$

$$\dot{Y}_{ij} = x_i u_j - u_j x_i, \quad i, j = 1, 2, \dots, m. \tag{2.33}$$

For instance, consider a system, of the form $\dot{x} = B(x)u$, where there are 4 independent controls and 10 state variables. In this example, $n = 10$ and $m = 4$. After verifying that the system meets the condition that $n = m(m + 1)/2$, we know we should be able to convert it to the generalized Heisenberg system,

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \quad (2.34)$$

$$\dot{Y} = \begin{bmatrix} 0 & u_1x_2 - u_2x_1 & -Y_{3,1} & u_1x_4 - u_4x_1 \\ -Y_{1,2} & 0 & u_2x_3 - u_3x_2 & -Y_{4,2} \\ u_3x_1 - u_1x_3 & -Y_{2,3} & 0 & u_3x_4 - u_4x_3 \\ -Y_{1,4} & u_4x_2 - u_2x_4 & -Y_{3,4} & 0 \end{bmatrix}. \quad (2.35)$$

An even further generalization expresses the Brockett, or Heisenberg system on Lie groups, and in doing so, drops the requirement for x and u to be vectors, Bloch [2003]

$$\dot{x} = u \quad (2.36)$$

$$\dot{y} = [x, u]. \quad (2.37)$$

The Heisenberg system more formally introduced and discussed in the following chapter.

Aside: For those not familiar with controls, systems are generally described to fit one of three forms, $\dot{x} = Ax + Bu$, $\dot{x} = B(x)u$ or $\dot{x} = f(x, u)$, where the first form is linear, the second, may or may not be linearizable, and the third is the most general.

In regards to the problem statement of this thesis, Brockett's condition for asymptotic stability indicates the problem will have to be solved with discontinuous controls.

Let us now consider an example to put some of these theorems into practice.

2.2 Examples of Nonholonomic Systems

In this section an example is given which comes from (Murray and Sastry, Murray and Sastry [1993]). The notation is changed to match the rest of this thesis, and some additional steps are given.

2.2.1 Nonholonomic Vehicle Model

Let us consider a simplified car model (depicted in Figure 2.2).

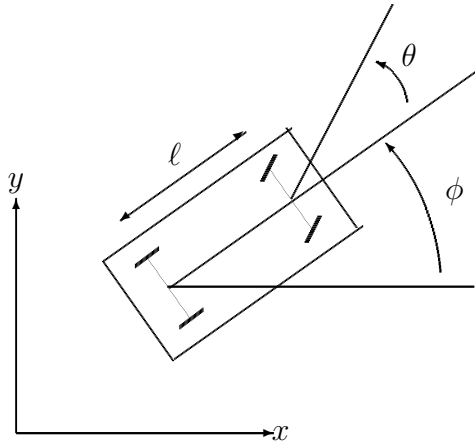


Figure 2.2: Single car

The equations of motion are given as follows:

$$\dot{x} = v \cos \phi, \quad (2.38)$$

$$\dot{y} = v \sin \phi, \quad (2.39)$$

$$\dot{\theta} = \omega, \quad (2.40)$$

$$\dot{\phi} = \frac{v}{\ell} \tan \theta, \quad (2.41)$$

where the coordinates (x,y) are taken from the center of the rear axle and the orientation θ is the angle between the x-axis and a line crossing the center of both front and rear axles, ℓ is the distance between the two axles, v is the speed, which can be

negative for backing up, u is the steering turning rate.

Let us make the substitutions into state variables, configuration $(x, y, \theta, \phi)^T = (x_1, x_2, x_3, x_4)^T = x$, and controls $(v, \omega) = (u_1, u_2)^T = u$. With these variable changes, our equations of motion become,

$$\dot{x}_1 = u_1 \cos x_4, \quad (2.42)$$

$$\dot{x}_2 = u_1 \sin x_4, \quad (2.43)$$

$$\dot{x}_3 = u_2, \quad (2.44)$$

$$\dot{x}_4 = \frac{u_1}{\ell} \tan x_3. \quad (2.45)$$

Notice that this system could be expressed the form $\dot{x} = B(x)u$, where x is a matrix and, \dot{x} and u are column vectors,

$$\underbrace{\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} \cos x_4 & 0 \\ \sin x_4 & 0 \\ 0 & 1 \\ \frac{1}{\ell} \tan x_3 & 0 \end{bmatrix}}_{B(x)} \underbrace{\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_u \quad (2.46)$$

In section (2.1.5) we stated that any system of this form, $\dot{x} = B(x)u$ can be expressed as a generalized Heisenberg system, and can be controlled with discontinuous feedback control.

For the purpose of this example though, it will be more useful to express the system as a single vector with basis elements given by $(\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_n})$, i.e.

$$\dot{x} = \left(\dot{x}_1 \frac{\partial}{\partial x_1} + \dot{x}_2 \frac{\partial}{\partial x_2} + \dots + \dot{x}_n \frac{\partial}{\partial x_n} \right). \quad (2.47)$$

Specifically,

$$\dot{x} = \left(u_1 \cos x_4 \frac{\partial}{\partial x_1} + u_1 \sin x_4 \frac{\partial}{\partial x_2} + u_2 \frac{\partial}{\partial x_3} + \frac{u_1}{\ell} \tan x_3 \frac{\partial}{\partial x_4} \right). \quad (2.48)$$

By grouping the terms and factoring out the controls, we can find our vector fields,

$(\{g_1(x), \dots, g_m(x)\}),$

$$\dot{x} = \underbrace{\left(\cos x_4 \frac{\partial}{\partial x_1} + \sin x_4 \frac{\partial}{\partial x_2} + \frac{1}{\ell} \tan x_3 \frac{\partial}{\partial x_4} \right)}_{g_1} u_1 + \underbrace{\left(\dot{x}_3 \frac{\partial}{\partial x_3} \right)}_{g_2} u_2. \quad (2.49)$$

We then find our filtration using Lie brackets: Murray and Sastry [1993]

$$\begin{aligned} g_1 &= \cos x_4 \frac{\partial}{\partial x_1} + \sin x_4 \frac{\partial}{\partial x_2} + \frac{1}{\ell} \tan x_3 \frac{\partial}{\partial x_4} \\ g_2 &= \frac{\partial}{\partial x_3} \\ g_3 = [g_1, g_2] &= \frac{-1}{\ell \cos^2 x_3} \frac{\partial}{\partial x_4} \\ g_4 = [g_1, g_3] &= \frac{-\sin x_4}{\ell \cos^2 x_3} \frac{\partial}{\partial x_1} + \frac{\cos x_4}{\ell \cos^2 x_3} \frac{\partial}{\partial x_2} \\ g_5 = [g_2, g_3] &= \frac{-2 \tan x_3}{\ell \cos^2 x_3} \frac{\partial}{\partial x_4} \end{aligned} \quad (2.50)$$

As can be shown, the vector fields, g_1 , g_2 , g_3 and g_4 span the entire tangent space (isomorphic to \mathbb{R}^4). This means that the system is controllable. Since two additional vector fields were produced in the filtration to span the involutive distribution, the degree of nonholonomy is 2.

2.2.2 Extension to N Trailers

The N-trailer extension is done by adding N two-wheeled trailers each one attached and pivoting at the center of the preceding trailer or car. The bar connecting the i 'th trailer to the $(i - 1)$ 'th axel is denoted, d_i . The orientation of the i 'th trailer is ϕ_i . See figure, 2.3.

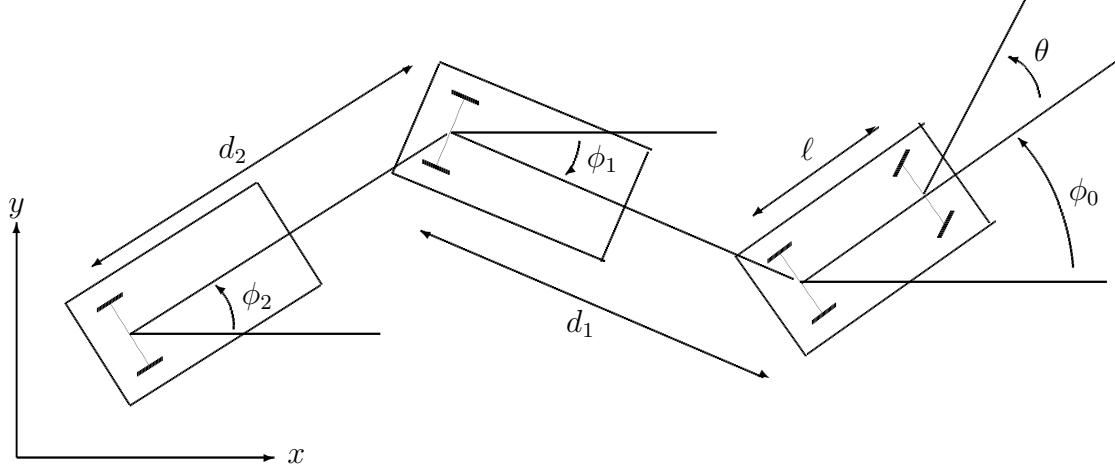


Figure 2.3: Car with N-trailers Murray and Sastry [1993]

The equations of motion for the first car remain the same as in the one car example,

$$\dot{x} = v_0 \cos \phi_0, \quad (2.51)$$

$$\dot{y} = v_0 \sin \phi_0, \quad (2.52)$$

$$\dot{\theta} = \omega, \quad (2.53)$$

$$\dot{\phi}_0 = \frac{v_0}{\ell} \tan \theta, \quad (2.54)$$

but the orientation is now denoted ϕ_0 , and speed, as v_0 .

The speed of each trailer has two components, the speed parallel to its wheels, v_i , and the speed perpendicular to its wheels, $\dot{\phi}_i$. We can write the speed of each trailer as a function of the speed of the trailer or car which is pulling it.

$$\dot{\phi}_i = \frac{1}{d_i} \sin(\phi_{i-1} - \phi_i) v_{i-1}, \quad (2.55)$$

$$v_i = \cos(\phi_{i-1} - \phi_i) v_{i-1} \quad (2.56)$$

Let us find the perpendicular speed of the first three trailers,

$$\dot{\phi}_1 = \frac{1}{d_1} \sin(\phi_0 - \phi_1) v_0, \quad (2.57)$$

$$v_1 = \cos(\phi_0 - \phi_1) v_0, \quad (2.58)$$

$$\dot{\phi}_2 = \frac{1}{d_2} \sin(\phi_1 - \phi_2) \cos(\phi_0 - \phi_1) v_0, \quad (2.59)$$

$$v_2 = \cos(\phi_1 - \phi_2) \cos(\phi_0 - \phi_1) v_0, \quad (2.60)$$

$$\dot{\phi}_3 = \frac{1}{d_3} \sin(\phi_2 - \phi_3) \cos(\phi_1 - \phi_2) \cos(\phi_0 - \phi_1) v_0. \quad (2.61)$$

From these results we can see that the perpendicular speed of the i 'th trailer is given by,

$$\dot{\phi}_i = \frac{1}{d_i} \left(\prod_{j=1}^{i-1} \cos(\phi_{j-1} - \phi_j) \right) \sin(\phi_{i-1} - \phi_i) v_0 \quad (2.62)$$

In our state-variables, where the configuration, $(x, y, \theta, \phi_0, \dots, \phi_N)$ is denoted $(x_1, x_2, x_3, x_4, \dots, x_{N+4})$, and the controls, (v_0, ω) are written as (u_1, u_2) . The equations of motion describing the N-trailer system, in state variables is then,

$$\dot{x}_1 = u_1 \cos x_4, \quad (2.63)$$

$$\dot{x}_2 = u_1 \sin x_4, \quad (2.64)$$

$$\dot{x}_3 = u_2, \quad (2.65)$$

$$\dot{x}_4 = \frac{u_1}{\ell} \tan x_3, \quad (2.66)$$

$$\dot{x}_{i+4} = \frac{1}{d_i} \left(\prod_{j=5}^{i+3} \cos(x_{j-1} - x_j) \right) \sin(x_{i+3} - x_{i+4}) u_1. \quad (2.67)$$

For higher numbers of N , it is obvious that finding the filtration would be very tedious. In Laumond [1990] it was shown that the degree of nonholonomy is $N + 2$.

Chapter 3

Nonholonomic Control Systems

The methods of nonholonomic path planning have become very well developed. Through iterative methods, people are able to steer complicated shapes through complicated environments. Examples of some very impressive path planners are (LaValle [2006], Sekhavat et al. [1998], & Srinivasan et al. [2005]). For the purpose of this thesis, not much research was made into the area of iterative path planners, because of the perceived difficulty to apply them to closed loop control.

3.1 Nonholonomic Path Planning with Obstacles

Control by Constant Radius turns and Straight Lines

Some of the earliest attempts in nonholonomic path planning were by combining a minimal number of constant radius turns and straight lines. For the car example, this is equivalent to limiting the steering angle to either full right (ϕ_{max}), full left ($-\phi_{max}$), or straight ($\phi = 0$). This approach was first developed by L.E.Dubins. In his 1957 paper Dubins [1957], he found the optimal paths under these constant radius constraints. Dubins also restricted the system to forward movement.

Later Reeds and Shepp, expanded on this research to allow reversing. If both forward and backward motion is allowed, the trajectories are called Reeds-Shepp Curves. If only forward motion is allowed the trajectories created are called Dubins Curves,

and if the time that it takes to rotate the robot body around a central axis (for differential drive systems) is included in the cost functional, then the trajectories created are called Balkcom-Mason curves. LaValle [2006].

Dubins Curves

In 1957 L.E. Dubins published a paper Dubins [1957] on what he called an R-geodesic, and what has come to be known as a Dubins curve. The Idea was to find a path of minimum length between two points and orientations but with the limitation that the path must only be composed of straight lines and curves of some constant radius R . In two dimensional space Dubins curves are constructed as combinations of straight lines (denoted L) and curved paths (denoted C), so that a particular path could be denoted as type CLC, LCL or CCC. In this paper Dubins showed that no R-geodesic will be composed of more than three segments. An easily read and expanded description of the use of Dubins curves is provided in LaValle [2006]. In LaValle's description the path segments described above are denoted; "R" for right turn, "L" for left turn or "S" for straight. The subscripts " α " and " γ " are used to denote the angle traversed for right and left paths respectively, and the subscript "d" denotes the distance for straight lines. Figure 3.1 which was taken from LaValle [2006], shows two possible Dubins Curves.

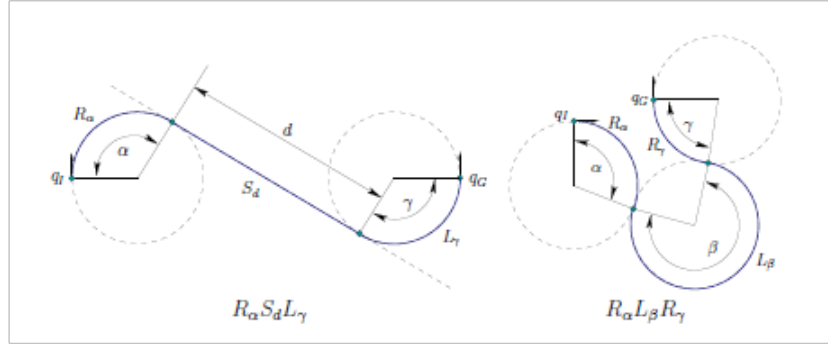


Figure 3.1: The two possible shapes of Dubins Curves. LaValle [2006]

Reeds-Shepp Curves

In their paper Reeds and Shepp [1990], J. Reeds and L. Shepp, expanded on the Dubins curves by allowing for reversal in directions. In this paper they showed that the shortest path composed solely of straight lines, constant radius turns and direction reversals, can be described by one of 48 possible combinations, and no more than five segments.

Balkcom-Mason Curves

The next extension along this line was to consider the time it takes to reverse direction, when picking an optimal path. This was done by D. Balkcom and M. Mason in their paper Balkcom and Mason [2002].

3.2 Nonholonomic Feedback Control

3.2.1 The Heisenberg System

The kinematic model for nonholonomic systems called the nonholonomic integrator or Heisenberg system has been studied extensively. The name of this system is borrowed from quantum mechanics, because the Lie algebra for these systems is identical to one of a like named quantum mechanical system.

The idea is to transform the three state variables into a form where two of them can be controlled in such a way that the third reaches the origin prior to either of the two controlled variables. The most commonly cited form of the Heisenberg system is as follows as described by Brockett [1983].

$$\dot{x}_1 = u_1, \quad (3.1)$$

$$\dot{x}_2 = u_2, \quad (3.2)$$

$$\dot{x}_3 = x_1 u_2 - x_2 u_1, \quad (3.3)$$

where:

$$u_1, u_2 \quad \text{controls,}$$

$$x_1, x_2, x_3 \quad \text{state variables.}$$

Nonholonomic Vehicle Example

A commonly cited example of a nonholonomic system is a simple four-wheel car. Consider the simplified car model depicted in figure, (3.2), and described by equations, (3.4, 3.5 and 3.6).

$$\dot{x} = v \cos \phi, \quad (3.4)$$

$$\dot{y} = v \sin \phi, \quad (3.5)$$

$$\omega = \dot{\phi} = \frac{v}{\ell} \tan \theta \quad (3.6)$$

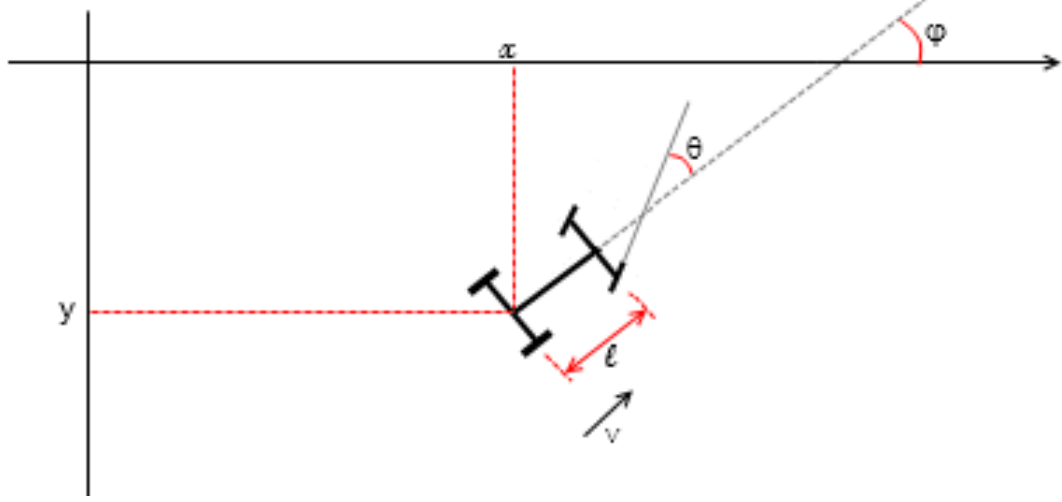


Figure 3.2: Classic nonholonomic car model

Where, the coordinates (x,y) are taken from the center of the rear axle and the orientation, θ , is the angle between the x -axis and a line crossing the center of both front and rear axles. ℓ is the distance between the two axles.

The controls considered are (1) the steering angle, ϕ , with respect to the car and (2) the speed, v , (which can be forwards or reverse).

Another common system for this type of problem which is kinematically equivalent is a three-wheeled car with only one wheel in the front. Although this simplification may be slightly easier to visualize, it changes none of math, so one could consider either case interchangeably. For an example of a paper using this form see: LaValle [2006]

The above system is not integrable because θ and ϕ as functions of time are unknown.

This model transformed to Heisenberg space as follows.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = T(\phi) \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}, \quad (3.7)$$

The transformation from the configuration space, \mathcal{CS} to the Heisenberg space, \mathcal{H} ,

is given by the following: Bloch [2003]

$$T(\phi) = \begin{bmatrix} 0 & 0 & 1 \\ \cos \phi & \sin \phi & 0 \\ \phi \cos \phi - 2 \sin \phi & \phi \sin \phi + 2 \cos \phi & 0 \end{bmatrix} \quad (3.8)$$

We need to find the transformed controls. To do this, let's find the time derivative of the transformation of our system. If we use the notation $[x, y, \phi]^T = \mathbf{x}$, then we can find the derivative using the product rule. I.e. $\frac{d}{dt}(T\mathbf{x}) = \dot{T}\mathbf{x} + T\dot{\mathbf{x}}$ or,

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \frac{dT(\phi)}{dt} \begin{bmatrix} x \\ y \\ \phi \end{bmatrix} + T(\phi) \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix} \quad (3.9)$$

Let's look at the time derivative of the transformation matrix,

$$\begin{aligned} \frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 0 \\ -\omega \sin \phi & \omega \cos \phi & 0 \\ -\omega \cos \phi - \omega \phi \sin \phi & -\omega \sin \phi + \omega \phi \cos \phi & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ \phi \end{bmatrix} \\ &+ \begin{bmatrix} 0 & 0 & 1 \\ \cos \phi & \sin \phi & 0 \\ \phi \cos \phi - 2 \sin \phi & \phi \sin \phi + 2 \cos \phi & 0 \end{bmatrix} \cdot \begin{bmatrix} v \cos \phi \\ v \sin \phi \\ \omega \end{bmatrix} \end{aligned} \quad (3.10)$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ -x\omega \sin \phi + y\omega \cos \phi \\ -x\omega \cos \phi - x\omega \phi \sin \phi - y\omega \sin \phi + y\omega \phi \cos \phi \end{bmatrix} + \begin{bmatrix} \omega \\ v \\ v\phi \end{bmatrix} \quad (3.11)$$

which leads to,

$$\dot{x}_1 = \omega \quad (3.12)$$

$$\dot{x}_2 = -x\omega \sin \phi + y\omega \cos \phi + v \quad (3.13)$$

$$\dot{x}_3 = -\omega(x \cos \phi + y \sin \phi) + \underbrace{\phi(-x\omega \sin \phi + y\omega \cos \phi + v)}_{\dot{x}_2}. \quad (3.14)$$

Now, let's look at the transformation of our system, $x = T\mathbf{x}$, to see if it will help to eliminate the configuration space variables, (x, y, ϕ) .

$$x_1 = \phi \quad (3.15)$$

$$x_2 = x \cos \phi + y \sin \phi \quad (3.16)$$

$$x_3 = x\phi \cos \phi - 2x \sin \phi + y\phi \sin \phi + 2y \cos \phi \quad (3.17)$$

Notice that,

$$2 \frac{\dot{x}_2 - v}{\omega} = x_3 - x_1 x_2 \quad (3.18)$$

and

$$\dot{x}_3 = -\omega x_2 + x_1 \dot{x}_2 \quad (3.19)$$

This leads to,

$$\dot{x}_1 = \omega \quad (3.20)$$

$$\dot{x}_2 = \omega \left(\frac{x_3}{2} - \frac{x_1 x_2}{2} \right) + v \quad (3.21)$$

$$\dot{x}_3 = -\omega x_2 + x_1 \dot{x}_2 \quad (3.22)$$

where $\omega = \frac{v}{\ell} \tan \theta$.

