A New Model-Free Sliding Mode Control Method with Estimation of Control Input Error

Raul Mittmann Reis
rmr2448@rit.edu

Follow this and additional works at: https://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
A New Model-Free Sliding Mode Control Method with Estimation of Control Input Error

by

Raul Mittmann Reis

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Electrical Engineering

Supervised by

Associate Professor Dr. Agamemnon Crassidis
Department of Electrical and Microelectronic Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, NY
May 2016

Approved By:

Dr. Agamemnon Crassidis, Associate Professor
Thesis Advisor, Department of Mechanical Engineering

Dr. Jason Kolodziej, Associate Professor
Committee Member, Department of Mechanical Engineering

Dr. Sergey Lyshevski, Professor
Committee Member, Department of Electrical and Microelectronic Engineering

Dr. Sohail Dianat, Professor
Department Head, Department of Electrical and Microelectronic Engineering
Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Dr. Agamemnon Crassidis for the continuous support of my study, patience, motivation and immense knowledge.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Sergey Lyshevski, Dr. Jason Kolodziej and Dr. Sohail Dianat, for their insightful comments and encouragement.

Most importantly, none of this would have been possible without the support and patience of my family: my parents Jair José Werle Reis and Angelita Beatriz Mittmann Reis. Even with all the distance between me and my family, I never felt far from home.

Finally, I appreciate the financial support from CAPES Foundation, Ministry of Education of Brazil, under grant 88888.075780/2013-00, and the support provided by IIE.
Abstract

A new type of sliding mode controller scheme, which requires no knowledge of system model, is derived in this work. The controller is solely based on previous control inputs and state measurements to generate the updated control input effort. The only knowledge required to derive the controller is the system order and the bounds of the control input gain, if one exists. The switching gain, which is required to drive the system states onto the sliding surface in the presence of disturbances and uncertainties, is derived using Lyapunov’s stability theorem, ensuring closed-loop asymptotic stability. The chattering effect, which is excited by the switching gain due to high activity of the control input, is reduced by using a smoothing boundary layer into the control law form. Simulations are performed, using first and second-order, linear and nonlinear systems, to test the performance of the new control law. In the last part of this work, the problem with state measurement noise is addressed. Results of the simulations validates the feasibility of the proposed control scheme.
Table of Contents

List of Figures ........................................................................................................ vi
List of Tables ........................................................................................................... x
Nomenclature .......................................................................................................... xi

1 Introduction ............................................................................................................ 1
  1.1. Motivation ........................................................................................................ 1
  1.2. Background Research on SMC ...................................................................... 1
  1.2.1 General SMC Schemes ................................................................................ 1
  1.2.2 Model-Free SMC Schemes .......................................................................... 4
  1.3. Research goals ................................................................................................ 5

2 Fundamentals of Lyapunov Theory ..................................................................... 7
  2.1. Nonlinear and Linear Systems ........................................................................ 7
  2.1.1 Companion Form .......................................................................................... 8
  2.2. Autonomous Systems ..................................................................................... 9
  2.3. Equilibrium Points .......................................................................................... 10
  2.3.1 Illustrative Example ..................................................................................... 10
  2.4. Concepts of Stability ....................................................................................... 11
  2.5. Positive Definite Functions ............................................................................ 13
  2.6. Lyapunov’s Direct Method .............................................................................. 14
  2.6.1 Illustrative Example: ................................................................................... 17

3 The Sliding Mode Control Method ....................................................................... 18
  3.1. Derivation ........................................................................................................ 18
  3.2. Boundary Layer .............................................................................................. 21
  3.3. Illustrative Example: ....................................................................................... 22
  3.3.1 Simulation ..................................................................................................... 25
  3.3.2 Inclusion of the boundary layer .................................................................. 28

4 Model-Free SMC with unitary control input gain .............................................. 32
  4.1. System Description ......................................................................................... 32
  4.2. First-Order System ........................................................................................ 33
  4.2.1 Proof of the Controller Form ....................................................................... 34
  4.2.2 Switching Gain ............................................................................................ 35
  4.2.3 Boundary Layer ........................................................................................... 36
  4.2.4 Illustrative Examples .................................................................................... 36
  4.2.4.1 First-order linear system ......................................................................... 37
  4.2.4.2 First-order nonlinear system ................................................................... 44
  4.3. Second-Order System .................................................................................... 50
  4.3.1 Proof of the Controller Form ....................................................................... 51
  4.3.2 Switching Gain ............................................................................................ 52
  4.3.3 Boundary Layer ........................................................................................... 53
  4.3.4 Illustrative Examples .................................................................................... 54
  4.3.4.1 Second-order linear system ..................................................................... 54
  4.3.4.2 Second-order nonlinear system ................................................................. 58

5 Model-Free SMC with non-unitary Control Input Gain ...................................... 64
  5.1. System Description ......................................................................................... 64
  5.2. First-Order System ........................................................................................ 65
  5.2.1 Proof of the Controller Form ....................................................................... 66
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Example of a pendulum</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>System trajectories with different systems</td>
<td>12</td>
</tr>
<tr>
<td>2.3</td>
<td>Lyapunov surfaces</td>
<td>15</td>
</tr>
<tr>
<td>2.4</td>
<td>Open Lyapunov surfaces</td>
<td>16</td>
</tr>
<tr>
<td>3.1</td>
<td>Sliding Condition</td>
<td>20</td>
</tr>
<tr>
<td>3.2</td>
<td>Variation of system parameters</td>
<td>26</td>
</tr>
<tr>
<td>3.3</td>
<td>Position comparison</td>
<td>26</td>
</tr>
<tr>
<td>3.4</td>
<td>Position tracking error</td>
<td>26</td>
</tr>
<tr>
<td>3.5</td>
<td>Velocity comparison</td>
<td>27</td>
</tr>
<tr>
<td>3.6</td>
<td>Velocity tracking error</td>
<td>27</td>
</tr>
<tr>
<td>3.7</td>
<td>Acceleration comparison</td>
<td>27</td>
</tr>
<tr>
<td>3.8</td>
<td>Acceleration tracking error</td>
<td>27</td>
</tr>
<tr>
<td>3.9</td>
<td>Control effort</td>
<td>28</td>
</tr>
<tr>
<td>3.10</td>
<td>Sliding condition</td>
<td>28</td>
</tr>
<tr>
<td>3.11</td>
<td>Position comparison</td>
<td>29</td>
</tr>
<tr>
<td>3.12</td>
<td>Position tracking error</td>
<td>29</td>
</tr>
<tr>
<td>3.13</td>
<td>Velocity comparison</td>
<td>30</td>
</tr>
<tr>
<td>3.14</td>
<td>Velocity tracking error</td>
<td>30</td>
</tr>
<tr>
<td>3.15</td>
<td>Acceleration comparison</td>
<td>30</td>
</tr>
<tr>
<td>3.16</td>
<td>Acceleration tracking error</td>
<td>30</td>
</tr>
<tr>
<td>3.17</td>
<td>Control effort</td>
<td>31</td>
</tr>
<tr>
<td>3.18</td>
<td>Boundary layer</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>Open-Loop system of the model described at Eq. (4.31)</td>
<td>37</td>
</tr>
<tr>
<td>4.2</td>
<td>Switching gain described at Eq. (4.26)</td>
<td>37</td>
</tr>
<tr>
<td>4.3</td>
<td>Control law described at Eq. (4.27)</td>
<td>38</td>
</tr>
<tr>
<td>4.4</td>
<td>Desired Tracking block</td>
<td>38</td>
</tr>
<tr>
<td>4.5</td>
<td>Position comparison for the linear first-order system example with relay function</td>
<td>38</td>
</tr>
<tr>
<td>4.6</td>
<td>Position tracking error for the linear first-order system example with relay function</td>
<td>38</td>
</tr>
<tr>
<td>4.7</td>
<td>Velocity comparison for the linear first-order system example with relay function</td>
<td>39</td>
</tr>
<tr>
<td>4.8</td>
<td>Velocity tracking error for the linear first-order system example with relay function</td>
<td>39</td>
</tr>
<tr>
<td>4.9</td>
<td>Sliding condition for the linear first-order system example with relay function</td>
<td>39</td>
</tr>
<tr>
<td>4.10</td>
<td>Control effort for the linear first-order system example with relay function</td>
<td>39</td>
</tr>
<tr>
<td>4.11</td>
<td>Open-Loop system of the model described at Eq. (4.31)</td>
<td>40</td>
</tr>
<tr>
<td>4.12</td>
<td>Switching gain described at Eq. (4.26)</td>
<td>41</td>
</tr>
<tr>
<td>4.13</td>
<td>Control law with described at Eq. (4.28)</td>
<td>41</td>
</tr>
<tr>
<td>4.14</td>
<td>Desired Tracking block</td>
<td>41</td>
</tr>
<tr>
<td>4.15</td>
<td>Sign or Sat. block</td>
<td>41</td>
</tr>
<tr>
<td>4.16</td>
<td>Boundary Layer block</td>
<td>42</td>
</tr>
<tr>
<td>4.17</td>
<td>Position comparison for the linear first-order system example with boundary layer</td>
<td>42</td>
</tr>
<tr>
<td>4.18</td>
<td>Position tracking error for the linear first-order system example with boundary layer</td>
<td>42</td>
</tr>
<tr>
<td>4.19</td>
<td>Velocity comparison for the linear first-order system example with boundary layer</td>
<td>43</td>
</tr>
<tr>
<td>4.20</td>
<td>Velocity tracking error for the linear first-order system example with boundary layer</td>
<td>43</td>
</tr>
<tr>
<td>4.21</td>
<td>Sliding condition for the linear first-order system example with boundary layer</td>
<td>43</td>
</tr>
<tr>
<td>4.22</td>
<td>Control effort for the linear first-order system example with boundary layer</td>
<td>43</td>
</tr>
<tr>
<td>4.23</td>
<td>Open-Loop system of the model described at Eq. (4.33)</td>
<td>44</td>
</tr>
<tr>
<td>4.24</td>
<td>Switching gain described at Eq. (4.26)</td>
<td>44</td>
</tr>
<tr>
<td>4.25</td>
<td>Control law described at Eq. (4.27)</td>
<td>45</td>
</tr>
<tr>
<td>4.26</td>
<td>Desired Tracking block</td>
<td>45</td>
</tr>
<tr>
<td>4.27</td>
<td>Position comparison for the nonlinear first-order system example with relay function</td>
<td>45</td>
</tr>
<tr>
<td>4.28</td>
<td>Position tracking error for the nonlinear first-order system example with relay function</td>
<td>45</td>
</tr>
<tr>
<td>4.29</td>
<td>Velocity comparison for the nonlinear first-order system example with relay function</td>
<td>46</td>
</tr>
<tr>
<td>4.30</td>
<td>Velocity tracking error for the nonlinear first-order system example with relay function</td>
<td>46</td>
</tr>
<tr>
<td>4.31</td>
<td>Sliding condition for the nonlinear first-order system example with relay function</td>
<td>46</td>
</tr>
</tbody>
</table>
Figure 4.32: Control effort for the nonlinear first-order system example with relay function ........................................46
Figure 4.33: Open-Loop system of the system described at Eq. (4.33) .................................................................47
Figure 4.34: Control law described at Eq. (4.28) .........................................................................................................47
Figure 4.35: Desired Tracking block .........................................................................................................................48
Figure 4.36: Switching gain described at Eq. (4.26) .................................................................................................48
Figure 4.37: Sign or Sat. block ..................................................................................................................................48
Figure 4.38: Boundary Layer block ..........................................................................................................................48
Figure 4.39: Position comparison for the nonlinear first-order system example with boundary layer ..........49
Figure 4.40: Position tracking error for the nonlinear first-order system example with boundary layer ......49
Figure 4.41: Velocity comparison for the nonlinear first-order system example with boundary layer ......49
Figure 4.42: Velocity tracking error for the nonlinear first-order system example with boundary layer .....49
Figure 4.43: Sliding condition for the nonlinear first-order system example with boundary layer ........50
Figure 4.44: Control effort for the nonlinear first-order system example with boundary layer ....................50
Figure 4.45: Open-Loop system of the system described at Eq. (4.60) ..............................................................55
Figure 4.46: Control law described at Eq. (4.57) .......................................................................................................55
Figure 4.47: Desired Tracking block .........................................................................................................................55
Figure 4.48: Switching gain described at Eq. (4.55) .................................................................................................55
Figure 4.49: Sign or Sat. block ..................................................................................................................................56
Figure 4.50: Boundary Layer block ..........................................................................................................................56
Figure 4.51: Position comparison for the linear second-order system example with boundary layer .......56
Figure 4.52: Position tracking error for the linear second-order system example with boundary layer ....56
Figure 4.53: Velocity comparison for the linear second-order system example with boundary layer .......57
Figure 4.54: Velocity tracking error for the linear second-order system example with boundary layer .......57
Figure 4.55: Acceleration comparison for the linear second-order system example with boundary layer ....57
Figure 4.56: Acceleration tracking error for the linear second-order system example with boundary layer .57
Figure 4.57: Sliding condition for the linear second-order system example with boundary layer ...........58
Figure 4.58: Control effort for the linear second-order system example with boundary layer ....................58
Figure 4.59: Open-Loop system of the system described at Eq. (4.62) ..............................................................59
Figure 4.60: Control law described at Eq. (4.57) .......................................................................................................59
Figure 4.61: Desired Tracking block .........................................................................................................................60
Figure 4.62: Switching gain described at Eq. (4.55) .................................................................................................60
Figure 4.63: Sign or Sat. block ..................................................................................................................................60
Figure 4.64: Boundary Layer block ..........................................................................................................................60
Figure 4.65: Position comparison for the nonlinear second-order system example with boundary layer .....61
Figure 4.66: Position tracking error for the nonlinear second-order system example with boundary layer ..61
Figure 4.67: Velocity comparison for the nonlinear second-order system example with boundary layer .....61
Figure 4.68: Velocity tracking error for the nonlinear second-order system example with boundary layer .61
Figure 4.69: Acceleration comparison for the nonlinear second-order system example with boundary layer .........................................................................................................................62
Figure 4.70: Acceleration tracking error for the nonlinear second-order system example with boundary layer ........................................................................................................................................62
Figure 4.71: Sliding condition for the nonlinear second-order system example with boundary layer ..........62
Figure 4.72: Control effort for the nonlinear second-order system example with boundary layer ..........62
Figure 5.1: Control input gain variation over time ....................................................................................................70
Figure 5.2: Open-Loop system of the model described at Eq. (5.35) .................................................................71
Figure 5.3: Control law described at Eq. (5.33) ..........................................................................................................71
Figure 5.4: Desired Tracking block ..........................................................................................................................71
Figure 5.5: Switching gain described at Eq. (5.30) .................................................................................................72
Figure 5.6: Sign or Sat. block ..................................................................................................................................72
Figure 5.7: Boundary Layer block ..........................................................................................................................72
Figure 5.8: Control input gain block ........................................................................................................................72
Figure 5.9: Position comparison for the first-order linear example with non-unitary control input gain ......73
Figure 5.10: Position tracking error for the first-order linear example with non-unitary control input gain .73
Figure 5.11: Velocity comparison for the first-order linear example with non-unitary control input gain ....73
Figure 5.12: Velocity tracking error for the first-order linear example with non-unitary control input gain 73
Figure 6.23: Control effort for the second non-
Figure 6.22: Sliding condition for the second
Figure 6.20: Acceleration comparison for the second
Figure 6.17: Position tracking error for the second
Figure 6.15: Control effort for the first
Figure 6.13: Velocity tracking error for the first
Figure 6.12: Velocity comparison for the first
Figure 6.11: Position tracking error for the first
Figure 6.10: Position comparison for the first
Figure 6.6: Acceleration comparison for the second
Figure 6.5: Velocity tracking error for the second
Figure 6.3: Position tracking error for the second
Figure 6.2: Position comparison for the second
Figure 6.1: Noise sample

Figure 5.58: Control effort for the second non-
Figure 5.57: Sliding condition for the second
Figure 5.55: Acceleration comparison for the second
Figure 5.54: Sliding condition for the second
Figure 5.53: Acceleration comparison for the second
Figure 5.52: Sliding condition for the second
Figure 5.51: Acceleration comparison for the second
Figure 5.50: Sliding condition for the second
Figure 5.49: Acceleration comparison for the second
Figure 5.48: Sliding condition for the second
Figure 5.47: Acceleration comparison for the second
Figure 5.46: Sliding condition for the second
Figure 5.45: Acceleration comparison for the second
Figure 5.44: Sliding condition for the second
Figure 5.43: Acceleration comparison for the second
Figure 5.42: Sliding condition for the second
Figure 5.41: Acceleration comparison for the second
Figure 5.40: Sliding condition for the second
Figure 5.39: Acceleration comparison for the second
Figure 5.38: Sliding condition for the second
Figure 5.37: Acceleration comparison for the second
Figure 5.36: Sliding condition for the second
Figure 5.35: Acceleration comparison for the second
Figure 5.34: Sliding condition for the second
Figure 5.33: Acceleration comparison for the second
Figure 5.32: Sliding condition for the second
Figure 5.31: Acceleration comparison for the second
Figure 5.30: Sliding condition for the second
Figure 5.29: Acceleration comparison for the second
Figure 5.28: Sliding condition for the second
Figure 5.27: Acceleration comparison for the second
Figure 5.26: Sliding condition for the second
Figure 5.25: Acceleration comparison for the second
Figure 5.24: Sliding condition for the second
Figure 5.23: Acceleration comparison for the second
Figure 5.22: Sliding condition for the second
Figure 5.21: Acceleration comparison for the second
Figure 5.20: Sliding condition for the second
Figure 5.19: Acceleration comparison for the second
Figure 5.18: Sliding condition for the second
Figure 5.17: Acceleration comparison for the second
Figure 5.16: Sliding condition for the second
Figure 5.15: Acceleration comparison for the second
Figure 5.14: Sliding condition for the second
Figure 5.13: Acceleration comparison for the second
Figure 5.12: Sliding condition for the second
Figure 5.11: Acceleration comparison for the second
Figure 5.10: Sliding condition for the second
Figure 5.9: Acceleration comparison for the second
Figure 5.8: Sliding condition for the second
Figure 5.7: Acceleration comparison for the second
Figure 5.6: Sliding condition for the second
Figure 5.5: Acceleration comparison for the second
Figure 5.4: Sliding condition for the second
Figure 5.3: Acceleration comparison for the second
Figure 5.2: Sliding condition for the second
Figure 5.1: Acceleration comparison for the second

Figure 4.22: Sliding condition for the first
Figure 4.21: Acceleration comparison for the first
Figure 4.20: Sliding condition for the first
Figure 4.19: Acceleration comparison for the first
Figure 4.18: Sliding condition for the first
Figure 4.17: Acceleration comparison for the first
Figure 4.16: Sliding condition for the first
Figure 4.15: Acceleration comparison for the first
Figure 4.14: Sliding condition for the first
Figure 4.13: Acceleration comparison for the first
Figure 4.12: Sliding condition for the first
Figure 4.11: Acceleration comparison for the first
Figure 4.10: Sliding condition for the first
Figure 4.9: Acceleration comparison for the first
Figure 4.8: Sliding condition for the first
Figure 4.7: Acceleration comparison for the first
Figure 4.6: Sliding condition for the first
Figure 4.5: Acceleration comparison for the first
Figure 4.4: Sliding condition for the first
Figure 4.3: Acceleration comparison for the first
Figure 4.2: Sliding condition for the first
Figure 4.1: Acceleration comparison for the first
List of Tables

Table 3.1: Controller Parameters for the standard SMC.................................................................25
Table 4.1: Controller parameters for a first-order linear system with unitary control input gain ..........37
Table 4.2: Controller parameters for a first-order nonlinear system with unitary control input gain .......44
Table 4.3: Controller parameters for a second-order linear system with unitary control input gain ..........54
Table 4.4: Controller parameters for a second-order nonlinear system with unitary control input gain ......59
Table 5.1: Controller parameters for a first-order linear system with non-unitary control input gain .......70
Table 5.2: Controller parameters for a first-order nonlinear system with non-unitary control input gain .....75
Table 5.3: Controller parameters for a second-order linear system with non-unitary control input gain .....83
Table 5.4: Controller parameters for a second-order nonlinear system with non-unitary control input gain 88
Table 6.1: Controller parameters for a first-order linear system with measurement noise ....................102
Table 6.2: Controller parameters for a second-order nonlinear system with measurement noise ..........104
## Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[A(t)]$</td>
<td>State Matrix</td>
</tr>
<tr>
<td>$a$</td>
<td>Polynomial coefficients</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Constant</td>
</tr>
<tr>
<td>$[B(t)]$</td>
<td>Input Matrix</td>
</tr>
<tr>
<td>$b$</td>
<td>Control input gain</td>
</tr>
<tr>
<td>$b_u$</td>
<td>Upper bound of the control input gain</td>
</tr>
<tr>
<td>$b_l$</td>
<td>Lower bound of the control input gain</td>
</tr>
<tr>
<td>$b_p$</td>
<td>Friction coefficient at the pivot point of the pendulum</td>
</tr>
<tr>
<td>$B_\omega$</td>
<td>System bandwidth</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Control input gain constant variable</td>
</tr>
<tr>
<td>$c$</td>
<td>Damping coefficient</td>
</tr>
<tr>
<td>$d/dt$</td>
<td>Derivative in respect to time</td>
</tr>
<tr>
<td>$d/dx$</td>
<td>Derivative in respect to states</td>
</tr>
<tr>
<td>$D$</td>
<td>Open and connected subset of $\mathbb{R}$</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Constant</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Small positive constant</td>
</tr>
<tr>
<td>$\eta_0$</td>
<td>Small positive constant</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>Constant</td>
</tr>
<tr>
<td>$\varepsilon(u)$</td>
<td>Control input error</td>
</tr>
<tr>
<td>$\dot{\varepsilon}(u)$</td>
<td>Estimation of the control input error</td>
</tr>
<tr>
<td>$g$</td>
<td>Gravity constant</td>
</tr>
<tr>
<td>$K$</td>
<td>Switching gain</td>
</tr>
<tr>
<td>$\bar{K}$</td>
<td>Updated switching gain</td>
</tr>
<tr>
<td>$k$</td>
<td>Spring constant</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Slope of the sliding surface</td>
</tr>
<tr>
<td>$\lambda_l$</td>
<td>Exponential constant</td>
</tr>
<tr>
<td>$m$</td>
<td>Mass</td>
</tr>
<tr>
<td>$M$</td>
<td>Mass of the pendulum</td>
</tr>
<tr>
<td>$n$</td>
<td>Order of the system</td>
</tr>
<tr>
<td>$\Omega$</td>
<td>Lyapunov’s surface</td>
</tr>
<tr>
<td>$\varphi$</td>
<td>Boundary layer</td>
</tr>
<tr>
<td>$\dot{\varphi}$</td>
<td>Derivative of the boundary layer</td>
</tr>
<tr>
<td>$R$</td>
<td>Length of the pendulum</td>
</tr>
<tr>
<td>$\rho_n$</td>
<td>Noise probability density function</td>
</tr>
<tr>
<td>$s$</td>
<td>Sliding surface</td>
</tr>
<tr>
<td>$\dot{s}$</td>
<td>Derivative of the sliding surface</td>
</tr>
<tr>
<td>$sat(s/\varphi)$</td>
<td>Saturation function</td>
</tr>
<tr>
<td>$sgn(s)$</td>
<td>Relay function of the sliding surface</td>
</tr>
<tr>
<td>$\sigma_l$</td>
<td>Lower bound of the control input estimation error</td>
</tr>
<tr>
<td>$\sigma_u$</td>
<td>Upper bound of the control input estimation error</td>
</tr>
<tr>
<td>$\sigma_n^2$</td>
<td>Noise variance</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>$t$</td>
<td>Time</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Angle between the pendulum movement and the pendulum itself</td>
</tr>
<tr>
<td>$u$</td>
<td>Control input</td>
</tr>
<tr>
<td>$u_{k-1}$</td>
<td>Previous value of the control input</td>
</tr>
<tr>
<td>$u_{k-2}$</td>
<td>Previous value of previous control input</td>
</tr>
<tr>
<td>$V(\ddot{x})$</td>
<td>Candidate Lyapunov function</td>
</tr>
<tr>
<td>$\dot{V}(\ddot{x})$</td>
<td>Derivative of candidate Lyapunov function</td>
</tr>
<tr>
<td>$V_{pp}$</td>
<td>Peak to peak noise value</td>
</tr>
<tr>
<td>$x$</td>
<td>State measurement</td>
</tr>
<tr>
<td>$\dot{x}$</td>
<td>Derivative of state measurement</td>
</tr>
<tr>
<td>$\ddot{x}$</td>
<td>Second derivative of state measurement</td>
</tr>
<tr>
<td>$\ddot{x}$</td>
<td>Difference between state measurement and desired state</td>
</tr>
<tr>
<td>$x_0$</td>
<td>Initial value</td>
</tr>
<tr>
<td>$x^n$</td>
<td>Higher order state</td>
</tr>
<tr>
<td>$\ddot{x}$</td>
<td>Difference between state measurement and desired state</td>
</tr>
<tr>
<td>$x_d$</td>
<td>Desired state</td>
</tr>
<tr>
<td>$\dot{x}_d$</td>
<td>Derivative of the desired state</td>
</tr>
<tr>
<td>$\ddot{x}_d$</td>
<td>Second derivative of the desired state</td>
</tr>
<tr>
<td>$x_e$</td>
<td>Equilibrium point</td>
</tr>
<tr>
<td>$\bar{y}$</td>
<td>Noise mean</td>
</tr>
<tr>
<td>$y_{rms}$</td>
<td>Noise root square mean</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Motivation
Due to its robustness and ability to handle system uncertainties and disturbances, the sliding mode control (SMC) has received much attention in the past years. Since then, several different approaches have been developed, but the core remains the same and is composed by two main phases: the reaching phase and the sliding phase. The reaching phase is responsible to drive the system states onto the sliding surface while in the sliding phase the system trajectories slides through the sliding surface towards the origin. Lyapunov’s stability theory is used to guarantee asymptotic stability of the system trajectories during the reaching phase. To deal with the system uncertainties and disturbances, a discontinuous term is added to the control law form, so the controller is able to maintain state tracking onto the sliding surface. However, most all proposed SMC schemes require knowledge of the system’s mathematical model in order to develop a control law. Thus, for each system to be controlled, a different sliding mode controller must be derived and is the motivation for a new type of sliding mode controller scheme. Thus, the objective of this work is to develop a sliding mode controller which only relies on the previous control inputs, system state measurements, control input gain bounds, and system’s order, characterizing a model-free controller.