If we define the controls as follows,

$$\begin{aligned} u_1 &= \omega, \\ u_2 &= \omega \left(\frac{x_3}{2} - \frac{x_1 x_2}{2} \right) + v, \end{aligned}$$

we obtain the Heisenberg System,

$$\dot{x}_1 = u_1 \tag{3.23}$$

$$\dot{x}_2 = u_2 \tag{3.24}$$

$$\dot{x}_3 = x_1 u_2 - x_2 u_1 \tag{3.25}$$

In the next sections we will discuss ways to control this system.

3.2.2 Bloch-Drakunov Controller

Several sliding mode controllers were developed by Bloch and Drakunov Bloch and Drakunov [1996] in the 1990's, which are capable of stabilizing nonholonomic systems to the origin.

The derivation of these controllers is well presented in several sources including, Bloch and Drakunov [1996], Bloch [2003]. This derivation is shown again here, because in later chapters of this thesis, this same procedure is followed, but with various alterations. It is believed that having a full derivation here will provide context needed for later chapters.

Consider the following controller from Bloch and Drakunov [1996].

$$u_1 = -\alpha x_1 + \beta x_2 \text{sign}(x_3) \tag{3.26}$$

$$u_2 = -\alpha x_2 - \beta x_1 \text{sign}(x_3) \tag{3.27}$$

Where α and β are positive constants.

Bloch and Drakunov picked a Lyapunov function, V , which only depends on two of the state variables:

$$V = \frac{1}{2}(x_1^2 + x_2^2). \tag{3.28}$$

Notice that for the Heisenberg system, x_3 must be stabilized at the same time as, or before x_1 and x_2 . The conditions for this to happen will be discussed later, but to start let's differentiate the Lyapunov function, 3.28, to test for asymptotic stability on x_1 and x_2 , i.e. on the plains of constant x_3 , (from here on the notation $\mathcal{H}_2(x_1)$ is

used to refer to these planes).

$$\dot{V} = x_1\dot{x}_1 + x_2\dot{x}_2 \quad (3.29)$$

$$= -\alpha x_1^2 + \beta x_1 x_2 \operatorname{sign}(x_3) - \alpha x_2^2 - \beta x_1 x_2 \operatorname{sign}(x_3) \quad (3.30)$$

$$= -2\alpha V \quad (3.31)$$

For positive values of α , $\dot{V} < 0$, which is a necessary condition for asymptotic stability. The expression for $\dot{V} < 0$ also has a very fortunate and useful result that the equation

$$\dot{V} = -2\alpha V, \quad (3.32)$$

has a very handy solution

$$V(t) = V(0)e^{-2\alpha t}. \quad (3.33)$$

We will use this later, but for now let us check what it takes to stabilize x_3 .

$$\dot{x}_3 = x_1 u_2 - x_2 u_1 \quad (3.34)$$

$$= -\alpha x_1 x_2 + \beta x_1^2 \operatorname{sign}(x_3) + \alpha x_1 x_2 + \beta x_2^2 \operatorname{sign}(x_3) \quad (3.35)$$

$$= -2\beta V \operatorname{sign}(x_3) \quad (3.36)$$

Integrating both sides of the equation

$$x_3(t) - x_3(0) = -2\beta \int_0^t V d\tau \operatorname{sign}(x_3) \quad (3.37)$$

$$x_3(t) = \underbrace{x_3(0)}_a - \underbrace{2\beta \operatorname{sign}(x_3) \int_0^t V d\tau}_b \quad (3.38)$$

Notice that part a and part b of the above equation, are of opposite signs. If the $x_3(0)$ is negative, part (b) will approach $|x_3(0)|$ as $\tau \rightarrow t$. Note that $\operatorname{sign}(x_3) = \operatorname{sign}(x_3(0))$ because x_3 will not cross the $x_3 = 0$ plane before time t . If $x_3(0)$ is positive, part (b) will be initially negative, and will equal $-x_3(0)$ at time t . From

this we have the condition that

$$2\beta \int_0^\infty V d\tau \geq |x_3(0)|. \quad (3.39)$$

This shows as expected that the stability of x_3 depends on V . If V goes to zero before x_3 , control of x_3 is lost. The condition that x_3 reaches zero before (or at the same time as) V , and therefor x_1 and x_2 is given by the integral, (3.37). Recall that due to the convenient result of differentiating the Lyapunov function, we have an expression for $V(t)$, (3.33), which can be integrated.

$$2\beta \int_0^\infty (V(0)e^{-2\alpha\tau}) d\tau \geq |x_3(0)| \quad (3.40)$$

$$\frac{\beta}{\alpha} V(0) \geq |x_3(0)| \quad (3.41)$$

$$\frac{\beta}{2\alpha} (x_1^2(0) + x_2^2(0)) \geq |x_3(0)| \quad (3.42)$$

for state dependant control, we may simply take the present time t , to be the initial time t_0 . This gives us the condition for stability by controllers, (3.26):

$$\frac{\beta}{2\alpha} (x_1^2 + x_2^2) \geq |x_3| \quad (3.43)$$

In the event that our system state lies inside the paraboloid defined by

$$\frac{\beta}{2\alpha} (x_1^2 + x_2^2) = |x_3|, \quad (3.44)$$

a different set of controllers can be used to push the system state back out of this paraboloid. An example of controls which would do this would be

$$\dot{x}_1 = \alpha x_1 \quad (3.45)$$

$$\dot{x}_2 = \alpha x_2. \quad (3.46)$$

If the system is not initially inside or on the paraboloid defined by equation, (3.44), it will arrive at the plane given by $x_3 = 0$ before x_1 and x_2 are zero. When this

happens, a more simplified controller can be used to drive the system to the origin along this plane:

$$\dot{x}_1 = -\alpha x_1 \quad (3.47)$$

$$\dot{x}_2 = -\alpha x_2 \quad (3.48)$$

A Simulink model of the system with this controller

The control scheme that was described in the previous section modeled in Matlab and Simulink. Here we will present that model and an example output. The model does stabilize the system.

Here is the main window of the Simulink model

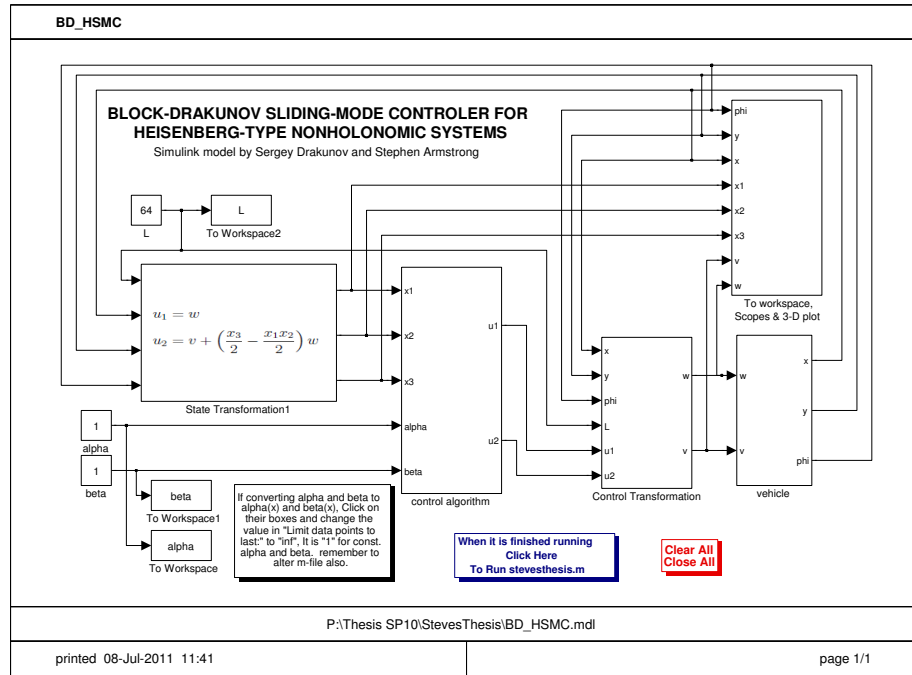


Figure 3.3: Bloch-Drakunov controller modeled with Simulink: main window

Here is the State transformation Block.

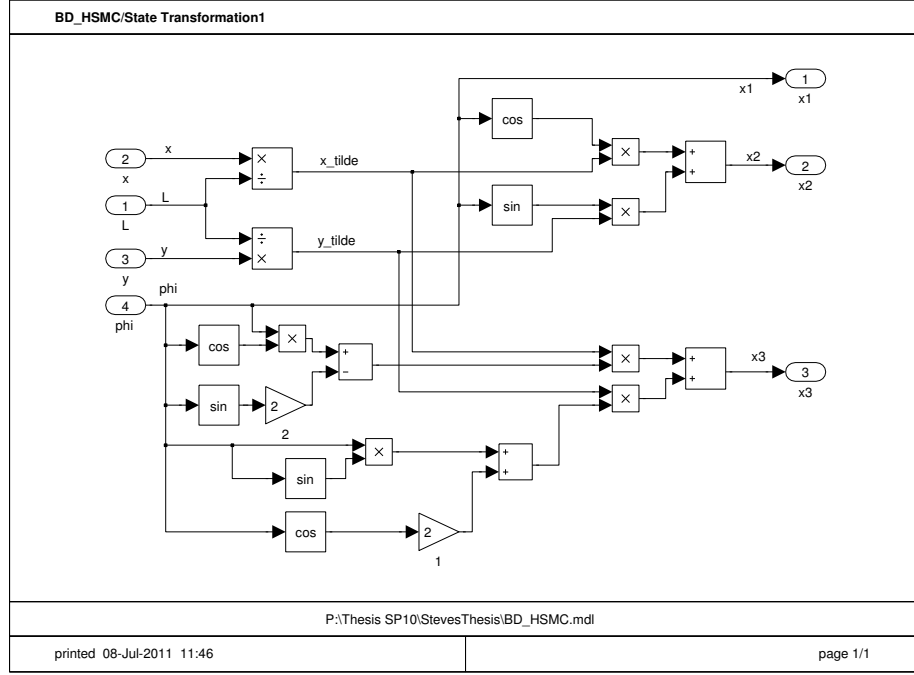


Figure 3.4: Bloch-Drakunov controller modeled with Simulink: state transformation

The state transformation is where the transformation from the configuration space \mathcal{CS} to the Heisenberg space \mathcal{H} takes place

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = T(\phi) \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}, \quad (3.49)$$

$$T(\phi) = \begin{bmatrix} 0 & 0 & 1 \\ \cos \phi & \sin \phi & 0 \\ \phi \cos \phi - 2 \sin \phi & \phi \sin \phi + 2 \cos \phi & 0 \end{bmatrix} \quad (3.50)$$

The controllers and switching conditions are shown next. The variable-control system depends on the system state in the Heisenberg space, namely whether it is

inside or outside the paraboloid defined by, $\frac{\beta}{2\alpha} (x_1^2 + x_2^2) \geq |x_3|$.

Controls	$\frac{\beta}{2\alpha} (x_1^2 + x_2^2) \geq x_3 $	$\frac{\beta}{2\alpha} (x_1^2 + x_2^2) < x_3 $	$x_3 = 0$
u_1	$-\alpha x_1 + \beta x_2 \text{sign}(x_3)$	αx_1	$-\alpha x_1$
u_2	$-\alpha x_2 - \beta x_1 \text{sign}(x_3)$	αx_2	$-\alpha x_2$

Table 3.1: Bloch-Drakunov stabilizing controller

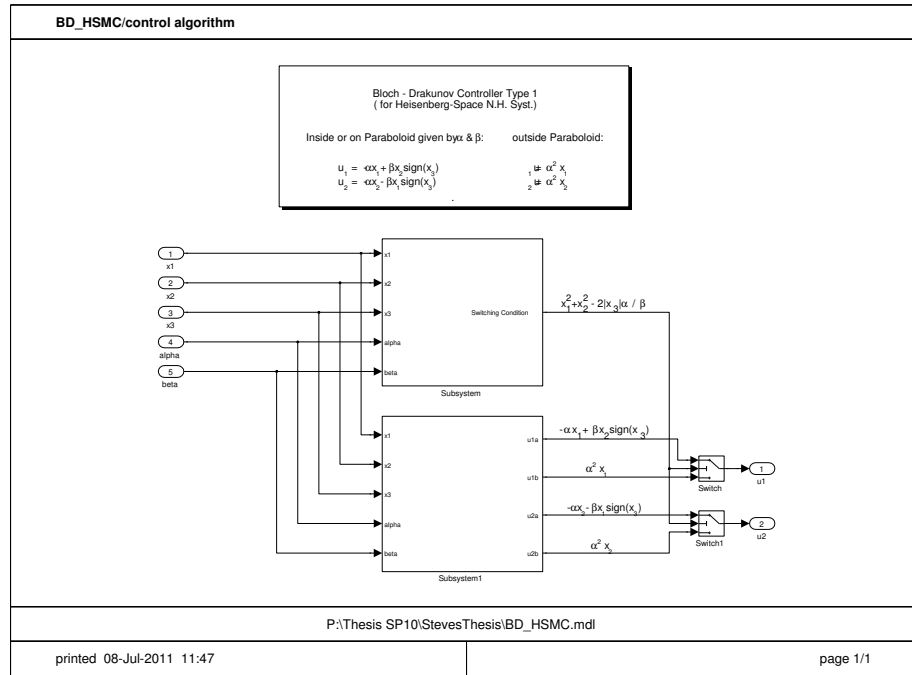


Figure 3.5: Bloch-Drakunov controller modeled with Simulink: variable structure control Algorithm

The switching condition block calculates the value of $\frac{\beta}{2\alpha} (x_1^2 + x_2^2)$.

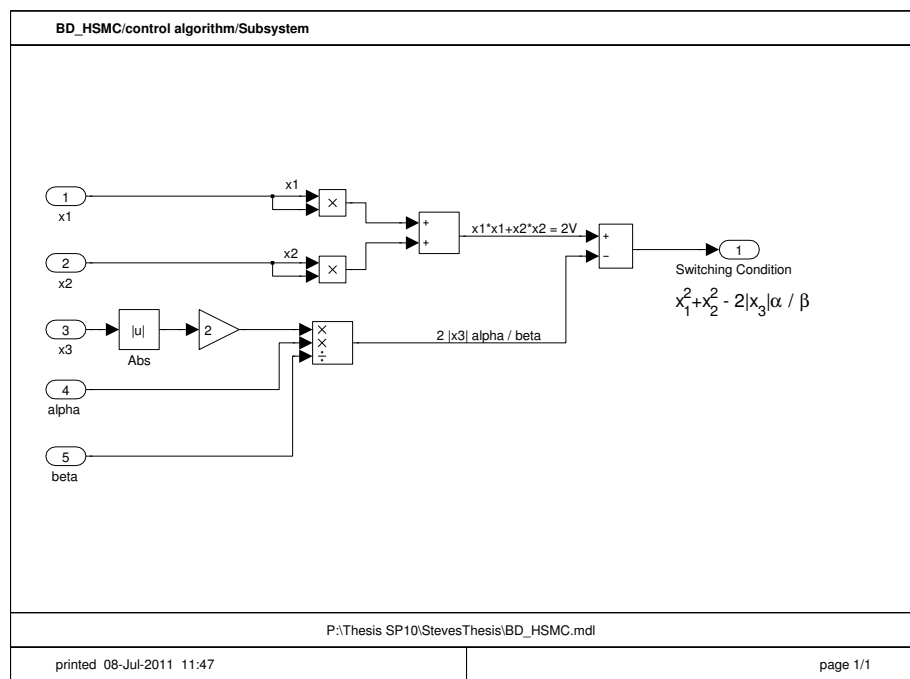


Figure 3.6: Bloch-Drakunov controller modeled with Simulink: switching condition

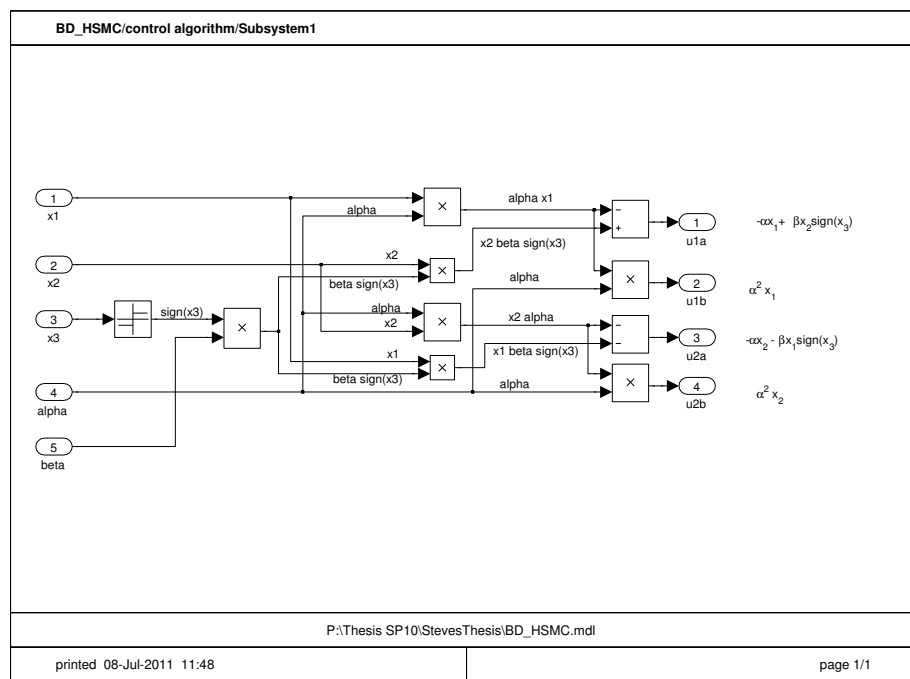


Figure 3.7: Bloch-Drakunov controller modeled with Simulink: controllers a & b

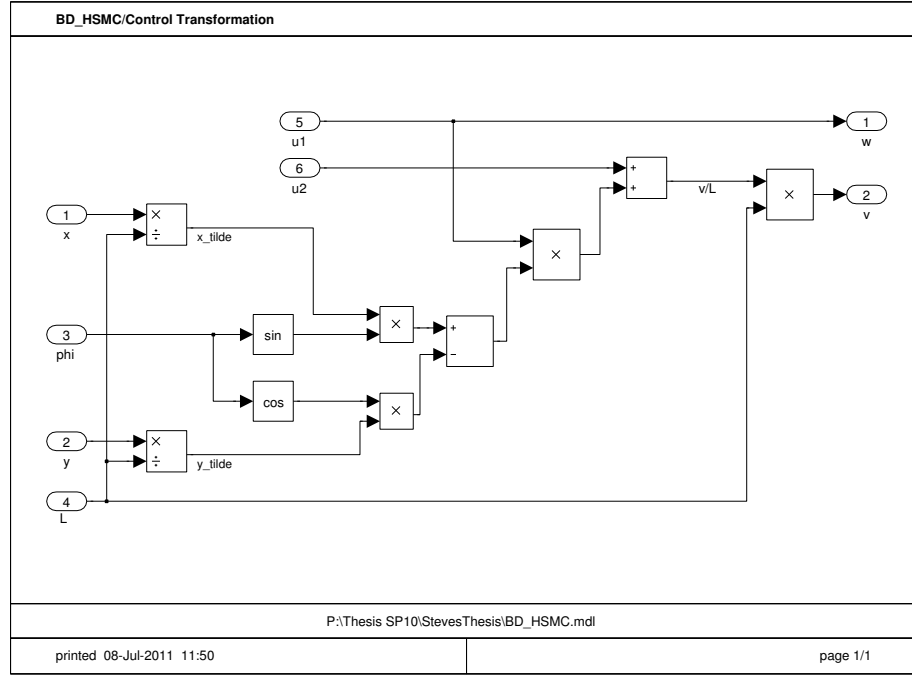


Figure 3.8: Bloch-Drakunov controller modeled with Simulink: control transformation

The controls were transformed back to state space with,

$$v = u_1$$

$$\omega = u_2 - \left(\frac{x_3}{2} - \frac{x_1 x_2}{2} \right) u_1$$

In the vehicle window is where the initial conditions are selected (by changing the initial conditions of the integrator blocks).