1.2 Background Research on SMC
In the next two sections, previous related research using the sliding mode control method is presented. The first section is focused on previous work that requires a system’s model to derive a sliding mode controller while the second one summarizes ongoing research involving model-free schemes.

1.2.1 General SMC Schemes
Laghrouche et al. [1] proposed a higher order sliding mode controller based on optimal linear quadratic control in order to apply to minimum-phase nonlinear SISO systems. The authors divide the problem in three steps. First, a higher order sliding mode problem is formulated in order to eliminate the chattering effect. Then, the authors consider the nonlinear uncertainties as bounded non-structured parametric uncertainties, so the system can be viewed as an uncertain linear system.
Finally, an optimal sliding mode controller is derived by designing a varying manifold (sliding surface) by minimizing a quadratic cost function over a finite amount of time. To test this SMC scheme, the author uses a kinematic model of an automobile. The objective of the control system is to steer an automobile from a given initial position onto a trajectory defined by the user. A fourth order sliding mode control was used with a time varying switching manifold (optimized by using a LQR scheme). The control system achieved excellent tracking response, with the error between the actual automobile’s trajectory and the desired one converging to zero without any chattering. The authors mention two additional advantages of this method: the simplicity of the control law and the possibility to define the convergence time a priori.

Cunha et al. [2] developed a SMC method for systems with an output tracking problem for linear multivariable systems of relative degree one. The authors define the method as a unit vector model-reference sliding mode controller (UV-MRSMC). The standard approach is to specify a desired closed-loop response using a reference model. Then, the controller is responsible for the task to track the response of this reference model by only using output measurements. Lastly, a third order system is used as example, where two different references were used as output tracking problem. The system’s output converged quickly to the reference state and the closed-loop system was globally exponentially stable, for both cases.

Yu et al. [3] proposed a new sliding mode design concept, referred to as Adaptive Seeking Sliding Mode Control (ASSM control), for a class of nonlinear systems. The authors addresses the problem of the high-gain feedback control effort that arises when the control system faces system disturbances and uncertainties. The ASSM control method has a floating control gain which is adjusted adaptively to overcome all possible unknown disturbances and uncertainties. While the method reserves all the features of classical SMC, this method is also continuous in nature and reduces the chattering effect. A cruise control system for off-road vehicles was used as example. The control system was able to follow the velocity reference, with some minor errors and the chattering effect was negligible.

Lee [4] presented a discrete-time SMC using fast output sampling. According to the author, the closed-loop system’s eigenvalues are arbitrarily assigned when designing the control system. Thus, the author designs the control system focusing in stability and transient response. To reduce the chattering effect, which is more noticeable in discrete-time, a boundary layer is used in the control
law form. A continuous time plant model with a serial type lightly damped resonance with a discrete-time controller was used as example to test the performance. The control system achieved an outstanding step response tracking and it was proved that the closed-loop system’s eigenvalues can be arbitrary assigned.

Ferrara et al. [5] addresses the problem of applying a SMC in systems with saturating actuators. The authors use a sub-optimal second-order sliding mode controller but with modifications in order to avoid control input saturation. The problem is the convergence of the sliding variable to zero in a finite amount of time is not always guarantee, if saturation occurs during the reaching phase. The proposed modification implies, in practice, once the control input reaches one of its saturation values (during the reaching phase), its value is forced to decrease successively. If the switching value has not been reached, then the control input increases again. This implies that the control input remains at the saturation value until the new switching value is reached. The authors also prove that, by using this technique, the system states converges to the origin in a finite time. Lastly, an example is presented and indeed the system states converge to the origin in a finite amount of time and the control input avoided the saturation limits.

Kai et al. [6] mentioned the problems of the sliding surface design and how it can affect the overall performance of the SMC when the sliding surface relates to uncertain physical quantities, if uncertainties occurs. Then, the authors propose a new robust design problem for the sliding surface applied for a class of uncertain MIMO nonlinear systems. The new method consists in including system’s uncertainties to the design of the sliding surface. Then, the reaching phase is designed (in accounting for the new sliding surface) to ensure stability of the closed-loop system. To test the new scheme, a second-order system was used as numerical example. Two different initial conditions were used and, in both cases, the system states converged to zero even with uncertainties included in the sliding surface.

Sen et al. [7] proposed an adaptive method based on SMC applied for quadrotor helicopters. The adaptive part of the scheme is the estimation of system’s uncertainties and perturbations bounds. The authors mention that, since those bounds are usually unknown, they are usually overestimated by the user which yields excessive gain. The excessive gain, on the other hand, is proportional to the magnitude of the chattering, which must be reduced. Thus, by estimating those bounds, the control law is updated accordingly and the chattering is greatly reduced. The method is tested using
a quadrotor helicopter for position tracking. A good tracking response was achieved with no noticeable chattering. The closed-loop system was shown to be stable, even without the knowledge of the uncertainties bounds, which were quickly estimated.

Wu et al. [8], addressed the problem of state estimation and SMC design for continuous time Markovian jump singular systems with unmeasured states. The authors mention the difficulty in designing a control system which is stable for the class of systems, since, not only asymptotic stability has to be considered, but the system regularity and impulse elimination are also needed to be examined. The problem is focused in the following dilemma: how to design an appropriate sliding surface and how to define strict LMI conditions of the stochastic stability for Markovian jump singular systems. The authors also developed an observer for the class of systems, as, in practice, system states are not always available due to the limit of physical conditions. Lastly, a Markovian jump singular system with two operating modes was illustrated as an example. The goal was to design an observer and, then, a SMC based on the state estimation provided by the observer. All the system states converged to zero for different initial conditions and the closed-loop system achieved stochastic stability.

1.2.2 Model-Free SMC Schemes

Raygosa-Barahona et al. [9] developed a second-order SMC combined with backstepping for underactuated underwater Remotely Operated Vehicles (ROVs) to track a desired path. The scheme does not have any explicit dependency with the dynamic model of the ROV, characterizing a model-free controller and was obtained by designing a regressor free second-order sliding mode controller as the auxiliary input control at each iteration of the backstepping procedure. The sliding mode theory is integrated with PID control though, which is not the case of this work, since it is based solely on the sliding mode control method. Lastly, the method is applied to a ROV with the objective to track a helix trajectory. The closed-loop response converged to the desired trajectory with no chattering.

Munoz-Vazquez et al. [10] developed a new controller method, based on SMC, in order to control the position of a quadrotor when its dynamic model is unknown. The controller is divided in three subsystems: model-free control subsystem, velocity field subsystem and sliding surface subsystem. The first subsystem is responsible to enforce the sliding mode condition for all time. The second one is used to design the velocity field and the last one assembles invariant manifolds
of position and orientation sliding surfaces. The authors tested the controller for two cases in a 3D environment, one without obstacles and the other one with. Both cases presented an outstanding tracking response, where the quadrotor followed the desired path without any chattering. However, a velocity field needs to be designed in order to derive the controller scheme.

Salgado-Jimenez et al. [11] introduced a model-free high order SMC applied to position control of a one degree-of-freedom underwater vehicle. The new method does not require knowledge of the dynamics or parameters of the underwater vehicle using only the exponential convergence of the desired trajectory. The higher order SMC, referred in [1], is used to avoid chattering, since it can damage the actuators lifetime. However, the model-free controller is integrated with a PD control, which needs to be tuned. The controller was tested in a real physical system where two trajectories were tested: a sine and a triangular wave. In both cases, a smooth response was obtained, where the underwater vehicle followed the desired path with minor errors.

Mizov and Crassidis [12] [13] developed a model-free pure sliding mode control scheme to achieve accurate tracking performance for linear and nonlinear systems along with guaranteeing asymptotic stability for tracking convergence. The proposed controller only relies in previous control inputs, state measurements and the knowledge of the system order. To reduce the chattering effect, the authors used a boundary layer in the control law form. However, tracking precision was reduced but the control effort became smooth, which is required in most of control system applications. The method was tested on first and second-order systems, linear and nonlinear. The control problem was to drive the system states onto a desired trajectory, defined by the user. In every case, outstanding tracking response was obtained and asymptotic stability of the closed-loop system was observed. In this work, the model-free sliding mode control design is extended to systems with non-unitary control input gains and state measurement noise. Besides, the system approach is different, which results into a more precise SMC controller.

1.3. Research goals

The goals of this work is to derive a new model-free sliding mode controller scheme which is solely based in previous control inputs, state measurements, on the knowledge of the system’s order and control input gain bounds, if one exists. A new system approach is proposed, compared to what was developed [12] [13], where a new variable, called estimation error of the control input, is defined. By approximating the system using this approach, a more precise controller is derived,
where better closed-loop responses are expected. In addition, robustness against state measurement noise is expected, which must be considered when implementing a control system to real physical systems and usually limits the overall performance of the controller. The controller can be applied to a class of single-input systems described by the so-called companion form, linear or nonlinear.

The outline of the thesis is as follows: Chapter 2 introduces the concepts of stability and Lyapunov’s stability theorem, which are required for the development of a model-free SMC method. Chapter 3 outlines the standard sliding mode control method along with an illustrative example of its design. In Chapter 4, the new model-free sliding mode control method for systems with unitary control input gain is derived and in Chapter 5 the same control method is derived considering non-unitary control input gains. In both chapters, illustrative examples of a first and second-order systems (linear and nonlinear) are presented and the performance is summarized. Chapter 6 deals with the measurement noise issue when the model-free sliding mode controller is implemented, and two cases are examined. Conclusions and suggestions for future work are shown in Chapter 7.
Fundamentals of Lyapunov Theory

In this chapter a review of the Lyapunov’s direct method and related stability concepts used throughout the work developed in this thesis are presented. Stability, and in particular closed-loop stability, is a desired outcome in the design of control schemes when applied in both linear and nonlinear systems. Thus, control problems are mainly centered on the study of the system’s stability and should be carefully considered.

The difference between a stable and an unstable system is that stable systems operate under an expected behavior, while unstable systems are highly unpredictable. A pendulum with pivot friction, for example, is a stable system. If a disturbance is applied to the pendulum, it will return to the original position after a finite amount of time, which is an expected behavior. If the system was not stable, the behavior of the pendulum could be uncorrelated and thus difficult to predict. Therefore, closed-loop instability is undesirable in most cases.

The traditional tool for analyzing the stability of a system is the theory introduced by Alexandr Mikhailovich Lyapunov, a Russian mathematician [14] [15]. Lyapunov proposed two stability theorems: one known as the linearization method and the other one as the direct method. The first method consist of linearizing the nonlinear system around an operational point and study the stability of the linearized system near the operating point. The second one, which is more robust, uses the concept of the energy of the system under study. In this work, only the direct method is considered in the development of a model-free control strategy.

The chapter is organized as follows: Section 2.1 defines linear and nonlinear systems. The definition of autonomous systems is presented in Section 2.2. The definition of equilibrium points and concepts of stability are presented in Section 2.3 and 2.4, respectively. Section 2.5 briefly outlines the concept of positive-definiteness while Section 2.6 presents Lyapunov’s direct method.

2.1. Nonlinear and Linear Systems

As mentioned previously, Lyapunov’s stability theory can be applied to nonlinear or linear systems. A nonlinear dynamic system can be usually represented by a set of differential equations in the form of:

\[
\dot{x} = \tilde{f}(\tilde{x}, \tilde{u}, t)
\]  

(2.1)
where $\vec{f}$ is a $nx1$ nonlinear vector function, $\vec{x}$ is the $nx1$ state vector, $\vec{u}$ is the control input and $t$ is time. The number of system states defines the order of the system. The solution of the differential equation described at Eq. (2.1) is generally referred as the state’s trajectory or system’s trajectory. They represent a curve which varies with time in the state space, while a point represents a specific value at a specific time.

The control input usually depends only on the state measurements, however, can also be time-dependent, as shown below:

$$\vec{u} = \vec{g}(\vec{x}, t)$$

To generalize the equation defined at Eq. (2.1), the equation above can be substituted, resulting the in following:

$$\dot{\vec{x}} = \vec{f}[\vec{x}, \vec{g}(\vec{x}, t), t]$$

Linear systems are special cases of the system described above. As the name suggests, the differential equations are linearly dependent of the states and the control input, which can be mathematically written as:

$$\dot{\vec{x}} = [A]\vec{x} + [B]\vec{u}$$

where $[A]$ is a $nxn$ matrix, known as state matrix, and $[B]$ is a $nxm$ matrix, known as input matrix. The value $m$ corresponds to the number of inputs of the system. For a single-input system, for example, $m = 1$.

### 2.1.1 Companion Form

The model-free SMC method can be applied to a class of nonlinear systems described by the so-called companion form. A system is said to be in companion form if its dynamics are represented by:

$$x^n = f(\vec{x}) + b(\vec{x})u$$

where $x^n$ is the higher order state, and $f(\vec{x})$ and $b(\vec{x})$ are nonlinear functions of the states. In state-space representation, Eq. (2.5) can be also be written as:
\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} x_2 \\ \vdots \\ x_{n-1} \\ x_n \\ f(\vec{x}) + b(\vec{x})u \end{bmatrix}
\]  

(2.6)

In other words, the control input and the nonlinear terms affect only the higher order state \(x^n\). For linear cases, the companion form is defined as:

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u
\]

(2.7)

where the \(a\) values are the polynomial coefficients that characterize the linear system.

### 2.2. Autonomous Systems

The system described by Eq. (2.1) and Eq. (2.4) varies with time. Also, the control input can be designed in a manner that will vary with time as shown in Eq. (2.2). Systems where their properties change with time are known as non-autonomous systems. Slotine and Li [15] defines autonomous systems as follows:

**Definition 2.2.1** - The nonlinear system is said to be autonomous if does not depend explicitly on time, i.e., if the system state equations can be written as:

\[
\dot{\vec{x}} = \vec{f}(\vec{x})
\]

(2.8)

otherwise, the system is called non-autonomous.

Obviously, linear time invariant systems are autonomous and linear time varying systems are non-autonomous. The fundamental difference between autonomous systems and non-autonomous systems is that the state’s trajectory is independent of the initial time. All physical systems, by nature, are non-autonomous systems. For example, the damping coefficient of a mass-damper-spring type problem varies as time passes by, since the spring may rust over time, which will modify the spring properties. Hence, the systems trajectories will depend on when the system is disturbed. However, most system’s properties change slowly in practice, in a manner that this effect can be neglected without causing any significant problems. For this work, all systems will be assumed to be autonomous.
2.3. Equilibrium Points

When a point represents a state trajectory, it is referred as an equilibrium point. Slotine and Li [15] defines equilibrium point as:

**Definition 2.3.1-** A state $\mathbf{x}_e$ is an equilibrium state (or equilibrium point) of the system if once $\mathbf{x}(t)$ is equal to $\mathbf{x}_e$, it remains equal to $\mathbf{x}_e$ for all future time.

Mathematically, this means that the constant vector $\mathbf{x}_e$ satisfies the equation below:

\[
\dot{\mathbf{x}} = \mathbf{0} = f(\mathbf{x}_e) \tag{2.9}
\]

This implies that once the state trajectories reach the equilibrium point, the derivatives of the states are equal to zero, which means they will not move away from the equilibrium point. Many stability concepts are focused on equilibrium points, as it will be seen later.

**2.3.1 Illustrative Example**

Consider the pendulum represented by the figure below:

![Pendulum](Image)

Figure 2.1: Example of a pendulum

The dynamics of the pendulum can be described by:

\[
MR^2 \ddot{\theta} + b_p \dot{\theta} + MgR \sin \theta = 0 \tag{2.10}
\]

where $R$ is the pendulum length, $M$ is the pendulum mass, $b_p$ is the friction coefficient at the pivot point, $g$ is the gravity constant and $\theta$ is the angle between the center of the pendulum movement and the pendulum by itself. Defining the states of the system as:

\[
x_1 = \theta \\
x_2 = \dot{\theta} \tag{2.11}
\]
And replacing into equation Eq. (2.10), we obtain:

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\frac{b}{MR^2}x_2 - \frac{g}{R}\sin x_1
\end{align*}
\]  

(12)

To find the equilibrium points, both equations must be set to zero to find the solution. Clearly, the system equilibrium points are:

\[
\begin{align*}
x_2 &= 0 \\
\sin(x_1) &= 0 \Rightarrow x_1 = n\pi, \quad n = 1, 2, 3 \ldots
\end{align*}
\]  

(13)

This example clearly shows that a nonlinear system can have infinite equilibrium points. Linear systems, however, have a single equilibrium point at the origin if the matrix \( A \) is nonsingular. If it is nonsingular, then it will have infinite equilibrium points which are contained in the null space of the matrix \( A \).

### 2.4. Concepts of Stability

Consider the following system:

\[
\dot{x} = \tilde{f}(\tilde{x})
\]  

(14)

where \( \tilde{f}: D \to \mathbb{R}^n \) is a locally Lipschitz map from \( D \) into \( \mathbb{R}^n \), \( D \) is an open and connected subset of \( \mathbb{R} \), i.e., \( D \subset \mathbb{R}^n \). In addition, another concept must be introduced, which is the concept of initial condition. As the name suggests, an initial condition is the initial value of the states at \( t = 0 \):

\[
\tilde{x}(t_0) = \tilde{x}_0
\]  

(15)

With those concepts established, Marquez [14] proposes the following definitions for stability:

**Definition 2.4.1** – The equilibrium point \( \tilde{x}_e \) is said to be stable if for each \( \varepsilon > 0 \), \( \exists \delta = \delta(\varepsilon) > 0 \):

\[
||\tilde{x}_0 - \tilde{x}_e|| \leq \delta \Rightarrow ||\tilde{x}(t) - \tilde{x}_e|| < \varepsilon, \forall t \geq t_0
\]  

(16)

In other words, for every initial condition inside the region bounded by \( \delta \), the correspondent solution remains inside the region \( \varepsilon \). This definition is the weakest concept of stability, as remaining “near” the equilibrium point is not sufficient for the majority of control problems.
**Definition 2.4.2** – The equilibrium point \( \vec{x}_e \) is said to be convergent if there exists a \( \delta_1 \) value such as:

\[
|\vec{x}_0 - \vec{x}_e| \leq \delta_1 \Rightarrow \lim_{t \to \infty} \vec{x}(t) = \vec{x}_e
\]  

What the definition means is, for every initial condition \( \vec{x}_0 \) inside the region limited by \( \delta_1 \), the correspondent solution converges to the equilibrium point as times goes to infinity. Note that stability does not imply convergence and convergence does not imply stability. These concepts may seem similar but are different.

**Definition 2.4.3** – The equilibrium point \( \vec{x}_e \) of the system is said to be asymptotically stable if it is both stable and convergent.

This definition does not determine how fast the state trajectory goes to the equilibrium point though. Hence, Marquez [14] uses this last definition for stability:

**Definition 2.4.4** – The equilibrium point \( \vec{x}_e \) of the system is said to be locally exponentially stable if there exist two real constants \( \alpha, \lambda_l \) such that:

\[
|\vec{x}(t) - \vec{x}_e| \leq \alpha |\vec{x}_0 - \vec{x}_e| e^{-\lambda_l t}, \forall t > 0
\]  

whenever \( |\vec{x}_0 - \vec{x}_e| < \delta \). The last definition means that exponential stability implies asymptotic stability, while the converse is not true. Figure 2.2 illustrates the stability concepts presented above:
The state trajectory number 1 (blue), represents an asymptotically stable system, since it is stable (is within the region $\delta$) and converge to the equilibrium point. The second state’s trajectory, number 2 (red curve), is only stable, since it does not converge to the equilibrium point but stays within the boundary region. The last state’s trajectory, number 3 (green), is clearly unstable, since it diverges away from boundary region and does not converge to any point. Another case can exist where the state’s trajectory converges but the system is not stable. For example, if the state’s trajectory drifts away from the boundary layer, but after a time returns to it and also converges to an equilibrium point, then it would be characterized as unstable and convergent.

2.5. Positive Definite Functions

The essence of the Lyapunov’s stability theory is the analysis and construction of a class of functions representing the energy of the system under study and uses the stability concepts presented previously. To be able to analyze those functions, the concept of positive-definiteness must be established. Marquez [14] uses the following definition:

**Definition 2.5.1** - A function $V: D \rightarrow \mathbb{R}$ is said to be positive semi definite in $D$ if it satisfies the following condition:

\[
\begin{align*}
(i) & \quad 0 \in D \text{ and } V(0) = 0 \\
(ii) & \quad V(\tilde{x}) \geq 0, \forall \tilde{x} \text{ in } D - \{0\}
\end{align*}
\]  

(2.19)

$V: D \rightarrow \mathbb{R}$ is said to be positive definite in $D$ if the condition (ii) is replaced by (ii’), defined below:

\[
(ii') \quad V(\tilde{x}) > 0 \text{ in } D - \{0\}
\]

(2.20)

Lastly, the function $V: D \rightarrow \mathbb{R}$ is said to be negative definite (semi definite in $D$) if $-V$ is positive definite (semi definite in $D$).

Lyapunov’s stability method is based on the notion that if a system is stable the system’s energy will dissipate as time passes by, no matter the condition and is why the concept of positive-definiteness is important. Since the objective of the Lyapunov’s direct method is to construct a function that characterizes the system’s energy, the next step is to differentiate it with respect to time and analyze the positive-definiteness (or negative-definiteness) of the resulting function, as shown below:
\[ \dot{V}(\vec{x}) = \frac{dV(\vec{x})}{dt} = \frac{\partial V(\vec{x})}{\partial \vec{x}} \frac{d\vec{x}}{dt} = \nabla V \cdot \vec{f}(\vec{x}) \]  (2.21)

As it will be seen in the next section, the equation defined above infers the system’s stability.