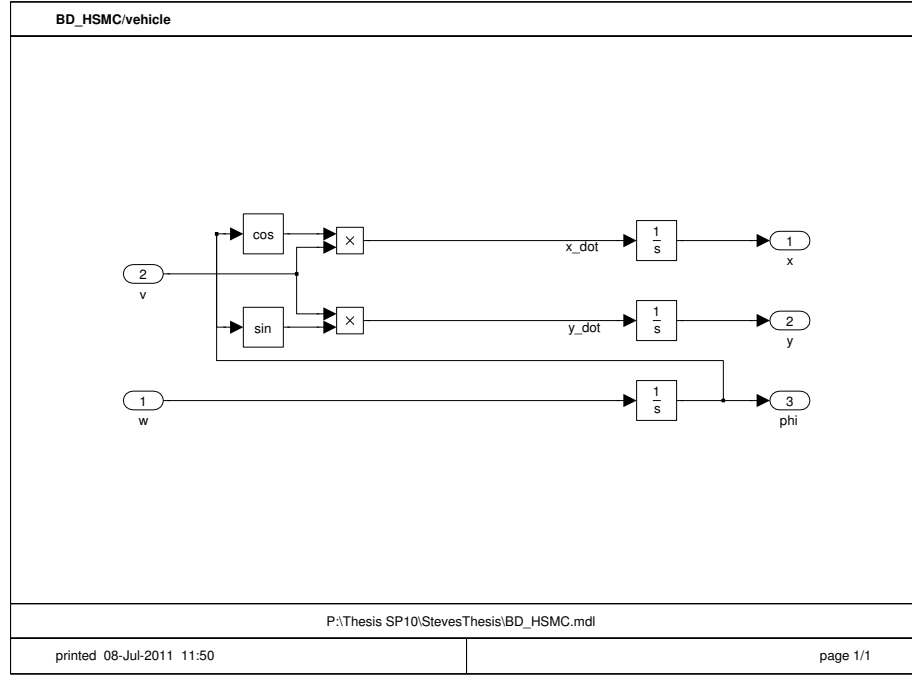


Figure 3.9: Vehicle model

Simulation Results

This control scheme works perfectly for unobstructed paths. In the diagram below, a simulated car is steered to the origin. Notice that there are two steps. First the car moves to the sliding surface in the Heisenberg Space, where it is then driven to the origin.

Here we show the convergence of two different simulations.

The first simulation had initial conditions: $(x, y, \phi)^T = (0, -20, \frac{\pi}{8})^T$.

The first two plots are of the convergence of the system states in configuration space.

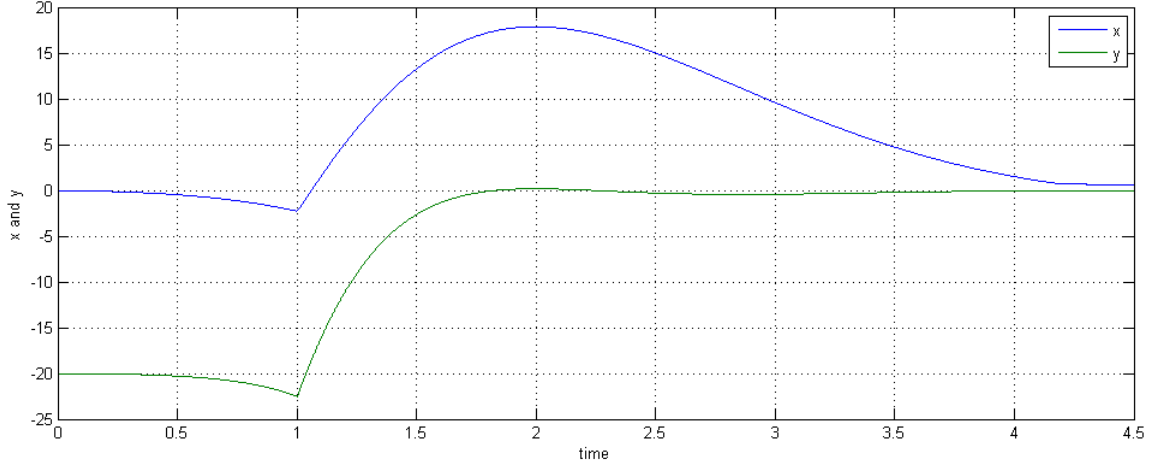


Figure 3.10: Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (x and y convergence)

Figure 3.10 shows the convergence of the location (x, y) in the configuration space.

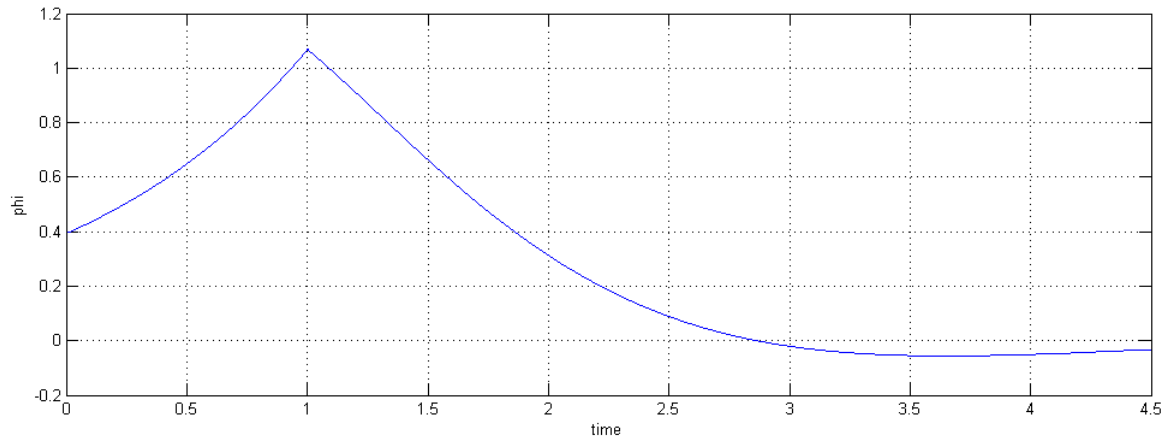


Figure 3.11: Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (ϕ convergence)

Figure 3.11 shows the convergence of the orientation ϕ in the configuration space.

The next two plots show the convergence of the system states in Heisenberg space and the Transformed controls in Heisenberg space.

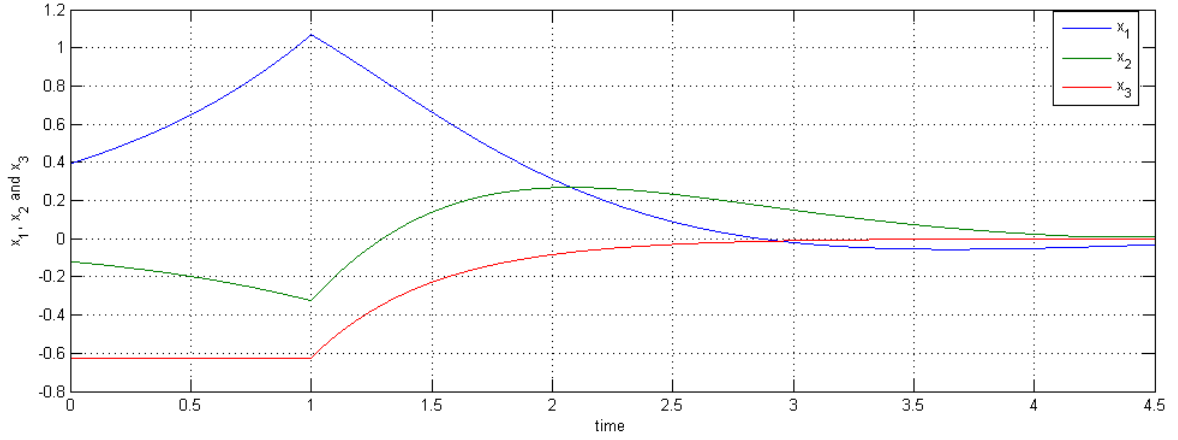


Figure 3.12: Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (x_1, x_2 and x_3 convergence)

Figure 3.12 shows the convergence of the configuration (x_1, x_2, x_3) in Heisenberg space.

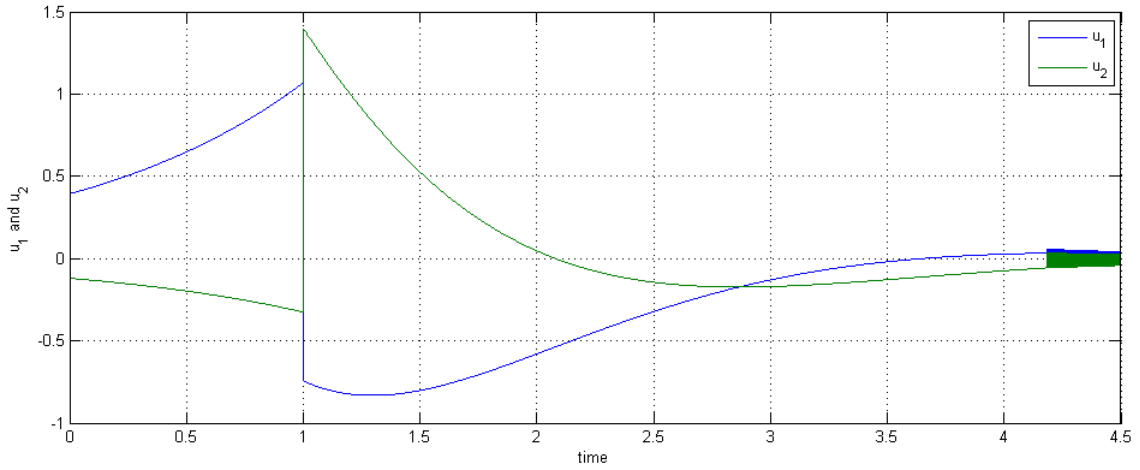


Figure 3.13: Model of Bloch-Drakunov controller with initial conditions $(0, -20, \pi/8)$ (u_1 and u_2 convergence)

Figure 3.13 shows the convergence of the controls (u_1, u_2) in Heisenberg space.

The second simulation had initial conditions: $(x, y, \phi)^T = (20, -20, \frac{\pi}{2})^T$.

Again, the first two plots for this simulation are of the convergence of the system states in configuration space.

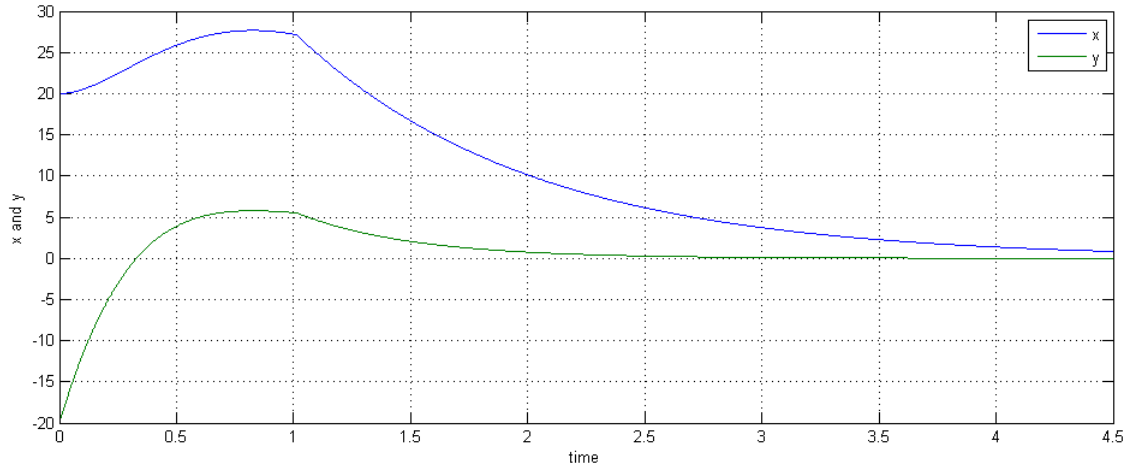


Figure 3.14: Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (x and y convergence)

Figure 3.14 shows the convergence of the location (x, y) in the configuration space.

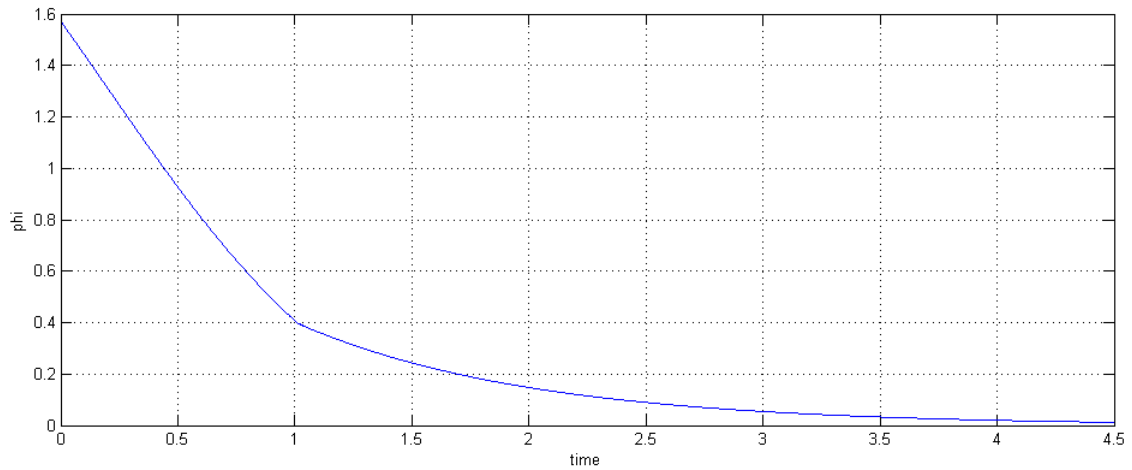


Figure 3.15: Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (ϕ convergence)

Figure 3.15 shows the convergence of the orientation ϕ in the configuration space.

The next two plots show the convergence of the system states in Heisenberg space and the Transformed controls in Heisenberg space.

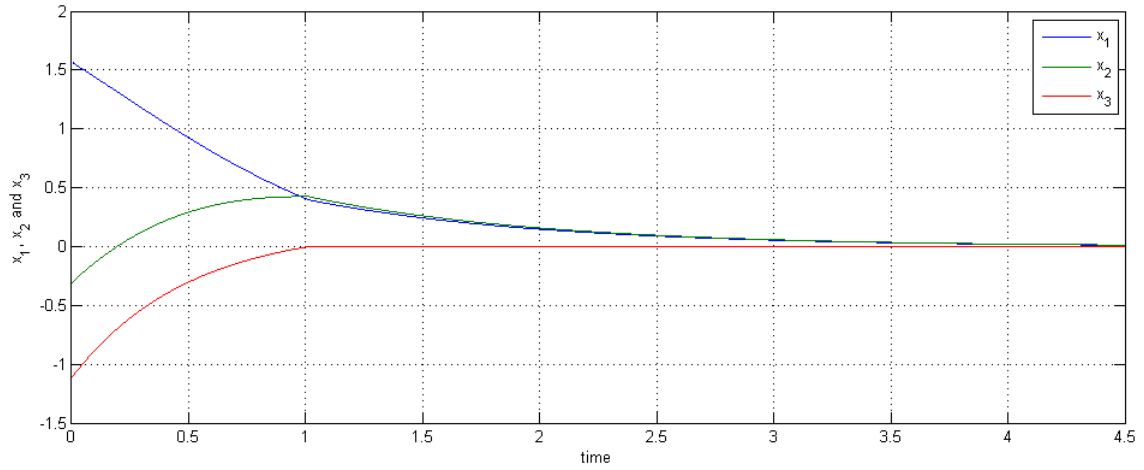


Figure 3.16: Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (x_1 , x_2 and x_3 convergence)

Figure 3.16 shows the convergence of the configuration (x_1, x_2, x_3) in Heisenberg space.

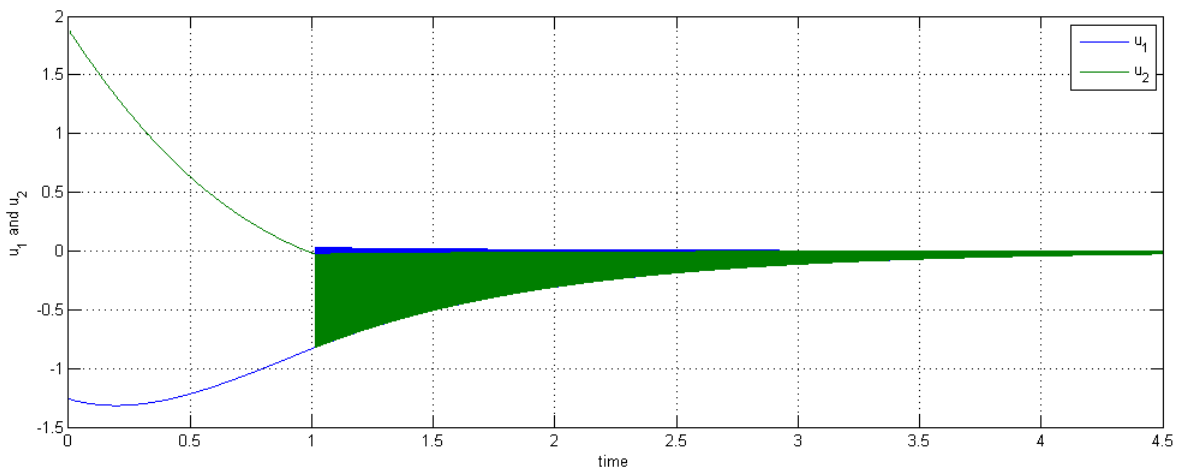


Figure 3.17: Model of Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$ (u_1 and u_2 convergence)

Figure 3.17 shows the convergence of the controls (u_1, u_2) in Heisenberg space.

The next figure, Figure 3.18 is of a simulation of a car following the path produced by the configuration states (x, y, ϕ) .

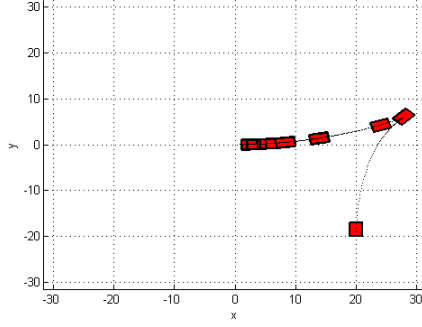


Figure 3.18: Model of car stabilized by Bloch-Drakunov controller with initial conditions $(20, -20, \pi/2)$

3.3 Feedback Control for Systems with Obstacles

3.3.1 Tracking

In the examples given so far in this thesis, all of them have been to drive the system to the origin of the configuration space. This is often done because it simplifies the math. If the intention of a controls problem is to drive the system to some point x^* , then we could always make the following state transformation,

$$\bar{x} = x - x^*. \quad (3.51)$$

The configuration x^* which we are trying to reach could be stationary in which the problem is called a stabilization problem. The target configuration could also be moving $x^* = x^*(t)$. This type of control problem is called a tracking problem. In a tracking problem we try to drive \bar{x} to zero, which causes the configuration x to stay on trajectory $x^*(t)$.

3.3.2 Example: Tracking in the Heisenberg System

Here we will follow an example tracking problem. This is from the same paper Bloch and Drakunov [1996] where Bloch and Drakunov first published their controllers for the Heisenberg system. It tracks some trajectory in Heisenberg space, while using the Bloch-Drakunov controller which we described previously.

Let $x^*(t) = [x_1^*(t), x_2^*(t), x_3^*(t)]^T$ be a smooth curve in the Heisenberg system state space \mathbb{R}^3 . We define a variable \hat{x}_3 as

$$\hat{x}_3(t) = x_3(t) - x_1^*(t)x_2(t) + x_2^*(t)x_1(t). \quad (3.52)$$

Differentiating, and using (3.3) we get

$$\dot{\hat{x}}_3(t) = \dot{x}_3 - \dot{x}_1^*x_2 - x_1^*\dot{x}_2 + \dot{x}_2^*x_1 + x_2^*\dot{x}_1, \quad (3.53)$$

$$= x_1u_2 - x_2u_1 - \dot{x}_1^*x_2 - x_1^*u_2 + \dot{x}_2^*x_1 + x_2^*u_1, \quad (3.54)$$

$$\dot{\hat{x}}_3(t) = \underbrace{(x_1 - x_1^*)}_{\bar{x}_1} \underbrace{(u_2 - \dot{x}_2^*)}_{\bar{u}_2} - \underbrace{(x_2 - x_2^*)}_{\bar{x}_2} \underbrace{(u_1 - \dot{x}_1^*)}_{\bar{u}_1} + \underbrace{2(x_1\dot{x}_2^* - \dot{x}_1^*x_2)}_{\bar{g}} + \underbrace{\dot{x}_1^*x_2^* - \dot{x}_2^*x_1^*}_{\dot{x}_3^*}. \quad (3.55)$$

Let us define a new variable g such that

$$g(t, x_1, x_2) = 2x_1\dot{x}_2^* - \dot{x}_2^*x_1^* - 2x_2\dot{x}_1^* + \dot{x}_1^*x_2^*, \quad (3.56)$$

and \bar{g} such that $\bar{g} = g - \dot{x}_3^*$. Let us also use notation standard for tracking problems, $\bar{x}_i = x_i - x_i^*$, $i = 1, \dots, n$. For the controls we make the substitutions, $\bar{u}_j = u_j - \dot{x}_j^*$, $i, j = 1, \dots, m$. With this notation, $\dot{\hat{x}}_3$ can be expressed as,

$$\dot{\hat{x}}_3(t) = \bar{x}_1\bar{u}_2 - \bar{x}_2\bar{u}_1 + \underbrace{\bar{g} + \dot{x}_3^*}_g. \quad (3.57)$$

If we define, $\dot{\hat{x}}_3 = \dot{\bar{x}}_3 + \dot{x}_3^*$, we get the the Heisenberg system but with tracking variables except for one important difference; The $\dot{\bar{x}}_3$ has a drift term \bar{g} . In terms of

this new variable, we can write $\dot{\hat{x}}_3$ as,

$$\dot{\hat{x}}_1 = \bar{u}_1, \quad (3.58)$$

$$\dot{\hat{x}}_2 = \bar{u}_2, \quad (3.59)$$

$$\dot{\hat{x}}_3 = \bar{x}_1 \bar{u}_2 - \bar{x}_2 \bar{u}_1 + \bar{g}, \quad (3.60)$$

The Bloch-Drakunov controller (see Bloch and Drakunov [1996] and section 3.2.2) in terms of our tracking variables is

$$\bar{u}_1 = -\alpha \bar{x}_1 + \beta \bar{x}_2 \operatorname{sign}(\bar{x}_3), \quad (3.61)$$

$$\bar{u}_2 = -\alpha \bar{x}_2 - \beta \bar{x}_1 \operatorname{sign}(\bar{x}_3), \quad (3.62)$$

and the new system of equations is

$$\dot{\hat{x}}_1 = -\alpha \bar{x}_1 + \beta \bar{x}_2 \operatorname{sign}(\bar{x}_3), \quad (3.63)$$

$$\dot{\hat{x}}_2 = -\alpha \bar{x}_2 - \beta \bar{x}_1 \operatorname{sign}(\bar{x}_3), \quad (3.64)$$

$$\dot{\hat{x}}_3 = -\beta(\bar{x}_1^2 + \bar{x}_2^2) \operatorname{sign}(\bar{x}_3) + \bar{g}. \quad (3.65)$$

If we now used the same switching conditions, (equations, 3.44, 3.45, 3.47) as in section, 3.2.2, to complete the control scheme, we would be able to drive the system to x^* , but once there, the controls would stop. Since this is a tracking problem, the target configuration x^* is continuously changing.