### 2.6. Lyapunov’s Direct Method.

As previously mentioned, Lyapunov’s direct method will be briefly explained in this work, since it has an important role for the development of the model-free SMC. The main idea of this method is to analyze the energy of the system. If the total energy of a system is continuously dissipated, then the system, whether linear or nonlinear, will eventually converge to an equilibrium point. Thus, we can draw conclusions of the system’s stability by studying that energy variation. Obviously, we must construct a function that describes the energy of the system and this function must obey some criteria. The function is commonly named as Lyapunov function. Marquez [14] defines the Lyapunov’s direct method by the following theorems:

**Theorem 2.6.1** (Stability Theorem). Let \( \vec{x} = 0 \) be an equilibrium point of \( \dot{\vec{x}} = \vec{f}(\vec{x}) \), where \( \vec{f}: D \rightarrow \mathbb{R}^n \), and let \( V: D \rightarrow \mathbb{R} \) be a continuously differentiable function such that:

1. \( V(0) = 0 \)
2. \( V(\vec{x}) > 0 \) in \( D \setminus \{0\} \)  
3. \( \dot{V}(\vec{x}) \leq 0 \) in \( D \setminus \{0\} \)  

then \( \vec{x} = 0 \) is stable.

In other words, the theorem implies that a sufficient condition for the stability of the equilibrium point \( \vec{x} = 0 \) is there exists a continuously differentiable positive definite function \( V(\vec{x}) \) such that \( \dot{V}(\vec{x}) \) is negative semi-definite in a neighborhood of \( \vec{x} = 0 \).

Positive definite functions can be used to characterize energy functions. If \( V(\vec{x}) = c \), where \( c \) is a constant, it defines what is called a Lyapunov surface. The surface defines a region of the state space that contains all Lyapunov surfaces of lesser value, i.e.:

\[ \vec{O}_1 = \{\vec{x} \in \vec{B}_r : V(\vec{x}) \leq c_1\} \]
\[ \vec{O}_2 = \{\vec{x} \in \vec{B}_r : V(\vec{x}) \leq c_2\} \]  (2.23)
where $\tilde{B}_r = \{ \tilde{x} \in \mathbb{R}^n : ||\tilde{x}|| \leq r \}$, and $c_1 > c_2$ are chosen such that $\tilde{O}_i \subset \tilde{B}_r, \ i = 1, 2, \ldots$ in a way that $\tilde{O}_2 \subset \tilde{O}_1$, as it can be seen in Figure 2.3. The condition $\dot{V}(\tilde{x}) \leq 0$ implies that when a trajectory crosses a Lyapunov surface, it can never come out of the surface again. Thus, a trajectory satisfying this condition is actually confined to the closed region $\tilde{O} = \{ \tilde{x} : V(\tilde{x}) \leq c \}$. This implies that the equilibrium point is stable, but says nothing about convergence, since the state trajectory can roam within the surface without converging to any particular point.

![Figure 2.3: Lyapunov surfaces](image)

**Theorem 2.6.2** (Asymptotic Stability Theorem). Under the conditions of Theorem 2.6.1, if $V(\tilde{x})$ is such that:

\begin{align*}
(i) \quad & V(0) = 0 \\
(ii) \quad & V(\tilde{x}) > 0 \text{ in } D - \{0\} \\
(iii) \quad & \dot{V}(\tilde{x}) < 0 \text{ in } D - \{0\}
\end{align*}

then $\tilde{x} = 0$ is asymptotically stable.

In this case, a trajectory can only move from a Lyapunov surface $V(\tilde{x}) = c$ into an inner Lyapunov surface. The condition for the derivative of the Lyapunov surface be negative definite instead of semi negative definite only strengthens the stability condition, since now the state’s trajectory will converge to an equilibrium point as the condition of negative-definiteness does not allow the state’s trajectory to remain in the same Lyapunov surface, only if that surface is the origin.

All the definitions defined above reveal that all of these concepts are local in character, i.e., depends on the initial condition of the states. For example, consider the definition 2.4.1 regarding
stability. The definition only states that if the initial conditions are near the equilibrium point, the solution (state’s trajectory) will remain near of it. For the asymptotically stability case (def. 2.4.3), the solution not only stays near the equilibrium point, but converges to it. Hence, it is vital to know the initial condition of the system as it can determine if the system’s trajectories will converge or not to an equilibrium point. There is a special case that for every possible initial condition the system’s trajectories will converge to an equilibrium point. This characteristic is known as global asymptotic stability. Marquez [14] uses the following definition for it:

**Definition 2.6.1** - The equilibrium state $\bar{x}_e$ is said to be asymptotically stable in the large, or globally asymptotically stable, if it is stable and every motion converges to the equilibrium point as $t \to \infty$.

However, suppose the following contour curve for a candidate Lyapunov function:

![Figure 2.4: Open Lyapunov surfaces](image)

Note the system’s trajectory passes through contours corresponding to smaller Lyapunov surfaces, but diverges from the equilibrium point in some cases. This happens when the contour curves of the Lyapunov function are open. To overcome that situation, a radial unboundedness condition must be set, to guarantee that all curves are closed. Marquez [14] define radial unboundedness as:

**Definition 2.6.2** - Let $V: D \to \mathbb{R}$ be a continuously differentiable function. Then $V(\bar{x})$ is said to be radially unbounded if:

$$V(\bar{x}) \to \infty \text{ as } ||\bar{x}|| \to \infty$$

Finally, the theorem for global asymptotic stability in Lyapunov sense can now be defined. Marquez [14] uses the following:
Theorem 2.6.3 (Global Asymptotic Stability). Under the conditions of Theorem 2.6.2, if \( V(\vec{x}) \) is such that:

\[
\begin{align*}
(i) \; V(0) &= 0 \\
(ii) \; V(\vec{x}) &> 0 \quad \forall \vec{x} \neq 0
\end{align*}
\]

(iii) \( V(\vec{x}) \) is radially unbounded

(iv) \( \dot{V}(\vec{x}) < 0 \quad \forall \vec{x} \neq 0 \) \hspace{1cm} (2.26)

then \( \vec{x} = 0 \) is globally asymptotically stable.

2.6.1 Illustrative Example:

Consider the following system:

\[
\begin{align*}
\dot{x}_1 &= x_2 - x_1(x_1^2 + x_2^2) \\
\dot{x}_2 &= -x_1 - x_2(x_1^2 + x_2^2)
\end{align*}
\]

Now, let us define the following Lyapunov function:

\[
V(\vec{x}) = x_1^2 + x_2^2 \tag{2.28}
\]

which clearly is positive definite. Now, differentiating Eq. (2.28) with respect to time, we obtain:

\[
\dot{V}(\vec{x}) = \frac{dV(\vec{x})}{d\vec{x}} \cdot \frac{d\vec{x}}{dt} = 2x_1\dot{x}_1 + 2x_2\dot{x}_2 = -2(x_1^2 + x_2^2)^2 \tag{2.29}
\]

which is negative definite. Also, since the states are unbounded, as \( V(\vec{x}) \to \infty \) as the states goes to infinite, the system is globally asymptotically stable. Note that the globalness characteristic implies that the only equilibrium point is set at the origin.
3 The Sliding Mode Control Method

In this chapter, an overview of the sliding mode control (SMC) method is presented with examples shown for the application of the control strategy. Slotine and Li [15] summarizes a general control design by the following: “given a physical system to be controlled and the specifications of its desired behavior, construct a feedback control law to make the closed-loop system display the desired behavior”. In general, there is two types of control problems: tracking and regulation problems. In this work, only the problem of tracking will be considered. The tracking problem can be defined as an attempt to minimize, as much as possible, the error between a desired signal and the actual output of the system under study. For example, controlling a drone along a certain desired path is a tracking problem. Hence, the objective of the controller is to assure that the drone will follow the desired path specified by the user. In reality, there is an infinite set of desired state trajectories that can be defined. Obviously, we are interested in trajectories that follow the concepts presented in the previous chapter, i.e., stable and convergent trajectories.

There are several approaches to control nonlinear systems, which are applicable to linear systems as well. Many of these methods, however, require considerable precise system models in order to derive the control law. Also, system uncertainties and disturbances are present and should be accounted for when designing a control system, since they may have noticeable negative effects on the overall performance of the controller. Fortunately, there is class of controllers that handle uncertainties and disturbances, commonly referred as robust controllers.

The sliding mode control method belong to a set of robust controllers. As the name suggests, the SMC alters the dynamics of the nonlinear system by applying a control signal forcing the system’s trajectories to slide along a sliding surface. The sliding mode control has a nominal part, but has an additional term, that is responsible in handling system uncertainties and disturbances.

The chapter is organized as follows. Section 3.1 presents the derivation of the standard sliding mode control method. Section 3.2 describes the insertion of a smoothing boundary layer into the control law and in Section 3.3 an illustration example is presented.

3.1. Derivation

Consider the following single-input system:
\[ x^n = f(\bar{x}) + b(\bar{x})u \]  
where \( \bar{x} \) is the state vector, \( u \) is the control input and \( n \) is the order of the system. In Eq. (3.1), the functions \( f(\bar{x}) \) and \( b(\bar{x}) \) are unknown, but their bound values are assumed known.

The control problem is to track a desired state vector, \( \bar{x}_d \), for the system characterized by the function defined above. Furthermore, in order to guarantee that the tracking problem will be solved using a finite control input \( u \), the following condition must be satisfied:

\[ \bar{x}_d(0) = \bar{x}(0) \]  
Otherwise, tracking can only be achieved after a transient. The equation above is quite obvious, since the states cannot jump from a certain value to another instantaneously. Now, let us define the following tracking error vector:

\[ \bar{\bar{x}} = \bar{x} - \bar{x}_d \]  
which is simply the difference between the system state and the desired state. In addition, a time-varying surface, in the state space vector, is defined as:

\[ s = \left( \frac{d}{dt} + \lambda \right)^{n-1} \bar{x} \]  
where \( \lambda \) is a strictly positive constant and is the slope of the sliding surface. Hence, given an initial condition, the problem of tracking the state vector is equivalent to that of maintaining the scalar quantity \( s \) at zero. Indeed, if the Eq. (3.4) is differentiated with respect to time the input appears in the resulting solution, so it is possible to design a control law that can solve the tracking problem. The problem of maintaining the scalar quantity \( s \) to zero, can be solved by choosing a control law such as:

\[ \frac{1}{2} \frac{d}{dt} s^2 \leq -\eta |s| \]  
where \( \eta \) is a strictly positive constant. The equation described above states that the squared distance to the surface, measured by the term \( s^2 \), decreases along all system trajectories. This implies that the state trajectories are always pointing to the sliding surface, as it can be seen in Figure 3.1. In
addition, the equality guarantees that when the system’s trajectories reach the sliding surface they will remain there, since the term $\frac{d}{dt}s^2$ will be equal to zero. Eq. (3.5) is known as sliding condition.

Figure 3.1: Sliding Condition

The control problem is summarized by designing a control law such that the sliding condition, Eq. (3.5), is always satisfied. Satisfying the sliding condition, we guarantee that the system will be asymptotically stable, since the Lyapunov’s stability criteria will be also satisfied. Still, it is important to note that we are dealing with mathematical models to design the control system, and these models are never perfect. To deal with modeling imprecision, and also with eventual disturbances, a discontinuous term must be added to the control law. The control law with the discontinuous term becomes:

$$\hat{u} = u - K \text{sgn}(s)$$

(3.6)

where $K$ is the switching gain and $\text{sgn}(s)$ is a signum function (or relay function), which is defined as:

$$\begin{align*}
\text{sgn}(s) = 1, & \text{ if } s > 0 \\
\text{sgn}(s) = -1, & \text{ if } s < 0
\end{align*}$$

(3.7)

In practice, the value of the sliding surface is never known with infinite precision and, in addition, the switching is not instantaneous. Consequently, the controller tend to produce a control signal that chatters, as it will be seen later. The effect of chattering is highly undesirable, since it leads to energy loss, control system damage, and excitation of unmodeled dynamics, due its high frequency characteristic.
In order to eliminate chattering, a smoothing boundary layer can be inserted to the control law. However, there is a tradeoff between tracking precision and smoothness. While the first approach handles parametric uncertainties, the second one guarantee robustness to high frequency unmodeled dynamics.

3.2. Boundary Layer

Many physical system types require that the chattering be reduced or eliminated. Slotine and Li [15] states that the smoothing control discontinuity essentially assigns a low pass filter structure to the local dynamics of the sliding surface, which eliminates the chattering. This can be easily understood considering that the chattering is a high frequency signal.

In order to maintain attractiveness of the boundary layer, if the boundary layer is time-varying, the sliding condition, Eq. (3.5), can be updated as:

\[
|s| \geq \varphi \rightarrow \frac{1}{2} \frac{d}{dt}s^2 \leq (\dot{\varphi} - \eta)|s|
\]  

(3.8)

The equation defined above guarantee that the distance to the boundary layer is always decreasing, similar to the original sliding condition. Also, the term implies that during the contraction of the boundary layer (\(\dot{\varphi} < 0\)) the boundary condition is more rigid while during the boundary layer expansion (\(\dot{\varphi} > 0\)) is less rigid. Furthermore, in order to satisfy Eq. (3.8), a new switching gain must be used:

\[
\bar{K} = K - \dot{\varphi}
\]  

(3.9)

And the control law becomes:

\[
\hat{u} = u - \bar{K}\text{sat}\left(\frac{s}{\varphi}\right)
\]  

(3.10)

where the \(\text{sat}\left(\frac{s}{\varphi}\right)\) function is defined as:

\[
\begin{align*}
\text{sat}\left(\frac{s}{\varphi}\right) &= \frac{s}{\varphi}, \text{ if } \left|\frac{s}{\varphi}\right| \leq 1 \\
\text{sat}\left(\frac{s}{\varphi}\right) &= \text{sgn}\left(\frac{s}{\varphi}\right), \text{ otherwise}
\end{align*}
\]  

(3.11)

Lastly, the dynamics of the boundary layer are determined by the following equation:
\[ \dot{\varphi} + \lambda \varphi = K(\ddot{x}_d) \]  

(3.12)

with \( \varphi(0) = \eta / \lambda \). Eq. (38) is known as the balance condition. With this approach, in place of “perfect” tracking, a tracking within a known precision is ensured.

### 3.3. Illustrative Example:

Consider the following system:

\[ \ddot{x} + f_1(t, x, \dot{x}) + f_2(t, x) = b(t)u \]  

(3.13)

with \( f_1(t, x, \dot{x}) = \alpha_1(t)|x|\dot{x}^2 \) and \( f_2(t, x) = \alpha_2(t)x^3 \cos(2x) \), where \( \alpha_1(t), \alpha_2(t) \) and \( b(t) \) are unknown time-varying functions with known bounds:

\[
\begin{align*}
4 &\leq b(t) \leq 7 \\
1 &\leq \alpha_1(t) \leq 2 \\
-1 &\leq \alpha_2(t) \leq 5
\end{align*}
\]

(3.14)

for \( \forall t > 0 \). The system’s trajectories will track the following function:

\[ x_d(t) = \sin \left( \frac{\pi t}{2} \right) \]  

(3.15)

The first step is to re-arrange Eq. (3.13) in terms of the higher order state, as in Eq. (3.1):

\[ \ddot{x} = -f_1(t, x, \dot{x}) - f_2(t, x) + b(t)u \]  

(3.16)

The sliding surface, described by Eq. (3.4), for a second-order system, is defined as:

\[ s = \ddot{x} + \lambda \dot{x} \]  

(3.17)

where \( \lambda \) is a positive constant and also the slope of the sliding surface, and \( \ddot{x} \) is the error between the desired state and the actual state (i.e., the measured state). To guarantee that once the state trajectories reach the sliding surface they will stay there, the derivative of the sliding surface must be set to zero:

\[ \dot{s} = \dddot{x} + \lambda \ddot{x} = 0 \]  

(3.18)

Substituting Eq. (3.13) and Eq. (3.3) into the equation above results in:

\[ \dot{s} = \dddot{x} - \ddot{x}_d + \lambda \ddot{x} = bu - f_1 - f_2 - \dddot{x}_d + \lambda \ddot{x} = 0 \]  

(3.19)

Finally, writing the equation in terms of the control input, the following control law is obtained:
\[ u = b^{-1} \{ f_1 + f_2 + [\ddot{x}_d - \lambda \dot{x}] \} \]  
(3.20)

Since the system is partially defined, i.e., contains uncertainties, a discontinuous term is added to the control law in order to project the system trajectories onto the sliding surface:

\[ u = b^{-1} \{ f_1 + f_2 + [\ddot{x}_d - \lambda \dot{x} - \eta \text{sgn}(s)] \} \]  
(3.21)

where \( \eta \) is a small positive constant and \( \text{sgn}(s) \) is a signum function, or relay function. To ensure that the system will be asymptotically stable during the reaching phase, the Lyapunov’s stability theorem must be applied. The following Lyapunov function is used:

\[ V(\ddot{x}) = \frac{1}{2} s^2 \]  
(3.22)

which is clearly a positive definite function and radially unbounded. Differentiating the Lyapunov function with respect to time:

\[ \dot{V}(\ddot{x}) = \frac{\partial V(\ddot{x})}{\partial s} \frac{\partial s}{\partial t} = s \dot{s} \]  
(3.23)

The function defined above must be negative definite to ensure global asymptotic stability in Lyapunov’s sense. Replacing Eq. (3.19) into it:

\[ \dot{V}(\ddot{x}) = s \dot{s} = s(bu - f_1 - f_2 - \ddot{x}_d + \lambda \dot{x}) < 0 \]  
(3.24)

Now, substituting the control law by Eq. (3.21):

\[ s(bb^{-1} \{ f_1 + f_2 + [\ddot{x}_d - \lambda \dot{x} - \eta \text{sgn}(s)] \} - f_1 - f_2 - \ddot{x}_d + \lambda \dot{x}) < 0 \]  
(3.25)

The result can be simplified into:

\[ \dot{V}(\ddot{x}) < -\eta |s| \]  
(3.26)

which is clearly a negative definite function. Thus, the control law was constructed correctly. Still, in place of using the value of \( \eta \) in the control law, which is a constant, a switching gain is required to handle the system uncertainties and disturbances. Therefore:

\[ u = b^{-1} \{ f_1 + f_2 + [\ddot{x}_d - \lambda \dot{x} - K\text{sgn}(s)] \} \]  
(3.27)

In physical systems, the exact values of the system’s parameters are never known. However, the bounds are usually known, which makes it possible to estimate those values and design a controller.
Replacing the closed-loop system, the sliding condition, Eq. (3.5), must be satisfied:

\[
\frac{1}{2} \frac{d}{dt} s^2 = s \dot{s} = s \left( b \ddot{u} - f_1 - f_2 - \ddot{x}_d + \lambda \ddot{x} \right) \leq -\eta |s| \quad (3.30)
\]

Replacing the updated control law by Eq. (3.29):

\[

s \dot{s} = s \left( b \hat{b}^{-1} \left[ \hat{f}_1 + \hat{f}_2 + [\ddot{x}_d - \lambda \ddot{x} - K \text{sgn}(s)] \right] \right) - f_1 - f_2 - \ddot{x}_d + \lambda \ddot{x} \quad (3.31)

\]

which can be rewritten as:

\[
s \dot{s} = s \left( (b \hat{b}^{-1}) \hat{f}_1 - f_1 + b \hat{b}^{-1} \hat{f}_2 - f_2 + (b \hat{b}^{-1}) \ddot{x}_d - \ddot{x}_d - (b \hat{b}^{-1}) \lambda \ddot{x} + \lambda \ddot{x} 

- (b \hat{b}^{-1}) (K) \text{sgn}(s) \right) \leq -\eta |s| \quad (3.32)
\]

Rewriting the equation above in terms of the switching gain:

\[
K |s| \geq s \left( (1 - \hat{b} b^{-1}) \hat{f}_1 - \hat{b} b^{-1} f_1 + \hat{b} b^{-1} \hat{f}_2 - \hat{b} b^{-1} f_2 + (1 - \hat{b} b^{-1}) \ddot{x}_d - \ddot{x}_d - (1 - \hat{b} b^{-1}) \lambda \ddot{x} + \hat{b} b^{-1} \lambda \ddot{x} \right) 

+ (1 - \hat{b} b^{-1}) \eta |s| \quad (3.33)
\]

Defining \( \hat{f}_i = f_i - (f_i - \hat{f}_i) \), where \( i = 1 \) or \( 2 \), and replacing into the equation defined above:

\[
K |s| \geq s \left( (1 - \hat{b} b^{-1}) \hat{f}_1 - \hat{b} b^{-1} f_1 - \hat{b} b^{-1} \hat{f}_2 - \hat{b} b^{-1} f_2 

+ (1 - \hat{b} b^{-1}) (\ddot{x}_d - \lambda \ddot{x}) \right) + (\hat{b} b^{-1}) \eta |s| \quad (3.34)
\]

Replacing \( \hat{b} b^{-1} \) by \( \beta \), which was defined at Eq. (3.28):
\[
K|s| \geq s \left\{ (1 - \beta) \hat{f}_1 - \beta (f_1 - \hat{f}_1) + (1 - \beta) \hat{f}_2 - \beta (f_2 - \hat{f}_2) + (1 - \beta) (\dot{x}_d - \lambda \hat{x}) \right\} \\
+ (\beta) \eta |s|
\]  
(3.35)

Since the equality states greater or equal, the absolute value of all terms is used to ensure that the controller will be the most conservative possible, i.e., will work for the most extreme case possible:

\[
K|s| = |s| \left\{ |(1 - \beta) \hat{f}_1| + |\beta (f_1 - \hat{f}_1)| + |(1 - \beta) \hat{f}_2| + |\beta (f_2 - \hat{f}_2)| \\
+ |(1 - \beta) (\dot{x}_d - \lambda \hat{x})| \right\} + (\beta) \eta |s|
\]  
(3.36)

Dividing both sides by |s|:

\[
K = |(1 - \beta) \hat{f}_1| + |\beta (f_1 - \hat{f}_1)| + |(1 - \beta) \hat{f}_2| + |\beta (f_2 - \hat{f}_2)| \\
+ |(1 - \beta) |(\dot{x}_d - \lambda \hat{x})| + (\beta) \eta
\]  
(3.37)

where \( \hat{f}_1 \) and \( \hat{f}_2 \) are defined as follows:

\[
\begin{align*}
\hat{f}_1 &= \hat{\alpha}_1 |x| \hat{x}^2 = \sqrt{2} |x| \hat{x}^2 \\
\hat{f}_2 &= \hat{\alpha}_2 x^3 \cos (2x) = 3x^3 \cos (2x)
\end{align*}
\]  
(3.38)

### 3.3.1 Simulation

With the control law, Eq. (3.29), sliding surface, Eq. (3.4), and switching gain, Eq. (3.37), defined, a control system was designed using Simulink and MATLAB to control the system described at Eq. (3.13). A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. For this simulation, the following desired tracking function was used:

\[
x_d = \sin \left( \frac{\pi}{2} t \right)
\]  
(3.39)

The controller parameters used are shown in the Table 3.1 below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>20</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 3.1: Controller Parameters for the standard SMC
To test the robustness of the controller against system uncertainties, the parameters $a_1$, $a_2$ and $b$ were made time varying, but still bounded within the assumed known values, as it can be seen in the figure below:

![Variation of the parameters over time](image)

Figure 3.2: Variation of system parameters

Figure 3.3 shows that the position state follows the desired position almost perfectly. Figure 3.4 displays the position tracking error for the closed-loop system and shows that the error is less than $14e^{-05}$, which proves precise tracking of the sliding mode controller.

![Position Comparison](image) ![Position tracking error](image)

Figure 3.3: Position comparison  
Figure 3.4: Position tracking error

Figure 3.5 shows that the difference between the velocity state and the desired one is negligible. Figure 3.6 displays the velocity tracking error of the system. The error is minimal too, with values less than $2.5e^{-04}$, but the effect of chattering can already be noticed, due to the relay function used in the control law.
Figure 3.5: Velocity comparison

Figure 3.6: Velocity tracking error

Figure 3.7 shows that the acceleration state’s trajectory is in agreement with the general reference signal, however, with poor performance. Figure 3.8 shows that the effect of chattering is large, which resulted in errors with magnitude of $4\times10^4$, as is the double of the desired acceleration amplitude.