In sliding mode control, the system state is confined the sliding manifold by rapidly switching between two controllers, which steer the system state to the manifold from either side. In addition to driving the system state to the sliding manifold, the controllers should also drive the system in the direction of the target position. With the above control scheme, from one side of the manifold, (outside the paraboloid) the system is driven with directional components pointing both towards the manifold and towards the target position, but from the other side of the manifold, (inside the paraboloid) the system state is only driven to the sliding manifold.

Table 3.2: Bloch-Drakunov Tracking Controller

Controls	$\frac{\beta}{2\alpha} (\bar{x}_1^2 + \bar{x}_2^2) \geq \bar{x}_3 $	$\frac{\beta}{2\alpha} (\bar{x}_1^2 + \bar{x}_2^2) < \bar{x}_3 $	$\bar{x}_3 = 0$
\bar{u}_1	$-\alpha\bar{x}_1 + \beta\bar{x}_2 \text{sign}(\bar{x}_3)$	$\alpha\bar{x}_1$	$-\alpha\bar{x}_1$
\bar{u}_2	$-\alpha\bar{x}_2 - \beta\bar{x}_1 \text{sign}(\bar{x}_3)$	$\alpha\bar{x}_2$	$-\alpha\bar{x}_2$

A more robust control scheme can be obtained by steering the configuration x to some ε -neighborhood of our desired path x^* . In this way, we may chose an ε which confines our system sufficiently close to our path.

The controls are the same as in the stabilizing controller (and Table 3.1) until the system is inside the ε -neighborhood. After that, the controls switch as described in table, (3.3).

Table 3.3: Bloch-Drakunov Tracking Controller

Controls	$\bar{x}_1^2 + \bar{x}_2^2 > \varepsilon^2$,	$\bar{x}_1^2 + \bar{x}_2^2 \leq \varepsilon^2$,
\bar{u}_1	$-\alpha\bar{x}_1 + \beta\bar{x}_2 \text{sign}(\bar{x}_3)$	$\alpha\bar{x}_1 + \beta\bar{x}_2 \text{sign}(\bar{x}_3)$
\bar{u}_2	$-\alpha\bar{x}_2 - \beta\bar{x}_1 \text{sign}(\bar{x}_3)$	$\alpha\bar{x}_2 - \beta\bar{x}_1 \text{sign}(\bar{x}_3)$

Once the system state has reached the ε -neighborhood of the path, the controls will switch between two conditions, and by choosing the value of ε we can effect the rate at which the system chatters. Larger values of ε produce slower chatter, but larger deviations from the path.

Again, we will check for convergence. Let us take the Lyapunov function as $V = \bar{x}_1^2 + \bar{x}_2^2$, which has the derivatives,

$$\dot{V} = \begin{cases} -4\alpha V & \text{if } V > \varepsilon^2 \\ 4\alpha V & \text{if } V \leq \varepsilon^2, \end{cases} \quad (3.66)$$

In the first case, where the system is outside the ε -neighborhood of x^* , the system is Lyapunov stable, in the case where it is inside, the controls drive V to ε^2 . Therefore the system has two sliding surfaces, first it has a paraboloidal sliding manifold in the $(\bar{x}_1, \bar{x}_2, \bar{x}_3)$. Then once it has reached tube around $x^*(t)$ defined by $\bar{x}_1^2 + \bar{x}_2^2 = \varepsilon^2$ it will slide along this tube as it follows the path $x^*(t)$

Once this second sliding manifold is reached, V becomes constant $V = \varepsilon^2$, and equation (3.65) becomes

$$\dot{\bar{x}}_3 = -\beta\varepsilon^2 \operatorname{sign}(\bar{x}_3) + \bar{g}. \quad (3.67)$$

Therefore \bar{x}_3 will converge if a value for β is chosen such that $\beta\varepsilon^2 > |\bar{g}|$.

Chapter 4

Optimal Control

4.1 Pontryagin's Maximum Principle

4.1.1 The General Case

Pontryagin's Maximum Principle is used to find an optimal control when the control is bounded (as opposed to calculus of variations which also gives optimal control but cannot accept bounded control space) Pontryagin et al. [1962].

The term optimal control for the sake of this paper is defined as the control which maximizes a cost functional in the form

$$J = \int_{t^i}^{t^f} f^0(x(t), u(t)) dt \quad (4.1)$$

when steering x from an initial state $x^i = x(t^i)$ to a final state $x^f = x(t^f)$ in the time interval $t \in [t^i, t^f]$, where $u(t)$ is the applied control.

Notice that this is the same as maximizing the performance index given by

$$\mathfrak{J} = - \int_{t^i}^{t^f} f^0(x(t), u(t)) dt. \quad (4.2)$$

Note also, that $u \in U$, where U is the set of admissible controls, and x is the

n -dimensional state variable, such that:

$$u(t) = [u_1(t), u_2(t), \dots, u_r(t)]^T, \quad (4.3)$$

$$x(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T. \quad (4.4)$$

A set of state functions relate the state variables and the control variables:

$$\frac{dx_i}{dt} = f^i(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_r(t)), \quad i = 1, 2, \dots, n \quad (4.5)$$

If the integral (4.1) is split into two integrals where, the first part is integrated from time $t^i = 0$ to time, t , and the second part is integrated from time, t to time, t^f , then both parts become time dependant.

$$J = \int_0^t f^0(x(\tau), u(\tau)) d\tau + \int_t^{t^f} f^0(x(\tau), u(\tau)) d\tau \quad (4.6)$$

Let the first integral in equation (4.6) be defined as $x_0(t)$, such that

$$x_0(t) = \int_{t^i}^t f^0(x(\tau), u(\tau)) d\tau, \quad (4.7)$$

$$J = x_0(t) + \int_t^{t^f} f^0(x(\tau), u(\tau)) d\tau. \quad (4.8)$$

Even though both terms in equation (4.6) are now time dependant, their sum J remains a constant. If equation (4.7) is differentiated, the result is an equation very similar to the state equations:

$$\frac{dx_0}{dt} = f^0(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_r(t)) \quad (4.9)$$

The state equations can be rewritten to include equation (4.9).

$$\frac{dx_i}{dt} = f^i(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_r(t)) \quad i = 0, 1, 2, \dots, n \quad (4.10)$$

In terms of state variables x and u equations (4.10) can be written in the form

$$\frac{dx_i}{dt} = f^i(x(t), u(t)) \quad i = 0, 1, 2, \dots, n. \quad (4.11)$$

Whenever possible the state-variable notation will be used to improve readability.

$$\frac{dx}{dt} = f(x(t), u(t)). \quad (4.12)$$

For readability define $f^i \equiv f^i(x(t), u(t))$. A set of auxiliary functions,

$$\psi = [\psi_0(t), \psi_1(t), \psi_2(t), \dots, \psi_n(t)]^T, \quad (4.13)$$

are now defined such that:

1. Their time derivatives are given by the following equation.

$$\frac{d\psi_i}{dt} = - \sum_{\alpha=0}^n \frac{\partial f^\alpha}{\partial x_i} \psi_\alpha, \quad i = 0, 1, \dots, n. \quad (4.14)$$

2. They are continuous and have continuous derivatives $\psi \in C^2$ almost everywhere. (They will have a finite number of point discontinuities, where the controls switch.) Notice that the differential in equation 4.14 can be pulled out of the summation:

$$\sum_{\alpha=0}^n \frac{\partial f^\alpha}{\partial x_i} \psi_\alpha = \sum_{\alpha=0}^n \frac{\partial}{\partial x_i} (f^\alpha \psi_\alpha) = \frac{\partial}{\partial x_i} \left(\sum_{\alpha=0}^n f^\alpha \psi_\alpha \right) \quad (4.15)$$

Let \mathcal{H} be defined such that

$$\mathcal{H}(\psi, x, u) = \sum_{\alpha=0}^n f^\alpha \psi_\alpha. \quad (4.16)$$

This permits equation 4.14 to be expressed in terms of \mathcal{H} as

$$\frac{d\psi_i}{dt} = - \frac{\partial \mathcal{H}}{\partial x_i}, \quad i = 0, 1, \dots, n. \quad (4.17)$$

Notice that

$$\frac{\partial}{\partial \psi_i} (\psi_\alpha f^\alpha) = \begin{cases} f^0, & \alpha \neq i \\ f^i, & \alpha = i \end{cases} \quad (4.18)$$

Therefore by differentiating equation (4.16) and using equation (4.11)

$$\frac{dx_i}{dt} = \frac{\partial \mathcal{H}}{\partial \psi_i}, \quad i = 0, 1, \dots, n. \quad (4.19)$$

Pontryagin's Maximum Principle, provides the means to find this Hamiltonian function. **Pontryagin's Maximum Principle:**

Here is Pontryagin's Maximum Principle from a translated version of his book Pontryagin et al. [1962]. (The notation has been changed to be consistent with that of this thesis.)

Let $u(t), t^i \leq t \leq t^f$, be an admissible control such that the corresponding trajectory $\mathbf{x}(t)$ [see (4.19)] which begins at the point \mathbf{x}^i at the time t^i passes, at some time t^f , through a point on line 2. In order that $u(t)$ be optimal it is necessary that there exist a nonzero continuous vector function $\psi = (\psi_0(t), \psi_1(t), \psi_2(t), \dots, \psi_n(t))$ corresponding to $u(t)$ and $\mathbf{x}(t)$ [see (4.17)], such that

1° for every $t^i \leq t \leq t^f$, the function $\mathcal{H}(\psi(t), x(t), u)$ of the variable $u \in U$ attains its maximum at the point $u = u(t)$:

$$\mathcal{H}(\psi(t), x(t), u) = \mathfrak{M}(\psi(t), x(t)); \quad (4.20)$$

2° at the terminal time t^f the relations

$$\psi_0(t^f) \leq 0, \quad \mathfrak{M}(\psi(t^f), x(t^f)) = 0 \quad (4.21)$$

are satisfied.

The optimal controls, u^* , are those which maximize the Hamiltonian, in the control space, $u \in U$.

$$\mathfrak{M}(\psi, x) = \sup_{u \in U} \mathcal{H}(\psi, x, u) = \mathcal{H}(\psi, x, u^*) \quad (4.22)$$

4.1.2 The Time-Optimal Case

In equation (4.1) the function $f^0(x(t), u(t))$ determines the quantity that is to be minimized for optimality. If time is to be minimized, J represents the minimal time for the state of the system to transition from x^i to x^f . (If we are thinking in terms of maximizing, we are maximizing the quantity $t^f - t^i$.) In this case $J = t^f - t^i$, i.e:

$$\int_{t^i}^{t^f} f^0(x(t), u(t)) dt = t^f - t^i \quad (4.23)$$

which implies,

$$f^0(x(t), u(t)) = 1. \quad (4.24)$$

There is no reason why the first term of \mathcal{H} can't be pulled out of the sum. In this case, notice that f^0 is equal to one, and \mathcal{H} can be expressed as follows.

$$\mathcal{H}(\psi, x, u) = \psi_0 + \sum_{\alpha=1}^n f^\alpha \psi_\alpha. \quad (4.25)$$

The point of this is to find a Hamiltonian that can be maximized over the control space. Since ψ_0 does not depend on the controls, having ψ_0 added on to the end of the sum in the Hamiltonian, serves as no aid to the purpose of finding the appropriate controls. Therefore, (for this more simplified form of Pontryagin's Maximum Theorem, which is the time optimal case) a simpler form of the Hamiltonian can be defined which does not depend on ψ_0 . Traditionally, the simplified functions from Pontryagin's Maximum Principle, which are only relevant to the time optimal case, are defined with non-curved letters. The following is the hamiltonian for the time optimal case.

$$H(\psi, x, u) = \sum_{\nu=1}^n f^\nu \psi_\nu. \quad (4.26)$$

Likewise to find the auxiliary equations, the first term of equation (4.14) is not needed:

$$\frac{d\psi_i}{dt} = - \sum_{\alpha=1}^n \frac{\partial f^\alpha}{\partial x_i} \psi_\alpha, \quad i = 1, 2, \dots, n. \quad (4.27)$$

also

$$\frac{d\psi_i}{dt} = -\frac{\partial H}{\partial x_i}, \quad i = 1, 2, \dots, n \quad (4.28)$$

$$\frac{dx_i}{dt} = \frac{\partial H}{\partial \psi_i}, \quad i = 1, 2, \dots, n. \quad (4.29)$$

and

$$M(\psi, x) = \sup_{u \in U} H(\psi, x, u) = H(\psi, x, u^*). \quad (4.30)$$

4.2 Applying Maximum Principle in Configuration Space

In this section, the Pontryagin's Maximum Principle for time-optimal control is applied to the same car model described earlier. Just as a reminder, here is the example that will be used:

$$\dot{x} = v \cos \phi \quad (4.31)$$

$$\dot{y} = v \sin \phi \quad (4.32)$$

$$\dot{\phi} = \frac{v}{\ell} \tan \theta \quad (4.33)$$

The control space, for simplicity, will be defined by $u_1 = v$ and $u_2 = \frac{\tan \theta}{\ell}$, and constrained as follows;

$$|u_1| \leq \eta, \quad |u_2| \leq \zeta, \quad (4.34)$$

where η , and ζ are constants which depend on the specifics of the vehicle.

Let the following notation be used, $x_1 = x$, $x_2 = y$ and $x_3 = \phi$.

In terms of these controls, the state functions are:

$$\dot{x}_1 = f^1 = u_1 \cos x_3 \quad (4.35)$$

$$\dot{x}_2 = f^2 = u_2 \sin x_3 \quad (4.36)$$

$$\dot{x}_3 = f^3 = u_1 u_2 \quad (4.37)$$

The auxiliary functions from equation, (4.27), are given by

$$\dot{\psi}_1 = -\frac{\partial f^1}{\partial x_1}\psi_1 - \frac{\partial f^2}{\partial x_1}\psi_2 - \frac{\partial f^3}{\partial x_1}\psi_3, \quad (4.38)$$

$$\dot{\psi}_2 = -\frac{\partial f^1}{\partial x_2}\psi_1 - \frac{\partial f^2}{\partial x_2}\psi_2 - \frac{\partial f^3}{\partial x_2}\psi_3, \quad (4.39)$$

$$\dot{\psi}_3 = -\frac{\partial f^1}{\partial x_3}\psi_1 - \frac{\partial f^2}{\partial x_3}\psi_2 - \frac{\partial f^3}{\partial x_3}\psi_3. \quad (4.40)$$

These evaluate to,

$$\dot{\psi}_1 = 0, \quad (4.41)$$

$$\dot{\psi}_2 = 0, \quad (4.42)$$

$$\dot{\psi}_3 = u_1 \sin(x_3) - u_1 \cos(x_3) = \dot{x}_2 - \dot{x}_1. \quad (4.43)$$

The hamiltonian given by equation, (4.26), is

$$H(\psi, x, u) = u_1 \cos x_3 \psi_1 + u_2 \sin x_3 \psi_2 + u_1 u_2 \psi_3. \quad (4.44)$$

The auxiliary states, $\psi_1 - \psi_3$, are found by integrating equations, (4.41 - 4.43).

$$\psi_1 = \psi(t=0) \equiv \psi_{10}, \quad (4.45)$$

$$\psi_2 = \psi(t=0) \equiv \psi_{20}, \quad (4.46)$$

$$\psi_3 = x_2 - x_1 + \psi_{20} - \psi_{10} + \psi_{30}. \quad (4.47)$$

Where ψ_{10} , ψ_{20} and ψ_{30} are initial conditions. Substituting the results from equations, (4.45 - 4.47), into the Hamiltonian, equation (4.44), produces

$$H(\psi, x, u) = u_1 \cos x_3 \psi_{10} + u_2 \sin x_3 \psi_{20} + u_1 u_2 (x_2 - x_1 + \psi_{20} - \psi_{10} + \psi_{30}). \quad (4.48)$$

4.3 Applying Maximum Principle in Heisenberg Space

4.3.1 Control Derivation

Here we will take the minimum time approach, therefore $J = t^f - t^i$ and $f^0(x(t) u(t)) = 1$ and by equation (4.11), we can obtain the set of differential equations;

$$\dot{x}_0 = f^0 = 1, \quad (4.49)$$

$$\dot{x}_1 = f^1 = u_1, \quad (4.50)$$

$$\dot{x}_2 = f^2 = u_2, \quad (4.51)$$

$$\dot{x}_3 = f^3 = x_1 u_2 - x_2 u_1. \quad (4.52)$$

We may obtain the axillary set of equations with the help of equation, (4.14). They are the following:

$$\dot{\psi}_0 = 0, \quad (4.53)$$

$$\dot{\psi}_1 = -u_2 \psi_3, \quad (4.54)$$

$$\dot{\psi}_2 = -u_1 \psi_3, \quad (4.55)$$

$$\dot{\psi}_3 = 0. \quad (4.56)$$

Applying equation, (4.26), we obtain the Hamiltonian:

$$H = \psi_1 u_1 + \psi_2 u_2 + \psi_3 (x_1 u_2 - x_2 u_1). \quad (4.57)$$

Alternatively, the Hamiltonian can be written in this way:

$$H = u_1 (\psi_1 - \psi_3 x_2) + u_2 (\psi_2 + \psi_3 x_1). \quad (4.58)$$

The controls are found which maximize the function, H .

$$M = \max(H) = \max(u_1(\psi_1 - \psi_3 x_2) + u_2(\psi_2 + \psi_3 x_1)). \quad (4.59)$$

We can consider a normalized control manifold such that, $|u_1| \leq 1$, $|u_2| \leq 1$. Notice that H is at its maximal when controls are chosen such that

$$M = |\psi_1 - \psi_3 x_2| + |\psi_2 + \psi_3 x_1|. \quad (4.60)$$

Which happens when, $|u_1| = 1$, $|u_2| = 1$, and leads to the controls:

$$u_1 = \text{sign}(\psi_1 - \psi_3 x_2), \quad (4.61)$$

$$u_2 = \text{sign}(\psi_2 + \psi_3 x_1). \quad (4.62)$$

Notice, that this is an example of bang-bang control which is always produced when the time-optimal condition is applied with Pontryagin's Maximum Principle and the assumption that condition, (4.24) is made.

4.3.2 Optimal Control as Partial Feedback

Controls, (4.62) can be expressed as

$$u_1 = \text{sign} \left(\psi_3 \left(\frac{\psi_1}{\psi_3} - x_2 \right) \right), \quad (4.63)$$

$$u_2 = \text{sign} \left(\psi_3 \left(\frac{\psi_2}{\psi_3} + x_1 \right) \right). \quad (4.64)$$

Given two arbitrary numbers, a and b , notice that $\text{sign } ab = \text{sign } a \text{ sign } b$. Therefore we can pull the ϕ_3 out as follows,

$$u_1 = \text{sign}(\psi_3) \text{sign} \left(\frac{\psi_1}{\psi_3} - x_2 \right), \quad (4.65)$$

$$u_2 = \text{sign}(\psi_3) \text{sign} \left(\frac{\psi_2}{\psi_3} + x_1 \right). \quad (4.66)$$

Now, notice that $\dot{\psi}_3 = 0$ therefor ψ_3 is a constant and $\psi_3 = \psi_3(t = 0)$. In other words, ψ_3 is an initial condition. As a reminder that ψ_3 is a constant and an initial condition, the following definition is made: $\psi_3 = \psi_3(0) \equiv \psi_{30}$. The control laws now become:

$$u_1 = \text{sign}(\psi_{30}) \text{sign}\left(\frac{\psi_1}{\psi_{30}} - x_2\right), \quad (4.67)$$

$$u_2 = \text{sign}(\psi_{30}) \text{sign}\left(\frac{\psi_2}{\psi_{30}} + x_1\right). \quad (4.68)$$

Also note:

$$\dot{\psi}_1 = -u_2 \psi_{30}, \quad (4.69)$$

$$\dot{\psi}_2 = u_1 \psi_{30}. \quad (4.70)$$

Recall that $u_1 = \dot{x}_1$ and $u_2 = \dot{x}_2$. Thus equations, (4.70), can be rearranged and integrated as follows:

$$\int \frac{d}{dt} \left(\frac{\psi_1}{\psi_{30}} \right) dt = \int -\dot{x}_2 dt, \quad (4.71)$$

$$\int \frac{d}{dt} \left(\frac{\psi_2}{\psi_{30}} \right) dt = \int \dot{x}_1 dt, \quad (4.72)$$

Evaluated, these integrals become:

$$\frac{\psi_1}{\psi_{30}} = -x_2 + c_1, \quad (4.73)$$

$$\frac{\psi_2}{\psi_{30}} = x_1 + c_2. \quad (4.74)$$

Combining equations, (4.68), and (4.74) produces a new set of controls which are dependant on the integration constants, c_1 and c_2 .

$$u_1 = \text{sign}(\psi_{30}) \text{sign}(c_2 - 2x_2), \quad (4.75)$$

$$u_2 = \text{sign}(\psi_{30}) \text{sign}(c_2 + 2x_1). \quad (4.76)$$

Again as in the previous example, the control laws which were produced depend on initial conditions, and therefore cannot be used to create purely state dependent control.

Chapter 5

Variable Structure Feedback Control for Nonholonomic Systems

In this Chapter, an alternative solution to the problem described in section 3.2.1 will be described. In this case the same vehicle is driven to the origin, but rather than using the Heisenberg system, a geometric-phase based technique. After this method is described, Similarities between these two methods will be pointed out. It will be shown that the Heisenberg system approach can be converted into chained form using a geometric-phase based technique. By doing so, a new perspective can be had as to the solution of this system. With this new perspective, a new controller is developed to solve the Heisenberg system. The goal is to find a new controller which solves this same system, but which can more easily be altered to incorporate the obstacles in the problem statement.