Figure 3.7: Acceleration comparison

Figure 3.8: Acceleration tracking error

Figure 3.9 shows that there is chattering of the control effort, which is unacceptable. Figure 3.10 displays the sliding condition of the closed-loop system. The sliding condition is satisfied during all simulation time.
In order to remove the chattering effect, a boundary layer is added to the control law. The high frequency chattering signal makes the controller unfeasible to implement in physical systems, since it excites unmodeled dynamics and can possibly damage the control system actuators.

### 3.3.2 Inclusion of the boundary layer

As mentioned in Section 3.2, a method to remove the chattering effect is to add a smoothing boundary layer to the control law. Therefore, the control law becomes:

\[
\hat{u} = \hat{b}^{-1} \left\{ \hat{f}_1 + \hat{f}_2 + \left[ \ddot{x}_d - \lambda \dot{x} - \tilde{K} \text{sat} \left( \frac{s}{\varphi} \right) \right] \right\}
\]  

(3.40)

where the switching gain is computed by the following:

\[
K(x_d) = \left| (1 - \beta)\hat{f}_1(x_d) \right| + \beta \left| f_1 - \hat{f}_1(x_d) \right| + \left| (1 - \beta)\hat{f}_2(x_d) \right| + \beta \left| f_2 - \hat{f}_2(x_d) \right| + \left| (1 - \beta) \left\| \left( \ddot{x}_d - \lambda \dot{x} \right) \right\| + (\beta) \eta \right|
\]  

(3.41)

with:

\[
\begin{align*}
\hat{f}_1 &= \hat{\alpha}_1 |x_d|\dot{x}_d^2 = \sqrt{2}|x_d|\dot{x}_d^2 \\
\hat{f}_2 &= \hat{\alpha}_2 x_d^3 \cos(2x_d) = 3x_d^3 \cos(2x_d)
\end{align*}
\]

(3.42)

Thus, using these updated equations for the control law, Eq. (3.40), and for the switching gain, Eq. (3.41), a Simulink diagram was built using MATLAB. As the other simulation, a sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The
identical function defined at Eq. (3.39) was used as reference and the same controller parameters (Table 3.1) were used. The parameters were made time-varying, as shown in Figure 3.2.

Figure 3.11 shows that the state position trajectory is tracking the desired position nearly perfectly. Figure 3.12 displays the position tracking error for the closed-loop system and shows that the error is less than $4e^{-03}$, which is not as precise as the previous simulation but is a small and acceptable error value.

Figure 3.11: Position comparison

Figure 3.12: Position tracking error

Figure 3.13 shows that the difference between the velocity trajectory and the reference signal is negligible. Figure 3.14 displays the velocity tracking error of the system. The error is greater compared to the previous simulation, but it is still minimal, with values less than $2.5e^{-03}$. The advantage of using the boundary layer can already be noticed. The chattering effect is eliminated and the response is quite smooth.
Figure 3.15 shows that the acceleration tracking response is in good agreement. Figure 3.16 displays the acceleration tracking error. The effect of chattering has been eliminated, implying into errors less than 8e-03. The response is more acceptable compared to the controller implemented with the relay function.

Figure 3.17 shows that the chattering effect of the control effort has been eliminated. Figure 3.18 displays the updated sliding condition using the boundary layer in place of the discontinuous function. The sliding condition is satisfied during the simulation time, since the sliding surface stays within the boundary layer.
For the controller implemented with a smoothing boundary layer, it was observed a loss of tracking precision for the position and velocity states. However, the tracking precision is within acceptable limits with an advantage gained in the elimination of control effort chattering. Due to the smoothing nature of the insertion of a boundary layer the acceleration response tracking is vastly improved. An important aspect of the sliding mode controller with a smoothing boundary layer is that it is possible to implement the controller in the majority of physical systems, since the chattering effect is eliminated and the responses are smooth.
4 Model-Free SMC with unitary control input gain

In this chapter, a model-free sliding mode controller is derived for systems with unitary control input gain. As shown in the previous chapter, the sliding mode controller can compensate for system disturbances and uncertainties. While system disturbances are caused by external factors, system uncertainties are caused by system modelling imprecision. Thus, system modelling is vital when designing a control system, as it can drastically affect the overall performance of the controller. In addition, if the system model is considerably complex, the derivation of the sliding mode control law can become cumbersome.

In order to avoid developing a system model, a new model-free sliding mode control scheme is proposed. The only knowledge required about the system to be controlled is the order of the system, assuming the control input gain is unitary. In addition, the system is assumed to be in companion form. The model-free sliding mode controller is solely based on the previous control inputs and the state measurements.

The chapter is organized as follows. Section 4.1 presents the system “model” approach that is used to derive the model-free sliding mode controller. Section 4.2 derives a model-free sliding mode controller for a first-order system. In Section 4.3, the model-free sliding mode controller is derived for a second-order system. In both of the last two sections, examples are presented and results are examined.

4.1 System Description

Consider a $n^{th}$-order single-input autonomous system. The following equality holds true for the system:

$$x^n = x^n + u - u_{k-1} - u + u_{k-1}$$

(4.1)

where $x^n$ represents the higher order state, $u$ is the control input, and $u_{k-1}$ is the previous value of the control input. The error between the control input and the previous control input is defined as:

$$\varepsilon(u) = u_{k-1} - u$$

(4.2)

Hence, the system can be written as:
\[ x^n = x^n + u - u_{k-1} + \varepsilon(u) \]  
(4.3)

In order to compute the control law, and to avoid an algebraic loop within the controller algorithm, an estimation of the control input error, defined at Eq. (4.4), is necessary. Thus, the estimation of the control input error is defined as:

\[ \hat{\varepsilon}(u) = u_{k-2} - u_{k-1} \]  
(4.4)

where \( u_{k-2} \) is the previous control input of the previous control input. Although the control input error is not known exactly, the error is assumed to be bounded as follows:

\[ (1 - \sigma_l)\hat{\varepsilon}(u) \leq \varepsilon(u) \leq (1 + \sigma_u)\hat{\varepsilon}(u) \]  
(4.5)

where \( \sigma_u \) is the upper bound and \( \sigma_l \) is the lower bound of the control input estimation error. If the sampling time is high enough, the values of the error’s bounds will be near zero since the estimation error will be approximately equal to the actual error. Since the next step to derive the model-free sliding mode controller requires the order of the system, the derivation is divided in two parts. The first part presents the derivation for a first-order system while the second one presents the derivation for a second-order system.

4.2. First-Order System

For a first-order system, Slotine and Li [15] defines the sliding surface as:

\[ s = x - x_d + \lambda \int_0^t (x - x_d) dr \]  
(4.6)

Differently from Eq. (3.4), the sliding surface defined above includes \( \lambda \) in the equation, which improves the controller overall performance. To ensure the state tracking trajectories remain on the sliding surface once they reach it, no movement should be allowed. In this manner, the system’s states remain inside the sliding surface once they reach it. Therefore:

\[ \dot{s} = \dot{x} - \dot{x}_d + \lambda (x - x_d) = 0 \]  
(4.7)

Replacing Eq. (4.1) into Eq. (4.7), the following is obtained:

\[ \dot{x} + u - u_{k-1} + \varepsilon(u) - \dot{x}_d + \lambda (x - x_d) = 0 \]  
(4.8)

Re-arranging the equation above in terms of the control input \( u \) results in:
\[ u = -(\dot{x} - \dot{x}_d) - \lambda(x - x_d) + u_{k-1} - \epsilon(u) \quad (4.9) \]

To the controller achieve robustness against system uncertainties and disturbances, a discontinuous term is added to the control law in order to move the system’s states back onto the sliding surface. Hence, the updated control law with the discontinuous term becomes:

\[ u = -(\dot{x} - \dot{x}_d) - \lambda(x - x_d) + u_{k-1} - \epsilon(u) - \eta \text{sgn}(s) \quad (4.10) \]

where \( \eta \) is a small positive constant and \( \text{sgn}(s) \) is the relay function of the sliding surface.

### 4.2.1 Proof of the Controller Form

To ensure that the closed-loop system’s trajectories will be asymptotically stable during the reaching phase, Lyapunov’s direct method is used. Thus, it is necessary a function that describes the system’s energy, which is defined as:

\[ V(\vec{x}) = \frac{1}{2} s^2 \quad (4.11) \]

Clearly the function defined above is positive definite, which implies that the system has initially positive energy. To obtain the energy rate of the system, Eq. (4.11) is differentiated with respect to time:

\[ \dot{V}(\vec{x}) = \frac{dV(\vec{x})}{d(\vec{x})} \frac{d\vec{x}}{dt} = \dot{s}s \leq 0 \quad (4.12) \]

Replacing Eq. (4.7) into the derivative of the Lyapunov function, the following is obtained:

\[ \dot{V}(\vec{x}) = s(\dot{x} - \dot{x}_d + \lambda(x - x_d)) \leq 0 \quad (4.13) \]

which can be further manipulated using Eq. (4.1):

\[ V(\vec{x}) = s(\dot{x} + u - u_{k-1} + \epsilon(u) - \dot{x}_d + \lambda(x - x_d)) \leq 0 \quad (4.14) \]

Finally, replacing the control law by Eq. (4.10):

\[ \dot{V}(\vec{x}) = s(\dot{x} - \dot{x}_d + u_{k-1} - \lambda(x - x_d) + \lambda(x - x_d) - \epsilon(u) - \eta \text{sgn}(s) - u_{k-1} \]

\[ + \epsilon(u) - \dot{x}_d) \leq 0 \quad (4.15) \]

which can be simplified to:

\[ \dot{V}(\vec{x}) = s(-\eta \text{sgn}(s)) \leq 0 \quad (4.16) \]
Using Eq. (3.7), the equation defined above can be simplified as:

\[-\eta |s| \leq 0 \quad (4.17)\]

Since \( \eta \) can only be positive values, the negative-definiteness of the Eq. (4.17) is assured. As the Lyapunov’s direct method is satisfied, the closed-loop system is asymptotically stable and the control law form defined at Eq. (4.10) is correct.

### 4.2.2 Switching Gain

The control law defined at Eq. (4.10) is updated as:

\[u = -(\dot{x} - \dot{x_d}) - \lambda (x - x_d) + u_{k-1} - \hat{\epsilon}(u) - Ksgn(s) \quad (4.18)\]

where \( \hat{\epsilon}(u) \) is the estimation of the control input error, described at Eq. (4.4), and \( K \) is the switching gain required to ensure asymptotic stability of the closed-loop system during the reaching phase. Thus, the sliding condition defined in Eq. (3.5) must be satisfied:

\[ss \dot{s} \leq -\eta |s| \quad (4.19)\]

Performing the same procedures as before (used to proof the controller form), but now with the updated control law, Eq. (4.18), the following is obtained:

\[s(\epsilon(u) - \hat{\epsilon}(u) - Ksgn(s)) \leq -\eta |s| \quad (4.20)\]

In order to be the most conservative possible, the upper bound of the control input error is chosen:

\[\epsilon(u) = (1 + \sigma_u)\hat{\epsilon}(u) \quad (4.21)\]

Replacing Eq. (4.21) into Eq. (4.20):

\[s(\epsilon(u) - \hat{\epsilon}(u) + \sigma_u \hat{\epsilon}(u) - Ksgn(s)) \leq -\eta |s| \quad (4.22)\]

which can be simplified to:

\[-K |s| + s\sigma_u \hat{\epsilon}(u) \leq -\eta |s| \quad (4.23)\]

Rewriting the equation above in terms of the switching gain:

\[\eta |s| + s\sigma_u \hat{\epsilon}(u) \leq K |s| \quad (4.24)\]

Once more, to be most conservative possible, the absolute value of both sides of equation is used,
which results into the following switching gain after dividing both sides by the absolute value of the sliding surface:

\[ K = |\sigma_u \dot{\epsilon}(u)| + \eta \]  
(4.25)

Replacing the estimation of the control input error by Eq. (4.4), the following is obtained:

\[ K = |\sigma_u (u_{k-2} - u_{k-1})| + \eta \]  
(4.26)

The control law can also be updated using Eq. (4.4):

\[ u = -(\dot{x} - \dot{x}_d) - \lambda (x - x_d) + 2u_{k-1} - u_{k-2} - K \text{sgn}(s) \]  
(4.27)

### 4.2.3 Boundary Layer

As shown in Section 3.3.1, the sliding mode controller introduces a high frequency signal to the closed-loop system, known as chattering, due the discontinuous term added to the control law. In order to reduce the chattering, a smoothing boundary layer is added in the control law, in place of the relay function. The procedure to add the boundary layer was shown with more details in Section 3.2. Hence, using Eq. (3.9), the control law becomes:

\[ u = -(\dot{x} - \dot{x}_d) - \lambda (x - x_d) + 2u_{k-1} - u_{k-2} - (K - \dot{\phi}) \text{sat}\left(\frac{s}{\phi}\right) \]  
(4.28)

Replacing the switching gain by Eq. (4.26), the control law is updated as:

\[ u = -(\dot{x} - \dot{x}_d) - \lambda (x - x_d) + 2u_{k-1} - u_{k-2} - (|\sigma_u (u_{k-2} - u_{k-1})| + \eta - \dot{\phi}) \text{sat}\left(\frac{s}{\phi}\right) \]  
(4.29)

where the boundary layer dynamics are defined as:

\[ \dot{\phi} = -\lambda \phi + \sigma_u (u_{k-2} - u_{k-1}) + \eta \]  
(4.30)

with \( \phi(0) = \eta/\lambda \).