The geometric phase technique is a way to effect a phase shift in some direction, by integrating along (or following) a closed path in a subspace which is normal to the direction of the desired shift Bloch et al. [1992].

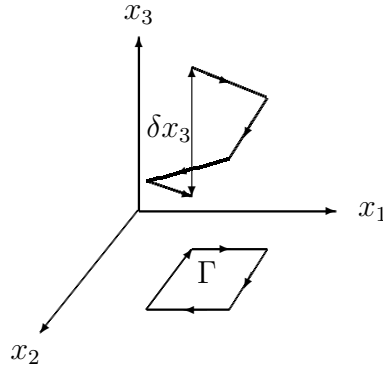


Figure 5.1: Geometric phase technique

Here is the path integral.

$$\delta x_3 = \oint_{\Gamma} \dot{x}_3(t) dt \quad (5.1)$$

Where δx_3 is the distance to be traversed in the x_3 direction. It should equal the initial value of x_3 if driving x to origin. Γ is a closed path in x_1 and x_2 which traverses δx_3 in the whole phase space. In other words, the projection of the path, Γ , on to the (x_1, x_2) plane would be a closed path, but in the three-dimensional Heisenberg space, the two ends of Γ are separated by a distance, δ .

In this section we first examine an example from a literature Bloch et al. [1992]. Next, by comparing this method with the our method of using the Bloch-Drakunov controller, we will develop a hybrid control algorithm which combines the geometric phase technique and the Heisenberg system. The aim of this exercise is that the new technique will lend itself more easily to obstacle avoidance.

5.1 Chained-Form Controller, Example: The Car

The following is an example of a chained-form controller to same car described in figure 3.2. This example is from Bloch et al. [1992], and Al-Ragib.

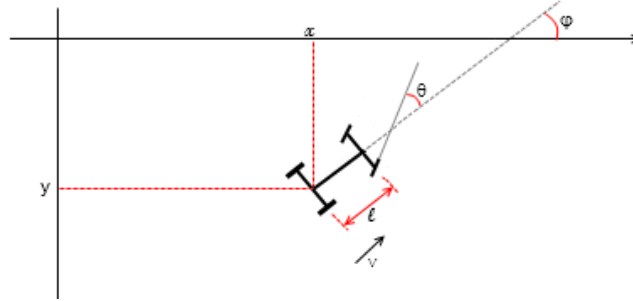


Figure 5.2: Classic nonholonomic car model

The equations of motion are equivalent to (equations, 3.4-3.6), but they are expressed in a way typical of chained form systems.

$$\dot{x} = v \cos \phi \quad (5.2)$$

$$\dot{y} = v \sin \phi \quad (5.3)$$

$$\dot{\phi} = \omega = \frac{v}{\ell} \tan \theta \quad (5.4)$$

$$\dot{v} = v_1 \quad (5.5)$$

$$\dot{\omega} = v_2 \quad (5.6)$$

The transformation into phase space is different than the one given for the examples in Heisenberg space. Bloch et al. [1992]

$$x_1 = x \cos \phi + y \sin \phi \quad (5.7)$$

$$x_2 = \phi \quad (5.8)$$

$$x_3 = -x \sin \phi + y \cos \phi \quad (5.9)$$

$$x_4 = \dot{x} \cos \phi + \dot{y} \sin \phi - \omega(x \sin \phi - y \cos \phi) \quad (5.10)$$

$$x_5 = \omega \quad (5.11)$$

Then we differentiate the phase space variables to try to get a system of equations

that is completely in phase space. We will use the nonholonomic constraint for this system: $\dot{x} \sin \phi - \dot{y} \cos \phi = 0$.

$$\dot{x}_1 = \underbrace{\dot{x} \cos \phi + \dot{y} \sin \phi - x\omega \sin \phi + y\omega \cos \phi}_{x_4} \quad (5.12)$$

$$\dot{x}_2 = \omega \quad (5.13)$$

$$\dot{x}_3 = \underbrace{-\dot{x} \sin \phi + \dot{y} \cos \phi}_{=0 \text{ (nonholonomic constraint)}} \underbrace{-x\omega \cos \phi - y\omega \sin \phi}_{-x_1 x_5} \quad (5.14)$$

$$\dot{x}_4 = \dot{x}_4 \quad (5.15)$$

$$\dot{x}_5 = \dot{\omega} \quad (5.16)$$

This system simplifies as:

$$\dot{x}_1 = x_4 \quad (5.17)$$

$$\dot{x}_2 = x_5 \quad (5.18)$$

$$\dot{x}_3 = -x_1 x_5 \quad (5.19)$$

$$\dot{x}_4 = \ddot{x}_1 \quad (5.20)$$

$$\dot{x}_5 = \ddot{x}_2 \quad (5.21)$$

Or if we use the following notation, $u_1 = \ddot{x}_1, u_2 = \ddot{x}_2$ Then we have the system:

$$\ddot{x}_1 = u_1 \quad (5.22)$$

$$\ddot{x}_2 = u_2 \quad (5.23)$$

$$\dot{x}_3 = -x_1 x_5 \quad (5.24)$$

which is very similar to the Heisenberg system but not symmetric.

For the rest of this example the author used one controller to drive the system to the origin of the (x_1, x_2) plane and a bang-bang control scheme to subsequently

drive x_3 to the origin. The projection of Γ onto the (x_1, x_2) plane was a square.

A solution to the N-car stabilization problem via the chained form method is presented in Sordalen and Wichlund [1993] and Sordalen [1993]

5.2 Hybrid Chain-form and Bloch-Drakunov type Heisenberg System Controller

In the previous section we noticed that the chained form used was very similar to the Heisenberg system. In this section we will follow the same procedure for chained form stabilizers but we will replace that system of equations with the Heisenberg system.

Let us define the variable

$$V = x_1^2 + x_2^2 \tag{5.25}$$

Chained form systems work by first driving V to zero, for example with some simple controls: $\dot{x}_1 = -\alpha x_1, \dot{x}_2 = -\alpha x_2$. Once the system is at the origin of some (x_1, x_2) plane, it is then steered to follow a closed path, Γ on the (x_1, x_2) plane, which causes x_3 to go to zero. This works because we have complete control over the variables, x_1 and x_2 . The challenge is to find a suitable path Γ , Suitability of a path is checked with the integral described above, in equation (5.1).

Recall from the Heisenberg system, (3.3), the expression for \dot{x}_3 , $\dot{x}_3 = x_1\dot{x}_2 - x_2\dot{x}_1$. Applying this to equation, 5.1,

$$\delta x_3 = \oint_{\Gamma} \dot{x}_3(t) dt \tag{5.26}$$

we have

$$\delta x_3 = \oint_{\Gamma} (x_1 \dot{x}_2 - x_2 \dot{x}_1) dt \quad (5.27)$$

$$= \oint_{\Gamma} x_1 d\dot{x}_2 dt - \oint_{\Gamma} x_2 d\dot{x}_1 dt \quad (5.28)$$

$$= \oint_{\Gamma} x_1 dx_2 - \oint_{\Gamma} x_2 dx_1. \quad (5.29)$$

Therefore,

$$x_3(t_f) = x_3(t_i) + \oint_{\Gamma} x_1 dx_2 - \oint_{\Gamma} x_2 dx_1. \quad (5.30)$$

If we stabilize to the origin, such that $x_3(t_f) = 0$, we have,

$$x_3(t_i) = \oint_{\Gamma} x_2 dx_1 - \oint_{\Gamma} x_1 dx_2. \quad (5.31)$$

Let Γ be the circle described by

$$x_1(s)^2 + x_2(s)^2 = R^2 \quad (5.32)$$

Where R is the radius and s is the distance traveled along the circular path Γ in the three dimensional phase space

Note that Bang-Bang control, obtained by applying the Pontryagin's Maximum Principle for the time-optimal case, often produces a rectangular Γ . A rectangular path is perfectly acceptable as well, but for this example we chose to use a circular path for simplicity. Bang-Bang controllers and Pontryagin's Maximum Principle are discussed in the optimal control chapter.

Using the following trigonometric substitutions, $x_1 = R \sin s$ and $x_2 = R \cos s$ and

integrating equation, (5.31), we get

$$x_3(t_i) = \oint_{\Gamma} R^2 \cos^2 s \, ds + \oint_{\Gamma} R^2 \sin^2 s \, ds \quad (5.33)$$

$$x_3(t_i) = \oint_{\Gamma} R^2 (\cos^2 s + \sin^2 s) \, ds, \quad (5.34)$$

$$x_3(t_i) = R^2 s_f, \quad (5.35)$$

where s_f is the length of Γ . We obtained the relation between the phase shift we require, δx_3 , in our case just $x_3(t_i)$, and the Radius of the circular path to follow on the (x_1, x_2) plane.

This was assuming that we only wished to make one full rotation, but if we needed to, we could have made more than one rotation in that case since Γ is a helix which traverses $2n\pi$ radians, we find the length of this helix.

$$s_f = \sqrt{(\delta x_3)^2 + (2n\pi R)^2} \quad (5.36)$$

Controls	$(x_2^2 - x_1^2)\beta \operatorname{sign}(x_3) \geq 0$	$(x_2^2 - x_1^2)\beta \operatorname{sign}(x_3) < 0$	$x_3 = 0$
u_1	$-\alpha x_2 + \beta x_1 \operatorname{sign}(x_3)$	$\alpha x_2 - \beta x_1 \operatorname{sign}(x_3)$	$-\alpha x_1$
u_2	$\alpha x_1 - \beta x_2 \operatorname{sign}(x_3)$	$-\alpha x_1 + \beta x_2 \operatorname{sign}(x_3)$	$-\alpha x_2$

Table 5.1: Altered and Bloch-Drakunov-Heisenberg Controller

5.3 Simulink Model of Altered Controller

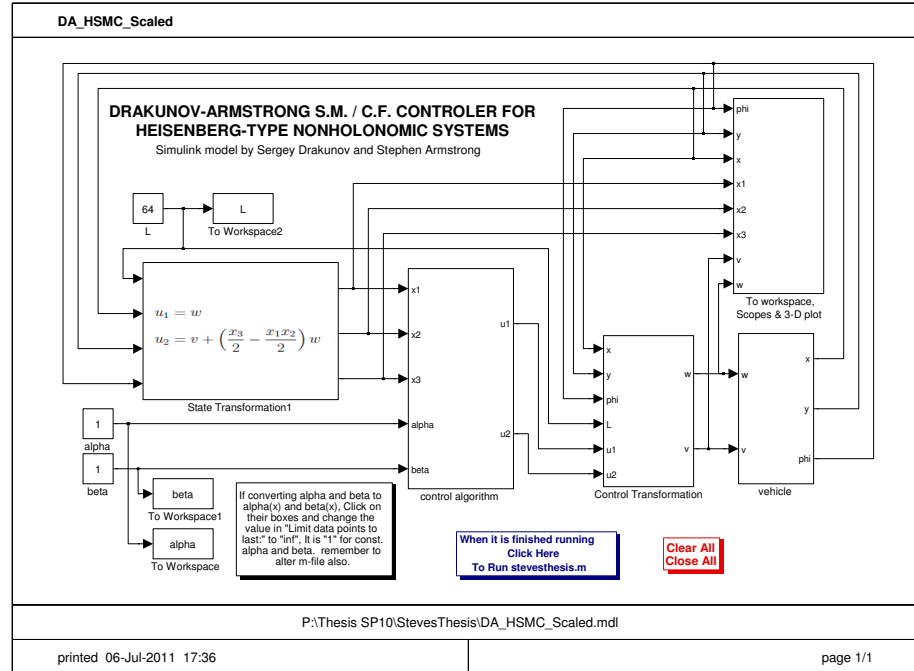


Figure 5.3: Altered Bloch-Drakunov controller modeled with Simulink: main window

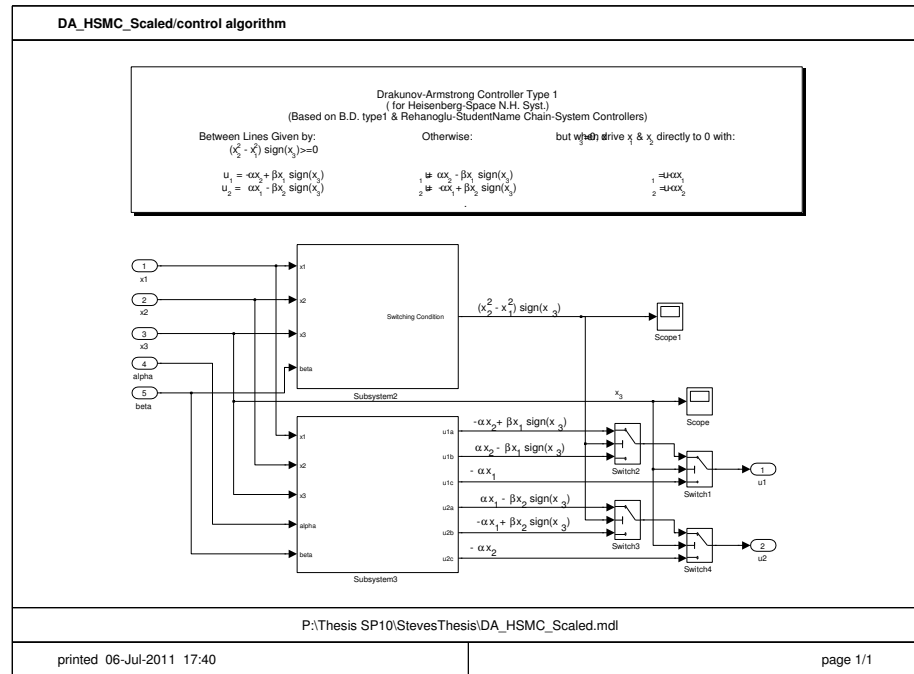


Figure 5.4: Altered Bloch-Drakunov controller modeled with Simulink: control algorithm

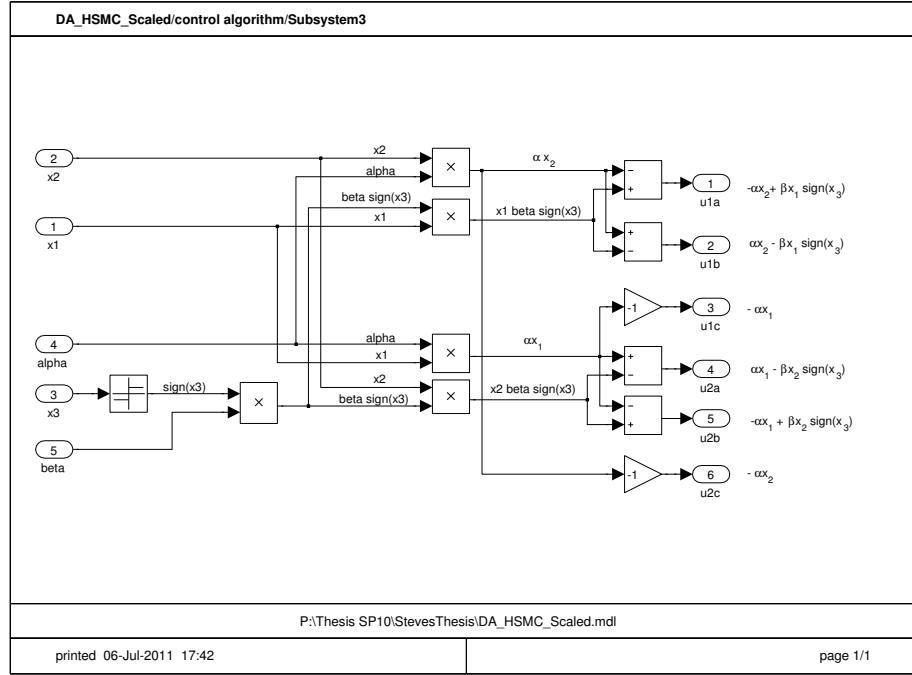


Figure 5.5: Altered Bloch-Drakunov controller modeled with Simulink: controllers a & b

Simulation Results

As with the Bloch Drakunov controller, this control scheme works for unobstructed paths. Again we will show the convergence of two different simulations.

The first simulation had initial conditions: $(x, y, \phi)^T = (20, 0, \frac{\pi}{2})^T$.

The first two plots are of the convergence of the system states in configuration space.

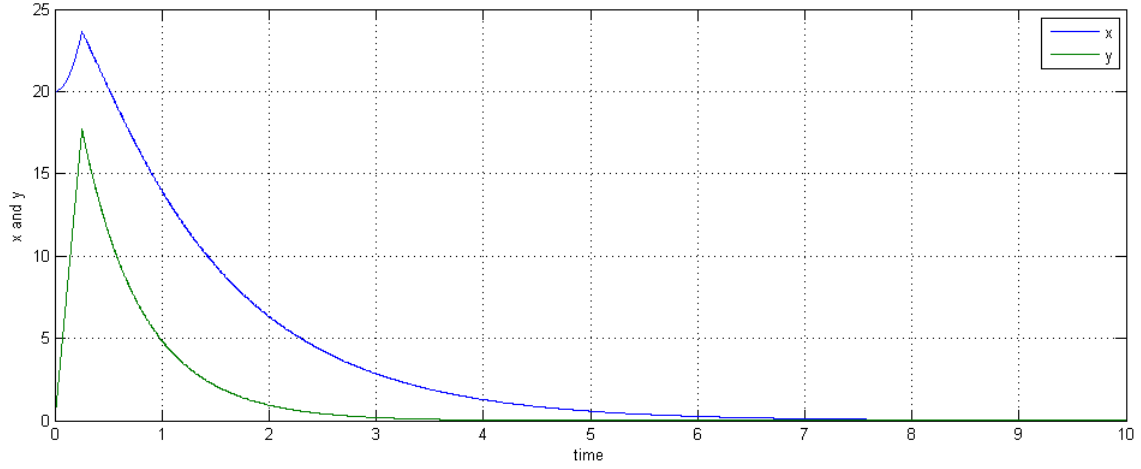


Figure 5.6: Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (x and y convergence)

Figure 5.6 shows the convergence of the location (x, y) in the configuration space.

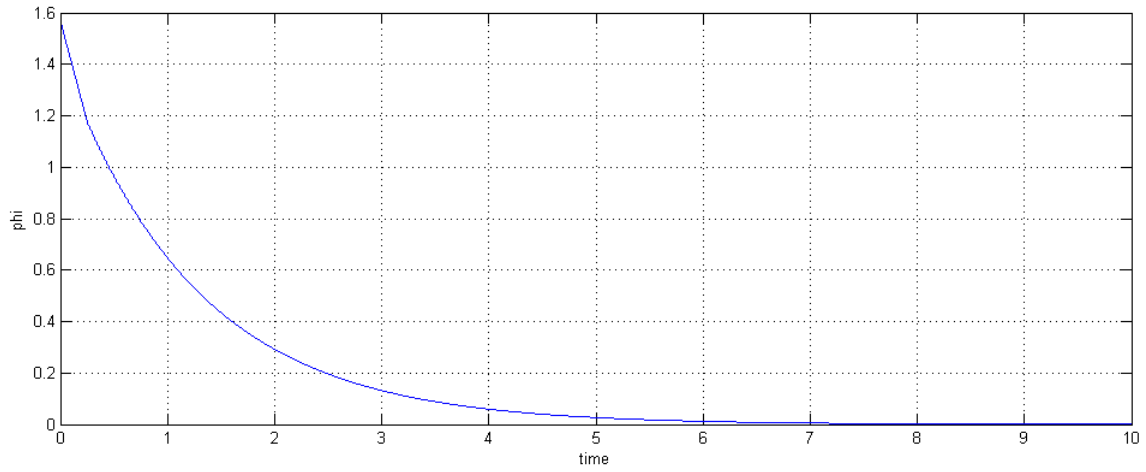


Figure 5.7: Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (ϕ convergence)

Figure 5.7 shows the convergence of the orientation ϕ in the configuration space.

The next two plots show the convergence of the system states in Heisenberg space and the Transformed controls in Heisenberg space.

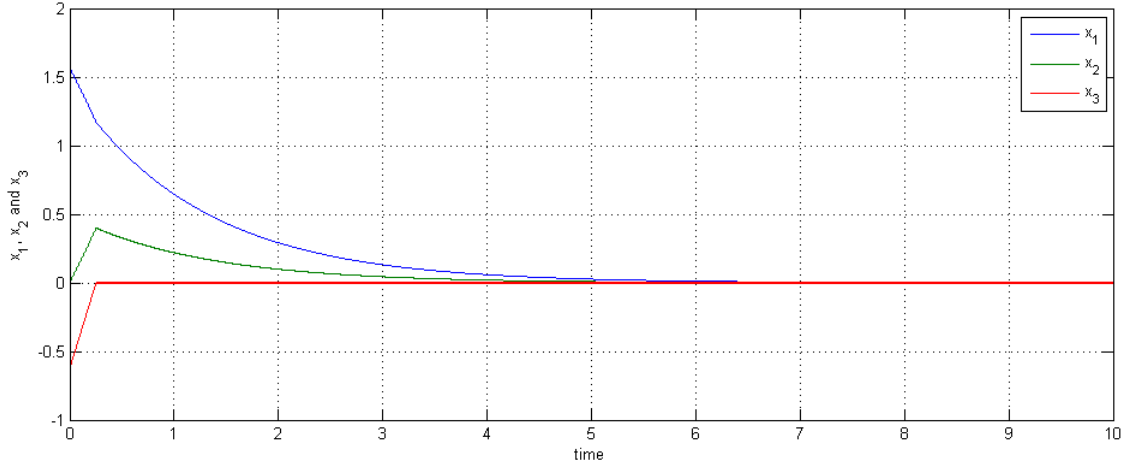


Figure 5.8: Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (x_1 , x_2 and x_3 convergence)

Figure 5.8 shows the convergence of the configuration (x_1, x_2, x_3) in Heisenberg space.

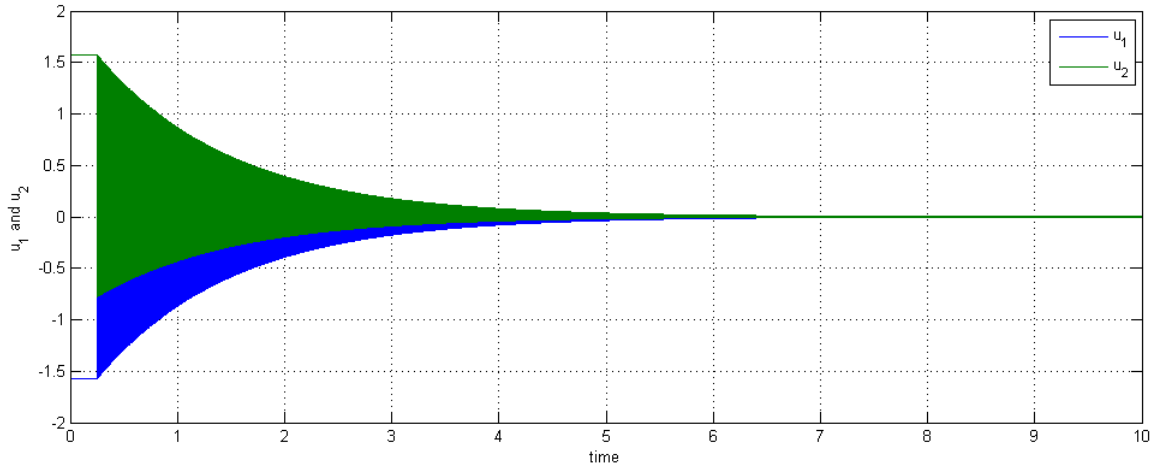


Figure 5.9: Model of Drakunov-Armstrong controller with initial conditions $(20, 0, \pi/2)$ (u_1 and u_2 convergence)

Figure 5.9 shows the convergence of the controls (u_1, u_2) in Heisenberg space.

The second simulation had initial conditions: $(x, y, phi)^T = (-15, -24, \pi)^T$.

Again, the first two plots for this simulation are of the convergence of the system states in configuration space.

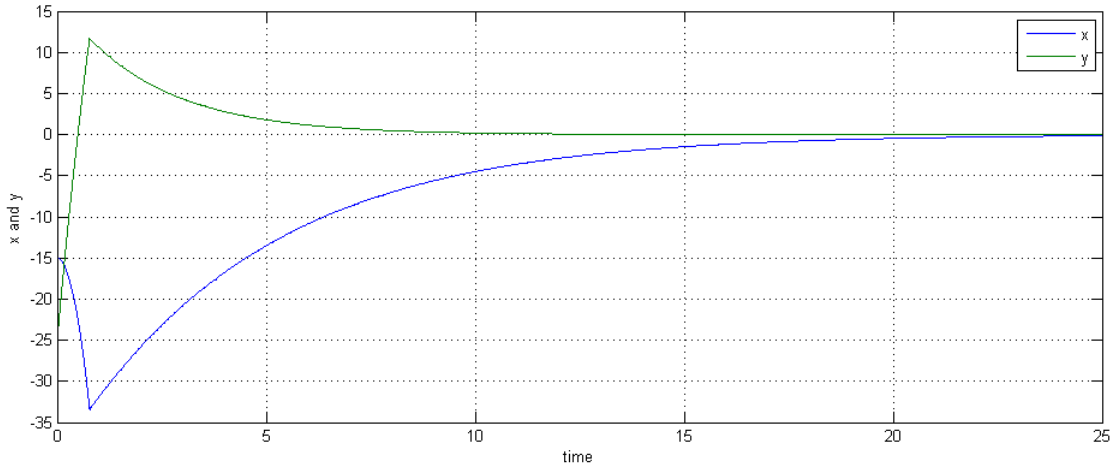


Figure 5.10: Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (x and y convergence)

Figure 5.10 shows the convergence of the location (x, y) in the configuration space.

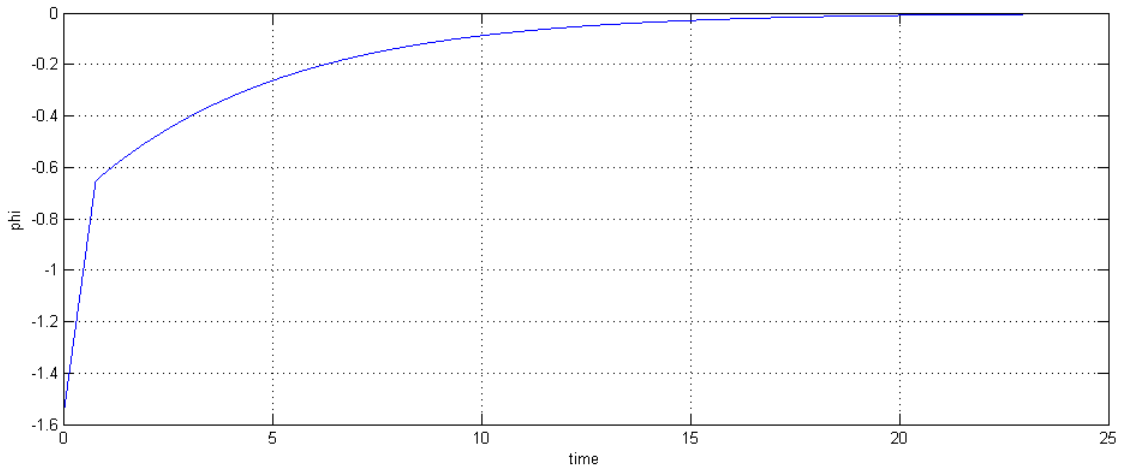


Figure 5.11: Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (ϕ convergence)

Figure 5.11 shows the convergence of the orientation ϕ in the configuration space.

The next two plots show the convergence of the system states in Heisenberg space and the Transformed controls in Heisenberg space.

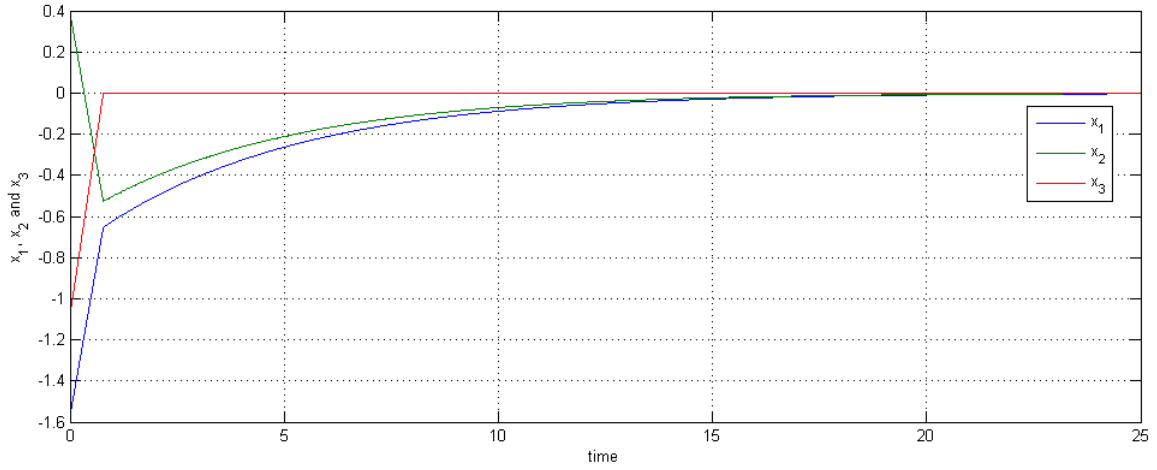


Figure 5.12: Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (x_1 , x_2 and x_3 convergence)

Figure 5.12 shows the convergence of the configuration (x_1, x_2, x_3) in Heisenberg space.

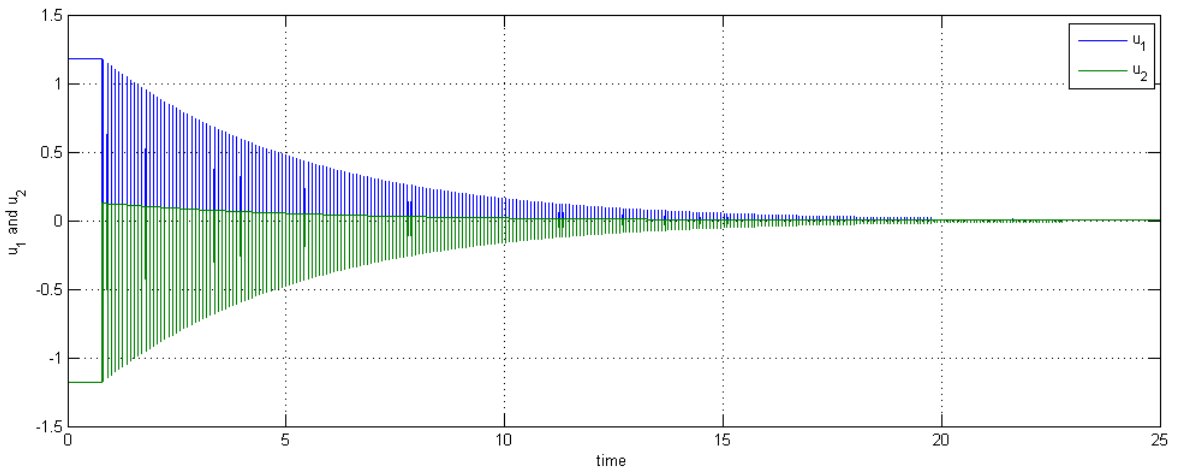


Figure 5.13: Model of Drakunov-Armstrong controller with initial conditions $(-15, -24, \pi)$ (u_1 and u_2 convergence)

Figure 5.13 shows the convergence of the controls (u_1, u_2) in Heisenberg space.

This controller converged slower than the Bloch-Drakunov controller, and was not an improvement.

Chapter 6

Potential Field Methods and Their Applications to Nonholonomic Systems

6.1 Potential Field Method for Obstacle Avoidance in Robotic Systems

Potential field method is a classical approach to path planning for robotic systems among stationary obstacles. By applying a virtual positive charge to each obstacle and a virtual negative charge to the goal location, the control is then designed in such a way that, the vehicle can be guided to follow the gradient of the potential field to the goal location. Orientation is usually ignored because it is assumed that the vehicle could simply be turned on its axis.

Potential field methods are often used in path planning problems, but they can also be applied to feedback control. For example, a continually updating gradient field could provide directional control in a closed loop.

A drawback to potential field methods is that they can produce local minima capable of trapping system. In path planning problems, these local minima can be avoided using a heuristic approach, by checking each potential path. Unfortunately

heuristic approaches are not practical for feedback control, so for feedback control with potential field methods, local minima are a real problem.

In this chapter we will attempt to use potential field methods to solve the non-holonomic feedback problem. To first gain some perspective we will examine a simple problem with only holonomic constraints. Then we will attempt to alter this method to work with a nonholonomic system.

A good source for background information on potential field methods is Khosla and Volpe [1988].

6.2 Potential Field Method for Obstacle Avoidance in Systems Without Nonholonomic Constraints

An example solution of the holonomic problem among movable boundaries using the potential field method is shown below.

In this example, we consider a holonomic mobile robot which moves on a bounded two dimensional plane. There are obstacles on the plane, and they are not required to remain stationary. The configuration space $\mathcal{CS} \in \mathbb{R}^2$ is just two dimensional on the (x, y) -plane. It is assumed that the present location of all objects (obstacles) in the configuration space is known.

A charge density ρ is applied to the boundaries of all obstacles, \mathfrak{O} and to the boundaries of the configuration space. A point charge, c^* is applied to the goal configuration, $X^* = (q_1^*, q_2^*)$, which for this example is just a location in the (x, y) -plane, where $X^* \in \mathcal{CS}$.

The state-dependant equations of motion are as follows,

$$\dot{x} = v \frac{\nabla p_x}{\|\nabla p\|} \quad \dot{y} = v \frac{\nabla p_y}{\|\nabla p\|} \quad (6.1)$$

where, $x, y \in \mathcal{CS}$. The speed v is a constant. p is the potential field caused by the charge densities, ρ on the boundaries and obstacles and by the point charge c^* at the

goal position. ∇p_x is the x -directional component of the gradient of the potential field. $\|\nabla p\|$ magnitude of the gradient of the potential field.

The potential field, p can be calculated as follows,

$$p(x, y) = \iint_{CS} \frac{\rho(\tilde{x}, \tilde{y})}{\sqrt{(x - \tilde{x})^2 + (y - \tilde{y})^2}} d\tilde{x}d\tilde{y} - \frac{c^*}{\sqrt{(q_1^* - x)^2 + (q_2^* - y)^2}} \quad (6.2)$$

6.2.1 Simulated Gradient of 2D Potential Field with Image Processing

In the above example it was assumed that the knowledge of the location of obstacles in the configuration space, would come in the form of an image.

An example black and white image was created, where obstacles were represented as black shapes on a white background. Some simple image processing code was written which

1. Scales the image to the length and width of the 2-dimensional configuration space
2. Finds the boundaries of these obstacles
3. Applies a positive charge density to these boundaries
4. Applies a negative point charge at the goal point
5. Calculates a potential field everywhere
6. Calculates the gradient of the potential field everywhere
7. Plots the steps and results.

The image first had to be scaled to the appropriate dimensions of the configuration space. The image scaling caused the edges to blur so then were sharpened. The sharpening was done by assigning a threshold value on the grayscale and making everything white or black again. (The image was inverted only because it was convenient for the calculations, dealing with the bitmap file format.) Next the boundaries were found by searching for pixels where one of the neighboring pixels had a different value. Positive charges were then assigned to white pixels and a stronger negative charge was assigned to the goal point. It was found that the negative charge should

be approximately three times the sum of all positive charges, in order to produce significant resolution in the potential field gradient. Finally the potential field and gradient was calculated everywhere in \mathcal{CS} .

These steps are depicted figure, 6.1,

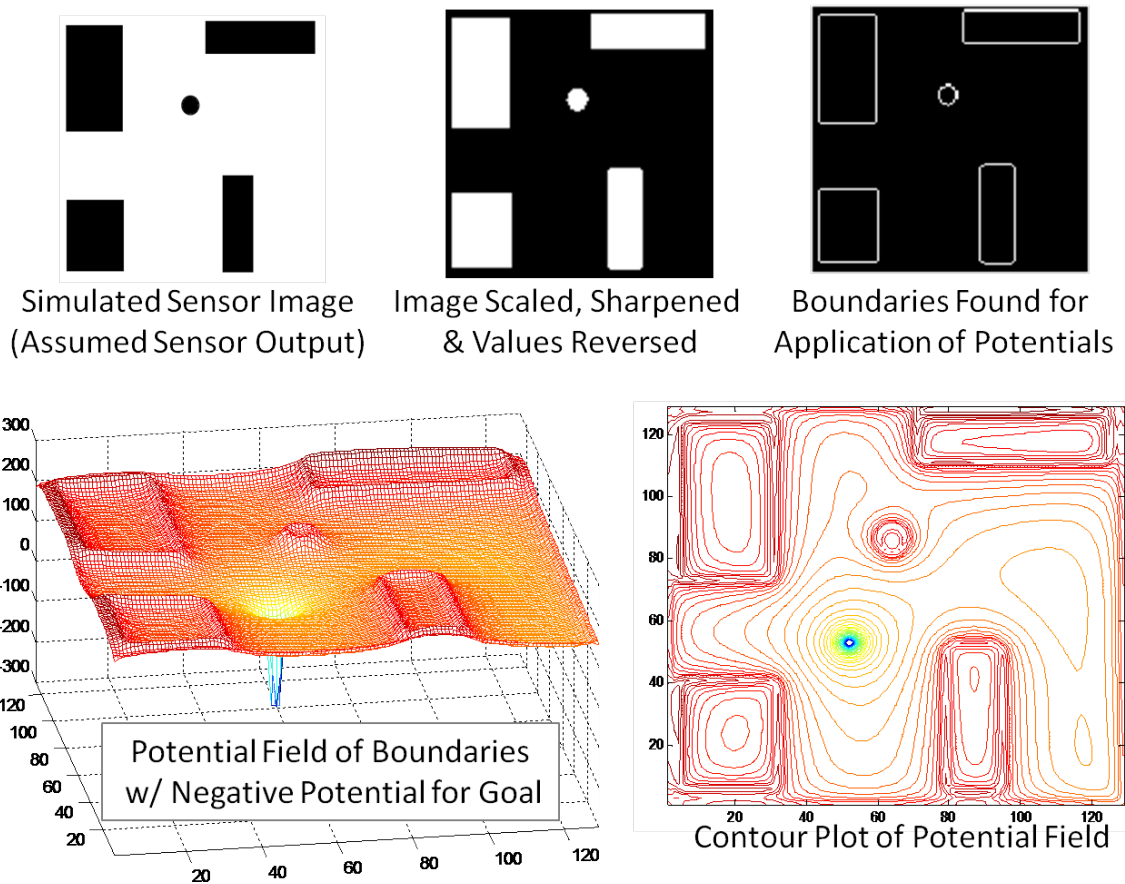


Figure 6.1: Producing a 2-dimensional potential field; the image processing steps

The contours of the potential field are shown above. A holonomic vehicle would follow the gradient of this field from any point to the goal location.

6.3 Transforming Potentials into Heisenberg Space with Paraboloidal Sliding Surface

We used the transformation from section (3.2.2) to see what the obstacles from the previous example, would look like in the Heisenberg Space. Recall that in this transformation, $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = T(\phi) \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}$, where, T is given by equation (3.8),

$$T(\phi) = \begin{bmatrix} 0 & \cos \phi & \sin \phi \\ \phi \cos \phi - 2 \sin \phi & \phi \sin \phi + 2 \cos \phi & 0 \end{bmatrix}$$

ϕ becomes x_1 , so it may be expected that any cross-section of x_2 and x_3 should resemble a skewed form of the original image of the obstacles in the (x, y) -plane. Figure, 6.2 shows what is produced.

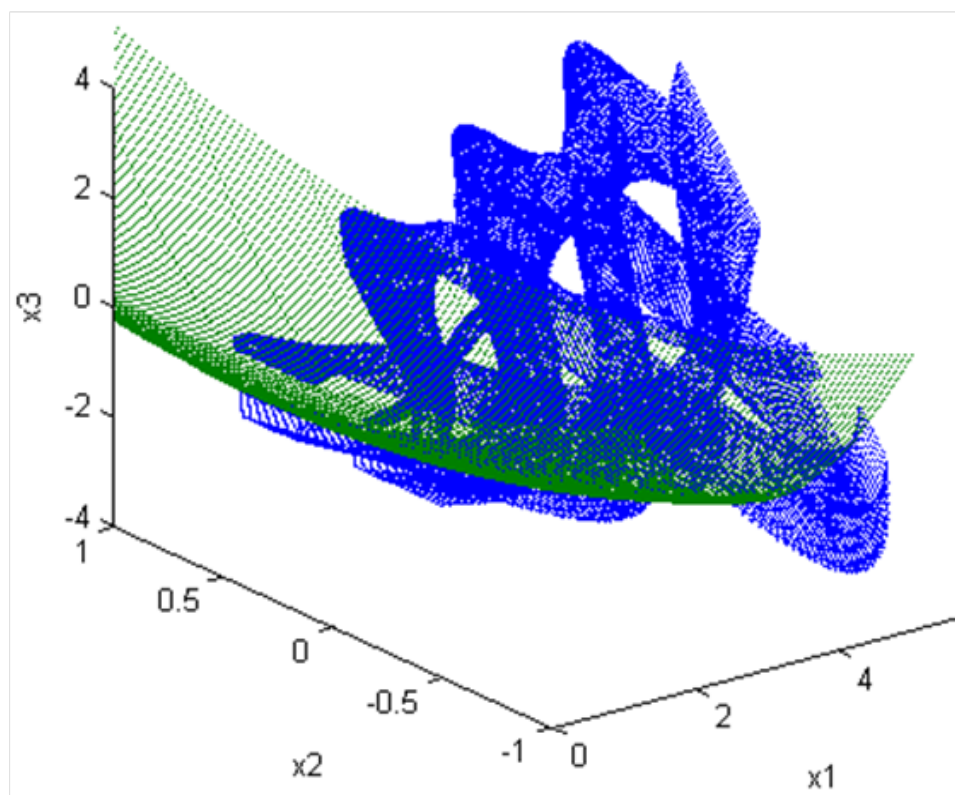


Figure 6.2: Simulated 2-D. obstacles transformed into Heisenberg space

6.4 Wrapping a Potential Field on a Sliding Manifold Using Geodesic Distance

This approach is a modification of the method described by Bloch and Drakunov. In this approach the Lyapunov function only includes two of the three dimensions of the transformed (Heisenberg) space; x_2 and x_3 . The controls are chosen to drive x_1 to zero at a rate faster than the Lyapunov function converges to zero. These controls take on the form of a switching function that maintains the state on a sliding surface. In the 3D Heisenberg space this sliding surface has the form of a paraboloid. The transformed obstacles have the effect of cutting areas out of this sliding surface. The boundaries of the intersection of the transformed obstacles on the sliding surface are applied with a charge density. The vertex of the paraboloid is also given a charge of opposite sign.

A two-dimensional electrostatic potential is calculated everywhere on this manifold by calculating the minimal geodesic distance from each charged point to each point on the surface.

The aim was to create a sliding mode controller which would keep the state on this surface while following the gradient of the potential field to the origin. The variable structure controller would have to have additional controllers to deal with the situation when the car must back up to go around an obstacle. It would also have to account for the maximum steering angle constraint.

Let \mathcal{LT} be the (x, y) -plane in the configuration space, \mathcal{CS} . Then $\mathcal{LT} \subset \mathcal{CS}$.

let $\gamma(x, y)$ represent the location of the borders of all obstacles in the (x, y) -plane, \mathcal{LT} . The set $\gamma(x, y)$ was mapped to the whole configuration space, \mathcal{CS} , with $\chi(x_i, y_j, \phi) = \gamma(x_i, y_j)$ for all points $(x_i, y_j) \in \mathcal{LT}$ and for all $\phi \in \mathcal{CS}$. I.e.

$$\{\chi_\phi \in \mathcal{CS}\} = \{\gamma \in \mathcal{LT}\} \forall \phi \in \mathcal{CS} \quad (6.3)$$

Next, $\chi(x, y, \phi)$ is mapped to Heisenberg space, H . This is done with the transformation matrix, T equation, (3.8), as follows, $\mu = T\chi$. Where $\chi \mapsto \mathcal{CS}$ and $\mu \mapsto H$.

Then two paraboloidal surfaces are created in the Heisenberg space, which share

a common vertex at the origin of \mathfrak{H} , and which have the form, $\frac{\beta}{2\alpha} (x_1^2 + x_2^2) = |x_3|$. A useful depiction of these paraboloids is given in Bloch and Drakunov [1996].

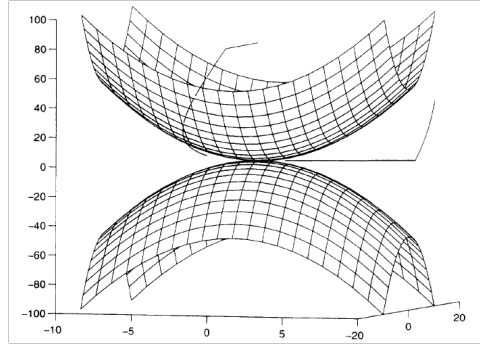


Figure 6.3: Original $\frac{\beta}{2\alpha} (x_1^2 + x_2^2) = |x_3|$ paraboloid drawing from Bloch and Drakunov [1996]

The values of α , and β , which could be functions of the control scheme were, for the sake of simulations in this thesis, taken to be unity.

A negative point charge c^* is applied at the origin of Heisenberg space. Figure, 6.2 shows what this looks like for the same obstacles analyzed in section, 6.2.1

Let \mathfrak{P} be the manifolds defined by $\frac{\beta}{2\alpha} (x_1^2 + x_2^2) = |x_3|$. The potential field was set up as follows. A positive charge density $\rho(x) \in \mathbb{H}$ was applied to the intersections of these paraboloidal surfaces, \mathfrak{P} and the transformed obstacles, μ . Therefore, $\rho = \mathfrak{P} \cap \mu$.

Now curves representing charge densities on the surface of two paraboloidal manifolds. Notice that ρ is entirely in \mathfrak{P} , $\rho \in P$.

We want to create a potential field where the distances considered are along curves which are on the manifold \mathfrak{P} , since these distances have to take into account the curvature of the manifold, we will have to use the minimum geodesic distances.

Here is a picture of what we are trying to accomplish, (figure, 6.4).

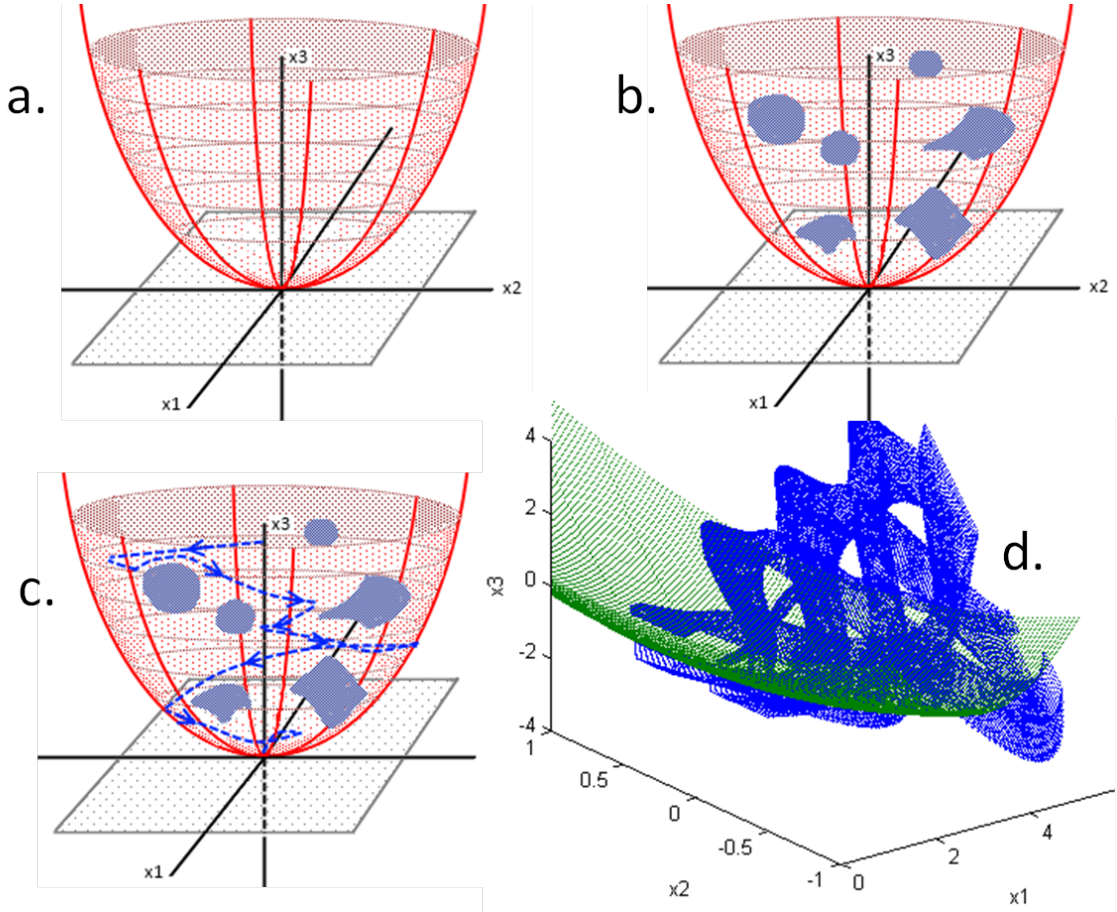


Figure 6.4: Wrapping potential field on to sliding manifold for feedback switching conditions

A convenient way to simplify geodesic problems is to take advantage of symmetry, if there is any. In the case of our system, which is just two paraboloids, we have symmetry around the x_3 -axis and around the $x_3 = 0$ plane. Let us define a local set of coordinates on the surface of the manifold, \mathfrak{P} . We will use the x_3 axis, which we will call ζ as one coordinate and the other coordinate, ψ will represent the counter clockwise angle around the x_3 -axis from the positive x_2 -axis. Therefore we have the following coordinate transformations, $x_1 = \sqrt{\frac{\alpha\zeta}{\beta}} \cos \psi$, $x_2 = \sqrt{\frac{\alpha\zeta}{\beta}} \sin \psi$, $x_3 = \zeta$.

An equation for a minimum geodesic arc length of a surface in three dimensional

cartesian coordinates is given by the following expression Weinstock [1974].

$$\mathfrak{l} = \int_a^b \sqrt{(dx_1)^2 + (dx_2)^2 + (dx_3)^2} \quad (6.4)$$

In terms of the local coordinates, (ζ, ψ) , expression (6.4) becomes, Weinstock [1974]

$$\mathfrak{l} = \int_a^b L d\zeta \quad (6.5)$$

Where L is given by, Weinstock [1974]

$$L = \sqrt{P + 2Q\dot{\psi} + R\dot{\psi}^2} \quad (6.6)$$

and

$$\dot{\psi} = \frac{d\psi}{d\zeta} \quad (6.7)$$

$$P = \left(\frac{\partial x_1}{\partial \zeta} \right)^2 + \left(\frac{\partial x_2}{\partial \zeta} \right)^2 + \left(\frac{\partial x_3}{\partial \zeta} \right)^2 \quad (6.8)$$

$$Q = \frac{\partial x_1}{\partial \zeta} \frac{\partial x_1}{\partial \psi} + \frac{\partial x_2}{\partial \zeta} \frac{\partial x_2}{\partial \psi} + \frac{\partial x_3}{\partial \zeta} \frac{\partial x_3}{\partial \psi} \quad (6.9)$$

$$R = \left(\frac{\partial x_1}{\partial \psi} \right)^2 + \left(\frac{\partial x_2}{\partial \psi} \right)^2 + \left(\frac{\partial x_3}{\partial \psi} \right)^2 \quad (6.10)$$

The integral (6.5) can be solved with the Euler-Lagrange equation,

$$\frac{dL}{d\psi} - \frac{d}{d\zeta} \left(\frac{L}{\dot{\psi}} \right) = 0 \quad (6.11)$$

which becomes, Weinstock [1974]

$$\frac{\frac{\partial P}{\partial \psi} + 2\frac{\partial Q}{\partial \psi}\dot{\psi} + \frac{\partial R}{\partial \psi}\dot{\psi}^2}{2\sqrt{P + 2Q\dot{\psi} + R\dot{\psi}^2}} - \frac{d}{d\zeta} \left(\frac{Q + R\dot{\psi}}{\sqrt{P + 2Q\dot{\psi} + R\dot{\psi}^2}} \right) = 0 \quad (6.12)$$

In the case of our paraboloid we have,

$$P = \frac{\alpha \cos^2 \psi}{4\beta\zeta} + \frac{\alpha \sin^2 \psi}{4\beta\zeta} + 1 = 1 + \frac{\alpha}{4\beta\zeta} \quad (6.13)$$

$$Q = -\frac{\sin \psi}{2\sqrt{\beta\zeta/\alpha}} + \frac{\cos \psi}{2\sqrt{\beta\zeta/\alpha}} = \frac{\cos \psi - \sin \psi}{2\sqrt{\beta\zeta/\alpha}} \quad (6.14)$$

$$R = \frac{\beta}{\alpha}\zeta \cos^2 \psi + \frac{\beta}{\alpha}\zeta \cos^2 \psi + 0 = \frac{\beta}{\alpha}\zeta \quad (6.15)$$

For our case, the solution to the Euler-Lagrange equation is given by Weinstock [1974], as,

$$\zeta - \frac{\alpha}{\beta}c^2 = \zeta(1 + 4c^2) \sin^2 \left(\psi - 2 \ln k \left(\left[2\sqrt{\frac{\beta}{\alpha}\zeta - c^2} + \sqrt{4\frac{\beta}{\alpha}\zeta + 1} \right] \right) \right), \quad (6.16)$$

where c and k are constants. The way that each minimum geodesic arc length, \mathfrak{l} , is obtained is by first, evaluating equation, (6.16), at both end points, (ζ_a, ψ_a) and (ζ_b, ψ_b) . Next solve the two equations simultaneously for c , and k

Let us consider the computational requirements to produce a gradient of this potential field on our manifold. If the charge distribution on the manifold, \mathfrak{P} , was resolved with n pixels, then calculating the potential field everywhere, would require, finding the minimum geodesic distance from each point to each point. This would take $(n-1)^2$ calculations of minimum geodesic arc length, and all of these calculations would have to be done before the gradient could be calculated.

Without an analytical solution to this integral, this approach would be too computationally intensive to be practical for a feedback controller.

6.5 Including Potential in the Lyapunov Function for Obstacle Avoidance

In this attempt, the controls will be taken exactly as they are from Bloch[03 pg.285] Bloch [2003] but the Lyapunov function will be altered to include the boundaries of the obstacles. The Lyapunov function is taken as a function of only x_1 , x_2 and

distance from obstacles on constant x_3 plane.

$$V = a(x_1^2 + x_2^2) - b \iint_{\mathcal{H}_2(x_3)} \{ (x_1 - \tilde{x}_1)^2 + (x_2 - \tilde{x}_2)^2 \} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.17)$$

Where:

$\mathcal{H}_2(x_3)$ is a plane of constant x_3

$(\tilde{x}_1, \tilde{x}_2)$ is an arbitrary point on $\mathcal{H}_2(x_3)$

$\rho(\tilde{x}_1, \tilde{x}_2)$ is $\begin{cases} 1 & \text{if } (\tilde{x}_1, \tilde{x}_2) \text{ lies on the boundary of an obstacle} \\ 0 & \text{if elsewhere} \end{cases}$

a is a constant weighing factor or "charge" of $(0, 0, x_3)$

b is a weighing factor of all combined obstacle charges on $\mathcal{H}_2(x_3)$

A necessary condition for asymptotic stability is that the time derivative of the Lyapunov function must be negative.

$$\dot{V} = 2ax_1\dot{x}_1 + 2ax_2\dot{x}_2 - b \frac{d}{dt} \iint_{\mathcal{H}_2(x_3)} \{ (x_1 - \tilde{x}_1)^2 + (x_2 - \tilde{x}_2)^2 \} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.18)$$

$$= 2a(x_1\dot{x}_1 + x_2\dot{x}_2) - b \iint_{\mathcal{H}_2(x_3)} \rho(\tilde{x}_1, \tilde{x}_2) (2x_1\dot{x}_1 + 2x_2\dot{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.19)$$

$$= 2a(x_1\dot{x}_1 + x_2\dot{x}_2) - 2b(x_1\dot{x}_1 + x_2\dot{x}_2) \iint_{\mathcal{H}_2(x_3)} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.20)$$

$$\dot{V} = (x_1\dot{x}_1 + x_2\dot{x}_2) (2a - 2b) \iint_{\mathcal{H}_2(x_3)} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.21)$$

Note that $\iint_{\mathcal{H}_2(x_3)} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2$ is just some constant if the obstacles aren't moving and could be continuously normalized to some constant if they are moving. Let this

constant be defined as:

$$\mathcal{P} \stackrel{def}{=} \iint_{\mathcal{H}_2(x_3)} \rho(\tilde{x}_1, \tilde{x}_2) d\tilde{x}_1 d\tilde{x}_2 \quad (6.22)$$

Also note that $\rho(\tilde{x}_1, \tilde{x}_2) \in \{0, \frac{1}{area}\} \forall (\tilde{x}_1, \tilde{x}_2) \rightarrow \rho \geq 0 \therefore \mathcal{P} > 0$ if there are any obstacles and $\mathcal{P} = 0$ if there are none.

$$\implies \dot{V} = (x_1\dot{x}_1 + x_2\dot{x}_2) (2a - 2b) (2a - 2b\mathcal{P}) \quad (6.23)$$

$$= \{-\alpha x_1^2 + \beta x_2 x_3 \text{sign } x_3 - \alpha x_2^2 - \beta x_1 x_2 \text{sign } x_3\} (2a - 2b\mathcal{P}) \quad (6.24)$$

$$\dot{V} = -2\alpha (x_1^2 + x_2^2) (2a - 2b\mathcal{P}) \quad (6.25)$$

$$\therefore a \text{ and } b \text{ can be chosen such that } \dot{V} < 0 \quad (6.26)$$

Again, as with the Bloch-Drakunov controller, the Lyapunov function was only on the plane of constant x_3 , $\mathcal{H}_2(x_3)$. Also recall that, with the Heisenberg system,

$$\dot{x}_1 = u_1, \quad (6.27)$$

$$\dot{x}_2 = u_2, \quad (6.28)$$

$$\dot{x}_3 = x_1 u_2 - x_2 u_1, \quad (6.29)$$

x_3 must be driven to zero before x_1 and x_2 . because when $x_1 = x_2 = 0$, x_3 is not controllable, i.e. $\dot{x}_3 = 0$.

As we did with the Bloch-Drakunov controller we will differentiate x_3 .

$$\dot{x}_3 = x_1 u_2 - x_2 u_1 \quad (6.30)$$

$$= -\alpha x_1 x_2 + \beta x_1^2 \text{sign}(x_3) + \alpha x_1 x_2 + \beta x_2^2 \text{sign}(x_3) \quad (6.31)$$

$$= -2\beta(x_1^2 + x_2^2) \text{sign}(x_3) \quad (6.32)$$

Integrating both sides of the equation

$$x_3(t) - x_3(0) = -2\beta \int_0^t (x_1^2 + x_2^2) d\tau \operatorname{sign}(x_3) \quad (6.33)$$

$$x_3(t) = \underbrace{x_3(0)}_a - \underbrace{2\beta \operatorname{sign}(x_3) \int_0^t (x_1^2 + x_2^2) d\tau}_b \quad (6.34)$$

Here everything is the same as for the Bloch-Drakunov controller described in section 3.2.2. Part a and part b of the above equation, are of opposite signs. If the $x_3(0)$ is negative, part (b) will approach $|x_3(0)|$ as $\tau \rightarrow t$. Note that $\operatorname{sign}(x_3) = \operatorname{sign}(x_3(0))$ because x_3 will not cross the $x_3 = 0$ plane before time, t . If $x_3(0)$ is positive, part (b) will be initially negative, and will equal $-x_3(0)$ at time, t . From this we have the condition that

$$2\beta \int_0^\infty (x_1^2 + x_2^2) d\tau \geq |x_3(0)|. \quad (6.35)$$

To integrate this, in the case of the simple Lyapunov function described earlier, $V = \frac{1}{2}(x_1^2 + x_2^2)$ we were able to take advantage of the fact that $V(t) = V(0)e^{-2\alpha\tau}$, but for the new Lyapunov function, (equation, 6.17), we don't $V(t)$. We can try integrating anyway.

$$2\beta \int_0^\infty (x_1^2 + x_2^2) d\tau \geq |x_3(0)| \quad (6.36)$$

We can't integrate this, so unfortunately we don't get the stability condition in a useful form. We will have to try something else

6.6 Combined Potential Fields and Sliding Modes

A method which eliminates the local minima problem and shortens the paths produced by the potential field method can be found in the literature Hashimoto et al. [1992]. In this paper, "Obstacle Avoidance Control in Multi-Dimensional Space Using Sliding Mode" Hashimoto et al. [1992] potential field path planning was combined

with a sliding mode approach for the controls. The potential field was set up as follows. A positive charge density was applied to all obstacles in the configuration space CS , (labeled as R^n in the following equations). Then a negative point charge c^* is applied at the goal point q^* in the n -dimensional configuration space

(Note that for a robot moving on a two-dimensional plane with configuration space $CS \in R^3$ where q_1 and q_2 are the x and y directions and q_3 is orientation, The charge densities of the obstacles are constant $\forall q_3$ but the goal is a single point in R^3).

This whole approach is almost exactly the same as the one used in this thesis with the following differences: 1) The model used in this paper was dynamic, whereas the model used in this thesis is only kinematic. 2) The model in this thesis is nonholonomic, so the sliding surfaces use a variation of the Heisenberg system, whereas the sliding surfaces in this paper were purely the force curves on the Laplace electrostatic potential field.

$$U(q) = \int_{R^n} \frac{c(\xi)}{(\sum_{i=1}^n (q_i - \xi_i)^2)^{\frac{1}{2}}} d\xi - \frac{c^*}{(\sum_{i=1}^n (q_i^* - q_i)^2)^{\frac{1}{2}}} \quad (6.37)$$

$$\int_{R^n} c(\xi) d\xi < c^* \quad (6.38)$$

Hashimoto et al. [1992]

6.7 Combined Potential Fields and Sliding Modes (A New Approach)

The potential field could be obtained using the electrostatic potential given by(6.39):

$$\psi(x_1, x_2, x_3) = \int_{R^3} \frac{c(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3)}{[\sum_{i=1}^3 (x_i - \tilde{x}_i)^2]^{\frac{1}{2}}} d\tilde{x}_1 d\tilde{x}_2 d\tilde{x}_3 - \frac{c^*}{[\sum_{i=1}^3 (z_i^* - x_i)^2]^{\frac{1}{2}}}, \quad (6.39)$$

where $c^* > 0$ and:

- $X = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3)$ an arbitrary point,
- $q^* = (z_1, z_2, z_3)$ goal point location,
- $\psi(x_1, x_2, x_3)$ potential field at point (x_1, x_2, x_3) ,
- c^* charge at goal point,
- $c(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3)$ is the density of charges at the obstacles.

To assure that the gradient of the potential field will point towards the goal point, the sum of the positive charges must be significantly less than the magnitude of the negative charge at the goal point. i.e.

$$\int_{R^3} c(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3) d\tilde{x}_1 d\tilde{x}_2 d\tilde{x}_3 < c^* \quad (6.40)$$

The desired trajectory (plan) for this form of potential-field directional control is given by the following equations:

$$\dot{x}_1^* = -\frac{\partial \psi}{\partial x_1}(x_1^*, x_2^*, x_3^*), \quad (6.41)$$

$$\dot{x}_2^* = -\frac{\partial \psi}{\partial x_2}(x_1^*, x_2^*, x_3^*), \quad (6.42)$$

$$\dot{x}_3^* = -\frac{\partial \psi}{\partial x_3}(x_1^*, x_2^*, x_3^*) \quad (6.43)$$

Lastly, the tracking algorithm described in section (3.3.2) is used to track the point $x^* = [x_1^*, x_2^*, x_3^*]^T$.

This algorithm was modeled in Matlab / Simulink. The model is given in the next section.

Chapter 7

Conclusions

The goal of this thesis was to review the nonholonomic systems theory and to design a stabilizing feedback control algorithm for nonholonomic systems in the situation when certain areas of the state space should be avoided during the system stabilization. In application to autonomous vehicles this problem can be interpreted as obstacles avoidance. After reviewing the literature on existing methods, the variable structure/sliding mode control was used as the mathematical tool for designing nonlinear feedback. Such control approach is known for outstanding robustness properties. It results in closed loop systems that can operate under extreme uncertainty and in the presence of strong disturbances. As a method for the obstacle avoidance the potential field approach was chosen. In contrast with the conventional use of this approach in robotic systems where the artificial potential field of forces is introduced in the system's physical space, it was considered in the state space of the canonical form of the nonholonomic system. Using Brockett's theorem we considered a generalized Heisenberg system as a canonical form and introduced the artificial potential field in the Heisenberg space. Several examples from literature were also worked out to provide insight into possible solutions.

Chapter 8

Matlab Code

The code for the various simulations used in this thesis is given here.

8.1 The Main File

There was one main file, which allowed the various functions to be commented or uncommented, depending on the task.

```
1 % First run the simulink file,
2 % (BD_HSMC.mdl or other controller that will produce the inputs)
3
4 function StevesThesis(x,y,phi,alpha,beta,L)
5 pix = 128; % # of pixels per demension pick a power of 2 if posible
6 DPS =[L L pi/3]; % (meters & rad) Dimensions of Physical Space
7 pic = imread('BlocksAndCircle.bmp');
8 % pic = imread('BandWlaserView1.bmp');
9 imshow(pic)
10 %% Setting up the constraints on the x-y plane in physical space
11 [I,BW,BW2]=constraints(L,pic);
12
13 %% Adding goal point to charge map and setting up weighting
14 ChargeMap=ObstacleandGoalCharges(BW2,L);
15
```

[illegible]

```
49
50 %% Plotting paraboloid in Heisenberg space
51 %%figure(1)
52 % hold on
53 % [j_length,i_length]=size(parabaloid)
54 % i=[1:ncols];
55 % j=[1:mrows];
56 %%scatter3(parabaloidx1,parabaloidx2,parabaloidx3,1)
57 %%axis([-1,1,-40,40,-100,100])
58 % view([-20,-15,20])
59 % mesh(i_length,j_length,parabaloid)
60
61 %% 4-Subplots showing image processing
62 % figure(2)
63 % subplot(2,2,1), imshow(pic)
64 % title('Simulated sensor data')
65 % subplot(2,2,2), imshow(I)
66 % title('Resizing pixel matrix leaves edges blurry.')
67 % subplot(2,2,3), imshow(BW)
68 % title('Edges sharpened and values reversed')
69 % subplot(2,2,4), imshow(BW2)
70 % title('Boundaries found for application of potentials')
71 % size(PField)
72
73 %% Plotting the potential field
74 figure(3)
75 [ncols, mrows]=size(PField);
76 i=[1:ncols];
77 j=[1:mrows];
78 mesh(i,j,PField(i,j))
79 title('Potential Field of Boundaries, Negative Potential for Goal')
80 axis([1,ncols,1,mrows,-300,300])
81 view([-20,-15,20])
```

```
82 clear i;
83 clear j;
84
85 %% Contour plot of potential field
86 % figure(4)
87 % [ncols, mrows]=size(PField);
88 % i=[1:ncols];
89 % j=[1:mrows];
90 % contour(i,j,PField(i,j),60)
91 % title('Contour')
92 % clear i;
93 % clear j;
94
95 %% Showing Gradient of Potential field
96 [spx11,spx12]=size(px);
97 [spy11,spy12]=size(py);
98
99 figure(5) % one arrow for every data point
100 i=[1:spx12];
101 j=[1:spy11];
102 quiver(i,j,-px2,-py2,5);
103 title({'Gradient of P.field','provides steering direction.'})
104 axis([1,spx12,1,spy11])
105 clear i;
106 clear j;
107
108 figure(6) % This has more spaced out arrows
109 arrowscale=7;
110 qtf=3; % Here only the multiples of the qtf'th elements are used
111 % for arrows this is because if an arrow is plotted for every
112 % value, the plot appears too hard to read. The gradient data is
113 % still retained for every point
114 i=[1:qtf:spx12];
```

```

115 j=[1:qtf:spy11];
116 px2=px(1:qtf:spx11,1:qtf:spx12);
117 py2=py(1:qtf:spy11,1:qtf:spy12);
118 quiver(i,j,-px2,-py2,arrowscale,'LineWidth',1);
119 title({'Gradient of P.field','provides steering direction.'})
120 axis([1,spx12,1,spy11])
121 clear i;
122 clear j;
123
124 %% The image processing as separate figures
125 % figure(7)
126 % imshow(pic)
127 % title({'Simulated Sensor Image','(Assumed Sensory Output)'})
128 %
129 % figure(7)
130 % imshow(I)
131 % title('Resizing pixel matrix leaves edges blurry.')
132 %
133 % figure(8)
134 % imshow(BW)
135 % title('Edges sharpened and values reversed')
136 %
137 % figure(9)
138 % imshow(BW2)
139 % title({'Boundaries found for','application of potentials'})

```

8.2 Animating the Car to Demonstrate Simulink Models

This code shows the a small rectangle following the path created by the selected controller.

```

1 function animatedcar(x,y,phi,DPS,BW2)

```

```
2
3  xmax = (DPS(1)-1)/2;
4  xmin = -1*xmax;
5  ymax = (DPS(2)-1)/2;
6  ymin = -1*ymax;
7  cl=3; %carlength
8  cw=2; %carwidth
9  s=length(x);
10 flx=zeros(1,s);
11 frx=zeros(1,s);
12 brx=zeros(1,s);
13 blx=zeros(1,s);
14 fly=zeros(1,s);
15 fry=zeros(1,s);
16 bry=zeros(1,s);
17 bly=zeros(1,s);
18 cp=zeros(1,s);
19 sp=zeros(1,s);
20 hd=[];
21 cp=cos(phi);
22 sp=sin(phi);
23 flx=x+cl.*cp-0.5*cw.*sp;
24 frx=x+cl.*cp+0.5*cw.*sp;
25 brx=x+0.5*cw.*sp;
26 blx=x-0.5*cw.*sp;
27 fly=y+cl.*sp+0.5*cw.*cp;
28 fry=y+cl.*sp-0.5*cw.*cp;
29 bry=y-0.5*cw.*cp;
30 bly=y+0.5*cw.*cp;
31
32 figure %Draw the figure to be animated
33 % imshow(BW2)
34 hold on;
```

```

35 i=1;
36 for i=1:s           % Animation loop
37     if (i-1)/10 == floor((i-1)/10) %plots 1 point /50 calculations
38         scatter(x(i),y(i),2,'k')
39     end
40     if (i-1)/600 == floor((i-1)/600); %Plots 1 pt./150 calculations
41     xd = [flx(i),frx(i),brx(i),blx(i),flx(i)]; %Corners of a square
42     yd = [fly(i),fry(i),bry(i),bly(i),fly(i)];
43     hd = fill(xd,yd,'r'); % Draw the square and save handle
44     set(hd,'FaceColor','none');
45     set(hd,'Xdata',xd,'Ydata',yd);
46     axis([xmin,xmax,ymin,ymax]);
47     pause(0.0005);
48 end
49 end
50 hold off

```

8.3 Image Processing and Potential Field Functions

8.3.1 Image Processing to Find Obstacles

This m-file reads the image, scales it, and then sharpens the edges, and finds the borders.

```

1 %% Putting constraints on x-y plane in physical space
2 function [I,BW,BW2]=constraints(L,pic)
3 % Is bitmap out of 32 or 256?
4 ColorDepth=max(max(pic));
5     %[m,n] = size(pic);
6     %margin = ((m*n)/(pix*pix))^.5/5;
7 margin = ColorDepth/128;
8 I = imresize(pic, [2*L+1 2*L+1]);

```

```

9  BW = zeros(2*L+1,2*L+1);
10 i=0;j=0;k=0;
11 for i=1:2*L
12     for j=1:2*L
13         if I(i,j)>= (ColorDepth-margin)
14             BW(i,j) = 0;
15         else %the way this is set up, this inverses the image
16             BW(i,j) = 1;
17         end
18     end
19 end
20
21 BW2 = ones(2*L+1,2*L+1);
22 surroundings = 1;
23 for i=2:2*L
24     for j=2:2*L
25         surroundings = BW(i,j)+BW(i-1,j)+BW(i-1,j-1)+BW(i,j-1)+...
26             BW(i+1,j)+BW(i+1,j+1)+BW(i,j+1)+BW(i-1,j+1)+BW(i+1,j-1);
27         if (surroundings >= 6) && (surroundings <= 8)
28             BW2(i,j) = 1;
29         else
30             BW2(i,j) = 0;
31         end
32     end
33 end

```

8.3.2 Placing Charges on Obstacles and Goal Position

This code, assigns a charge density to the borders of the obstacles and of the configuration space. It also assigns a point charge at the origin.

```

1  %% Adding goal point to charge map and setting up weighting
2  function ChargeMap=ObstacleandGoalCharges(BW2,L)

```



```
3 xp=26;
4 yp=-24;
5 xi = xp*L/size(BW2,1)+L;
6 yi = yp*L/size(BW2,2)+L;
7 XYGoal=round([xi,yi]);
8 Kg=275; %(unitless) Charge weighting of goal node
9 Ko=8;    %(unitless) Charge weighting of boundary nodes
10 Goalpoint = zeros(size(BW2,1),size(BW2,2));
11 Goalpoint(XYGoal(1),XYGoal(2))=-1*Kg;
12 ChargeMap=Goalpoint+(Ko*BW2);
```

8.3.3 Creating the Potential Field

Here the potential field is created in two dimensions.

```
1 function PField = PotentialField(ChargeMap)
2 %% Building the Potential Field
3 sizeCM=size(ChargeMap);
4 PField      = zeros(sizeCM);
5 a           = PField;
6 b           = a;
7 for Y = 1:sizeCM(2),
8 for X = 1:sizeCM(1),
9 for y = 1:sizeCM(2),
10 for x = 1:sizeCM(1),
11 a(y,x)=(((y-Y)^2)+((x-X)^2)).5;
12 end
13 end
14 a(Y,X)=1;
15 b=ChargeMap./a;
16 PField(Y,X)=sum(sum(b));
17 end
18 end
```

```
19 PField=rot90(PField',1);
```

8.4 Transformation to Heisenberg Space

In this code, the obstacles are transformed into Heisenberg space.

8.4.1 Putting Obstacles in Heisenberg Space

```
1
2 function [plotx1,plotx2,plotx3]=TransformedObstacles(BW2,DPS)
3 %% Putting Obstacles in Heisenberg Space
4 xmax = (DPS(1)-1)/2;
5 xmin =-1*xmax;
6 ymax = (DPS(2)-1)/2;
7 ymin =-1*ymax;
8 thetamax = DPS(3)/2;
9 thetamin =-1*thetamax;
10 phimax = 2*pi;
11 phimin = 0;
12 sizeBW2=size(BW2);
13 PixelsPerSideOfCS=sizeBW2(1);
14 dx=(xmax-xmin)/(sizeBW2(1)-1);
15 dy=(ymax-ymin)/(sizeBW2(2)-1);
16 dphi=(phimax-phimin)/(PixelsPerSideOfCS-1);
17
18 X = ( xmin : dx : xmax );
19 Y = ( ymin : dy : ymax );
20 Phi = (phimin:dphi:phimax);
21
22 i=0;
23 j=0;
24 k=0;
```

```
25 x=0;
26 y=0;
27 phi=0;
28 co=0;
29 cp=0;
30 cs=0;
31 i1=1;
32 i2=2;
33 c1=0;
34
35 for i1=2:PixelsPerSideOfCS-1,
36     c1=c1+sum(BW2(i1,:));
37 end
38 c3=PixelsPerSideOfCS*PixelsPerSideOfCS;
39 c4=c1/c3;
40
41 Xtilde=X./DPS(1);
42 Ytilde=Y./DPS(2);
43 z=((PixelsPerSideOfCS/2)^3)/2;
44 plotx1=zeros(1,c1);
45 plotx2=zeros(1,c1);
46 plotx3=zeros(1,c1);
47 c2=1;
48 for k=2:1:PixelsPerSideOfCS-1,
49     phi=Phi(k);
50     cp = cos(phi);
51     sp = sin(phi);
52     for j=2:1:PixelsPerSideOfCS-1,
53         y =Ytilde(j);
54         for i=2:1:PixelsPerSideOfCS-1,
55             x =Xtilde(i);
56             if BW2(i,j)==1,
57                 x1 = phi;
```

```

58         x2 =          x*cp + y*sp          ;
59         x3 = x*phi*cp - 2*x*sp + y*phi*sp + 2*y*cp ;
60         plotx1(c2)=x1;
61         plotx2(c2)=x2;
62         plotx3(c2)=x3;
63         c2=c2+1;
64         else
65             c2=c2;
66         end
67
68     end
69 end
70 end
71 i=1;
72 j=1;
73 k=0;
74 x=0;
75 y=0;
76 phi=0;

```

8.4.2 Adding in a Sample Paraboloid

Finally a sample paraboloid is added to the Heisenberg space to demonstrate the intersections of the obstacles and the sliding manifold.

```

1  %% Adding in a sample Paraboloid
2  function [parabalx1,parabalx2,parabalx3]=SManifold(pix,alpha,beta)
3  %paraboloid = zeros(pix,pix);
4  parabalx1 = zeros(1,pix);
5  parabalx2 = zeros(1,pix);
6  parabalx3 = zeros(1,pix);
7  i=1;
8  for i1 = -2:4/pix:2-4/pix,

```

```
9  for i2 = -2:4/pix:2-4/pix,
10  parabalx1(i)=i1;                % usefull for statater3
11  parabalx2(i)=i2;                % usefull for statater3
12  parabalx3(i)=(.5*beta/alpha)*(i1^2+i2^2); % usefull for statater3
13  % parabaloid(i2,i1)=(.5*beta/alpha)*(it^2+i2^2) % usefull for mesh
14  i=i+1;
15  end
16  end
17  i=0;
```

Bibliography

- E Al-Ragib. Master's thesis, King Fahd University of Petroleum and Minerals, DHahran.
- G. Artus, P. Morin, and C. Samson. Control of a maneuvering mobile robot by the transverse function approach: control design and simulation results, 2004.
- D.J. Balkcom and M.T. Mason. Time Optimal Trajectories for Bounded Velocity Differential Drive Vehicles. *International Journal of Robotics Research*, 21(3):199–217, 2002.
- A. M. Bloch, M. Reyhanoglu, and N. H. McClamroch. Control and Stabilization of Nonholonomic Dynamic Systems. *IEEE Transactions on Automatic Control*, 37(11):1746–1757, 1992.
- A.M. Bloch. *Nonholonomic mechanics and control*. Springer, 2003. ISBN 0387955356.
- A.M. Bloch and S.V. Drakunov. Stabilization and tracking in the nonholonomic integrator via sliding modes 1. *Systems & Control Letters*, 29(2):91–99, October 1996.
- A.M. Bloch, S.V. Drakunov, and Kinyon M.K. Stabilization of Brockett's generalized canonical driftless system. In *Proceedings of the 36th IEEE Conference on Decision and Control*, pages 4260–4265, San Diego, CA, December 1997.
- R. W. Brockett. Control theory and singular Riemannian geometry. In *New Directions in Applied Mathematics*, pages 11–27. Springer-Verlag, 1981.
- R. W. Brockett. Asymptotic stability and feedback stabilization. *Differential Geometric Control Theory*, pages 181–191, 1983.

- C. Canudas de Wit and O.J. Sordalen. Exponential Stabilization of Mobile Robots with Nonholonomic Constraints. *IEEE Transactions on Automatic Control*, 37(11):1791–1797, 1992.
- R. A. DeCarlo, S. H. Zak, and G. P. Matthews. Variable structure control of nonlinear multivariable systems: a tutorial. *Proceedings of the IEEE*, 76(3):212–232, March 1988. ISSN 00189219.
- S. V. Drakunov, T. Floquet, and Perruquetti W. Stabilization and tracking for an extended Heisenberg system with a drift. *Systems & Control Letters*, 54(5):435–445, May 2005.
- L.E. Dubins. On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79(3):497–516, July 1957.
- O.E. Fernandez, A.M. Bloch, and T. Mestdag. The Pontryagin maximum principle applied to nonholonomic mechanics. *2008 47th IEEE Conference on Decision and Control*, (1): 4306–4311, 2008.
- G.I. Frobenius. ber das Pfaff’sche probleme. *Journal fr Math.*, (82):230–315, 1877.
- A.A. Furtuna, D.J. Balkcom, H. Chitsaz, and P. Kavathekar. Generalizing the dubins and reeds-shepp cars: Fastest paths for bounded-velocity mobile robots. In *2008 IEEE International Conference on Robotics and Automation*, pages 2533–2539. IEEE, May 2008. ISBN 978-1-4244-1646-2.
- J. Guldner and V.I. Utkin. Stabilization of non-holonomic mobile robots using Lyapunov functions for navigation and sliding mode control. *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, (December):2967–2972, 1994.
- H. Hashimoto, Y. Kunii, F. Harashha, V.I. Utkin, and S.V. Drakunov. Obstacle Avoidance Control in Multi-Dimentional Space Using Sliding Mode. In *Proceedings of IEEE/RSJ International Conference On Intelligent Robots And Systems*, number 3, pages 697–702, Raleigh, NC, 1992. IEEE/RSJ.
- J Hespanha. Stabilization of nonholonomic integrators via logic-based switching. *Automatica*, 35(3):385–393, March 1999. ISSN 00051098.

- Y.K. Hwang and N. Ahuja. A Potential Field Approach to Path Planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32, 1992.
- J.P. Jiang and H. Nijmeijer. Tracking Control of Mobile Robots: A Case Study in Backstepping. *Automatica*, 33(7):1393–1399, 1997.
- J.P. Jiang and H. Nijmeijer. A Recursive Technique for Tracking Control of Nonholonomic Systems in Chained Form. *IEEE Transactions on Automatic Control*, 44(2):265–279, 1999.
- J.P. Jiang, A.A.J. Lefeber, and H. Nijmeijer. Saturated Stabilization and Tracking of a Nonholonomic Robot. *Systems & Control Letters*, 42:327–332, 2001.
- P. Khosla and R. Volpe. Superquadric artificial potentials for obstacle avoidance and approach. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1778–1784, Philadelphia, PA, 1988. IEEE Comput. Soc. Press.
- B. Kim and P. Tsiotras. Controllers for Unicycle-Type Wheeled Robots: Theoretical Results and Experimental Validation. *Transactions on Robotics and Automation*, 18(3):294–299, June 2002.
- K. Kondak and G. Hommel. Computation of Time Optimal Movements for Autonomous Parking of Non-Holonomic Mobile Platforms. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 2698–2703, Seoul, Korea, 2001. IEEE.
- J.P. Laumond. Nonholonomic Motion Planning versus Controllability via the Multibody Car System Example. Technical report, Department of Computer Science, Stanford University, October 1990.
- J.P. Laumond. *Robot motion planning and control*. 1998. ISBN 9783540762195.
- S.M. LaValle. *Planning algorithms*. Cambridge University Press, 2006. ISBN 0521862051.
- W.S. Levine. *The Control Handbook*. CRC Press, 1996. ISBN 0849385709.
- G.A.D. Lopes. *Perception Based Navigation for Underactuated Robots*. Ph.d, University of Michigan, 2007.

- P. Morin and C. Samson. Practical stabilization of driftless homogeneous systems based on the use of transverse periodic functions. *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, pages 1761–1766, 2001.
- P. Morin and C. Samson. Control of Nonholonomic Mobile Robots Based on the Transverse Function Approach. *IEEE Transactions on Robotics*, 25(5):1058–1073, October 2009. ISSN 1552-3098.
- R.M. Murray and S.S. Sastry. Nonholonomic motion planning: steering using sinusoids. *IEEE Transactions on Automatic Control*, 38(5):700–716, May 1993. ISSN 00189286.
- Y. Nakamura and H. Ezaki. Design of steering mechanism and control of nonholonomic trailer systems. *IEEE Transactions on Robotics and Automation*, 17(3):367–374, June 2001. ISSN 1042296X.
- C. Park, D.J. Scheeres, V. Guibout, and A. Bloch. Globally Optimal Feedback Control Law of the Underactuated Heisenberg System by Generating Functions. In *Proceedings of 45th IEEE Conference on Decision and Control*, pages 2687–2692, San Diego, CA, 2006. IEEE.
- I.E. Paromtchik and C. Laugier. Autonomous parallel parking of a nonholonomic vehicle. *Proceedings of Conference on Intelligent Vehicles*, pages 13–18.
- B.H. Partee, A.G.B. ter Meulen, and R.E. Wall. *Mathematical Methods in Linguistics*. Kluwer Academic Publishers, Netherlands, hardbound edition, 1990.
- K. Pathak and S.K. Agrawal. An Integrated Path-Planning and Control Approach for Nonholonomic Unicycles Using Switched Local Potentials. *Transactions on Robotics*, 21(6):1201–1208, December 2005.
- V.G. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, and E.F. Mishenko. *The Mathematical Theory of Optimal Processes*. Interscience Publishers, 1962.
- J.A. Reeds and L.A. Shepp. Optimal Paths for a Car that Goes Both Forwards and Backwards. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.

- M. Reyhanoglu and E. Al-Regib. Nonholonomic Motion Planning for Wheeled Mobile Systems Using Geometric Phase. In *Proceedings of IEEE International Symposium on Intelligent Control*, pages 135–140, Columbus, OH, 1994. IEEE.
- M. Reyhanoglu and T. Geluk. Switched Feedback Tracking Control of a Nonholonomic Mobile Robot. In *Proceedings IEEE Industrial Electronics Society*, pages 3810–3814. IEEE, 2006.
- M. Sampei. *A control strategy for a class of nonholonomic systems - time-state control form and its application*. IEEE. ISBN 0-7803-1968-0.
- C. Samson. Control of Chained Systems Application to Path Following and Time-Varying Point-Stabilization of Mobile Robots. *IEEE Transactions on Automatic Control*, 40(1): 64–77, 1995.
- S. Sekhavat, P. Svestka, J.P. Laumond, and M. H. Overmars. Multilevel Path Planning for Nonholonomic Robots Using Semiholonomic Subsystems. *The International Journal of Robotics Research*, 17(8):840–857, August 1998. ISSN 0278-3649.
- O.J. Sordalen. *Feedback Control of Nonholonomic Mobile Robots*. Dr. ing., Norwegian Institute of Technology, 1993.
- O.J. Sordalen and K.Y. Wichlund. Exponential stabilization of a car with n trailers. In *Proceedings of 32nd IEEE Conference on Decision and Control*, volume 2, pages 978–983. IEEE, 1993. ISBN 0-7803-1298-8.
- M. Srinivasan, R.A. Metoyer, and E.N. Mortensen. Controllable real-time locomotion using mobility maps. *Proceedings of Graphics Interface*, 112:51–59, 2005.
- R. Weinstock. *Calculus of Variations with Applications to Physics and Engineering*. Dover, 1974. ISBN 0486630692.