### 4.2.4 Illustrative Examples

Two illustrative examples are presented next, one as a first-order linear system and the other as a first-order nonlinear system. In both examples, the model-free sliding mode controller is implemented with the relay function, Eq. (4.27), and with the smoothing boundary layer, Eq. (4.28).
4.2.4.1 First-order linear system

Consider the following first-order linear model:

\[ \dot{x} + 5x = u \]  \hspace{1cm} (4.31)

where \( u \) is the control input, and \( \dot{x} \) and \( x \) are the state measurement variables of the system. The tracking problem will track the reference signal defined as:

\[ x_d(t) = \sin \left( \frac{\pi}{2} t \right) \]  \hspace{1cm} (4.32)

Using the control law, Eq. (4.27), and the switching gain, Eq. (4.26), a Simulink model was built, as shown in Figures 4.1, 4.2, 4.3 and 4.4. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>20</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4.1: Controller parameters for a first-order linear system with unitary control input gain

Figure 4.1: Open-Loop system of the model described at Eq. (4.31)

Figure 4.2: Switching gain described at Eq. (4.26)
Figure 4.3: Control law described at Eq. (4.27)

Figure 4.4: Desired Tracking block

Figure 4.5 shows that the state position trajectory follows the desired position with small errors. The position tracking error is shown in Figure 4.6. The error is less than 3e-05, which is negligible.

Figure 4.5: Position comparison for the linear first-order system example with relay function

Figure 4.6: Position tracking error for the linear first-order system example with relay function
Figure 4.7 shows that the agreement between the acceleration trajectory and the desired one is not acceptable. Due to the chattering introduced by the discontinuous term, the error is large. Figure 4.8 displays the velocity tracking error of the closed-loop system, which is less than 0.2.

Figure 4.7: Velocity comparison for the linear first-order system example with relay function

Figure 4.8: Velocity tracking error for the linear first-order system example with relay function

Figure 4.9 displays the sliding condition of the closed-loop system. The sliding condition is satisfied at all times. The control effort, displayed by Figure 4.10, displays significant chattering, which is unacceptable.

Figure 4.9: Sliding condition for the linear first-order system example with relay function

Figure 4.10: Control effort for the linear first-order system example with relay function

As shown in the previous results, there is a peak at the very beginning in the state trajectories and in the sliding condition plots due to an algebraic loop. As the control law uses the highest order state to compute the control signal, the highest order state will feed itself through the control input. For
a first-order system, the term $\dot{x}$ feeds itself through the control input, which results in an algebraic loop and also explains the chatter observed in $\dot{x}$, since the control input chatters as well. For the model-free sliding mode control scheme, the higher order state will always feed itself through the control input. Fortunately, the algebraic loop will not occur when the model-free sliding controller is implemented in real physical systems, since the measurements of the system’s states will be performed by a sensor, not a software solver. For that reason, this effect will be neglected for the rest of the simulations.

The model-free sliding mode controller scheme achieved an acceptable tracking response for the position state, but an unacceptable one for the velocity state. The control effort chattering, which is caused by the discontinuous function added to the control law, greatly affected the overall performance of the controller. To reduce the chattering, a smoothing boundary layer is included.

### 4.2.4.1.1 Inclusion of the boundary layer

One method to reduce, or remove, the control effort chattering is to add a smoothing boundary layer to the control law. The model-free sliding mode controller with boundary layer for a first-order system is shown in Section 4.2.3. The identical controller parameters used in the previous example, Table 4.1, and the same reference function, Eq. (4.32), are implemented.

Using the control law with implemented with boundary layer, Eq. (4.28) and the switching gain defined in Eq. (4.29), a Simulink model was built, as shown in Figures 4.11, 4.12, 4.13, 4.14, 4.15 and 4.16. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds.

![Figure 4.11: Open-Loop system of the model described at Eq. (4.31)](image-url)
Figure 4.12: Switching gain described at Eq. (4.26)

Figure 4.13: Control law with described at Eq. (4.28)

Figure 4.14: Desired Tracking block

Figure 4.15: Sign or Sat. block
Figure 4.16: Boundary Layer block

Figure 4.17 shows that the agreement between the state position’s trajectory and the desired one is outstanding. The position tracking error is displayed by Figure 4.18. The error is less than 1e-06, which is smaller compared to the position tracking error obtained using the controller implemented with the relay function.

Figure 4.17: Position comparison for the linear first-order system example with boundary layer  
Figure 4.18: Position tracking error for the linear first-order system example with boundary layer

Figure 4.19 shows that the velocity state trajectory is tracking the desired one with acceptable performance. The velocity tracking error is less than 3e-04, as displayed by Figure 4.20. The velocity tracking response greatly improved due the elimination of the chattering.

Figure 4.19: Position comparison for the linear first-order system example with boundary layer  
Figure 4.20: Position tracking error for the linear first-order system example with boundary layer
Figure 4.19: Velocity comparison for the linear first-order system example with boundary layer

Figure 4.20: Velocity tracking error for the linear first-order system example with boundary layer

Figure 4.21 displays the updated sliding condition of the closed-loop system. The sliding surface remains inside the boundary layer during the simulation time, which implies that the sliding condition is satisfied. The chattering was eliminated, since the control effort is smooth, as displayed by Figure 4.22.

The model-free sliding mode controller implemented with the smoothing boundary layer achieved better results compared to the one implemented with the relay function. The acceleration tracking response vastly improved. It was also observed an improvement in the position tracking response. The chattering was completely eliminated, which makes this controller applicable to real physical systems.
4.2.4.2 First-order nonlinear system

Consider the following first-order nonlinear model:

\[ \dot{x} + 5x^2 = u \]  

(4.33)

where \( u \) is the control input and, \( \dot{x} \) and \( x \) are the state measurement variables of the system. The tracking problem is to track the reference signal defined as:

\[ x_d(t) = \sin\left(\frac{\pi}{2} t\right) \]  

(4.34)

Using the control law, defined at Eq. (4.27) and the switching gain, Eq. (4.26), a Simulink model was built, as shown in Figures 4.23, 4.24, 4.25 and 4.26. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

Table 4.2: Controller parameters for a first-order nonlinear system with unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>20</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Figure 4.23: Open-Loop system of the model described at Eq. (4.33)

Figure 4.24: Switching gain described at Eq. (4.26)
Figure 4.25: Control law described at Eq. (4.27)

Figure 4.26: Desired Tracking block

Figure 4.27 shows that the state position measurement follows the desired position with small errors. The error is less than $3 \times 10^{-5}$, as displayed by Figure 4.28.

Figure 4.27: Position comparison for the nonlinear first-order system example with relay function

Figure 4.28: Position tracking error for the nonlinear first-order system example with relay function
Figure 4.29 shows that the velocity state is in agreement with the reference signal, however with poor performance. Figure 4.30 displays the velocity tracking error of the closed-loop system. The error is less than 0.25, which is unacceptable.

Figure 4.29: Velocity comparison for the nonlinear first-order system example with relay function
Figure 4.30: Velocity tracking error for the nonlinear first-order system example with relay function

Figure 4.31 displays the sliding condition of the closed-loop system and it is satisfied during the simulation time. Figure 4.32 shows that the control effort has a significant amount of chattering, which must be reduced or eliminated.

Figure 4.31: Sliding condition for the nonlinear first-order system example with relay function
Figure 4.32: Control effort for the nonlinear first-order system example with relay function

The model-free sliding mode controller scheme showed to be also applicable for nonlinear systems. The position track response is acceptable, but the velocity tracking response is not. The control effort has chattering, which must be eliminated in order to implement this controller into
real physical systems. However, it is confirmed that this controller doesn’t need to be updated or modified regarding the system under study, proving to be a model-free scheme. The same controller that was used for the linear example was used for the nonlinear.

### 4.2.4.2.1 Inclusion of the boundary layer

The procedure to implement the model-free sliding mode controller with boundary layer for a first-order system can be seen with more details in Section 4.2.3. The identical controller parameters of the previous example, Table 4.2, and the same reference function, Eq. (4.34), are used next.

Using the control law implemented with the boundary layer, Eq. (4.28), and the switching gain defined at Eq. (4.29), a Simulink model was built, as shown in Figures 4.33, 4.34, 4.35, 4.36, 4.37 and 4.38. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds.

![Figure 4.33: Open-Loop system of the system described at Eq. (4.33)](image1)

![Figure 4.34: Control law described at Eq. (4.28)](image2)
Figure 4.35: Desired Tracking block

Figure 4.36: Switching gain described at Eq. (4.26)

Figure 4.37: Sign or Sat. block

Figure 4.38: Boundary Layer block

Figure 4.39 shows that the agreement between the state position and the desired one is outstanding. As displayed by Figure 4.40, the position tracking error is extremely small, with values less than 2e-06. The position tracking response improved compared to the one obtained implementing the
controller with relay function.

Figure 4.39: Position comparison for the nonlinear first-order system example with boundary layer
Figure 4.40: Position tracking error for the nonlinear first-order system example with boundary layer

Figure 4.41 shows that the velocity tracking response greatly improved, compared to the results obtained with the control system implemented with the relay function. Figure 4.42 displays the velocity tracking error of the closed-loop system. The error is less than $3\times10^{-4}$, which is once again extremely small.

Figure 4.43 displays the updated sliding condition of the closed-loop system. The sliding surface remains inside the boundary layer all simulation time, which implies that the sliding condition is satisfied and the closed-loop system is asymptotically stable. The control effort chattering was eliminated, as shown in Figure 4.44.
The model-free sliding mode controller implemented with the smoothing boundary layer achieved better tracking results compared to the one implemented with the relay function, for the nonlinear and linear examples. The insertion of the boundary layer smoothed the control effort, eliminating the chatter and vastly improving the state velocity tracking response. The position tracking response slightly improved as well.

For the next examples illustrated in this thesis, only the controller implemented with the smoothing boundary layer will be used. Due the poor performance of the model-free sliding mode controller implemented with the relay function, and to its infeasibility to implement into real physical systems due the chattering, this controller will be ignored.

4.3. Second-Order System

For a second-order system, the following sliding surface is obtained using Eq. (3.4):

\[ s = \dot{x} - \dot{x}_d + \lambda (x - x_d) \]  

The control law is derived by guaranteeing that once the state trajectories reach the sliding surface they will remain there. Thus, the sliding surface must be differentiated with respect to time and resulting equation must be set to zero, as shown below:

\[ \dot{s} = \ddot{x} - \ddot{x}_d + \lambda (\dot{x} - \dot{x}_d) = 0 \]  

Replacing Eq. (4.1) into Eq. (4.36), the following is obtained:
\[ \ddot{x} + u - u_{k-1} + \varepsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d) = 0 \quad (4.37) \]

Re-arranging the equation above in terms of the control input \( u \):

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - \varepsilon(u) + u_{k-1} \quad (4.38) \]

Since physical systems are partially defined, i.e., contains uncertainties, a discontinuous term is added to the control law in order to move the system’s trajectories back to the sliding surface when they move out due those uncertainties. Thus, the control law becomes:

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - \varepsilon(u) + u_{k-1} - \eta sgn(s) \quad (4.39) \]

where \( \eta \) is a small positive constant and \( sgn(s) \) is the relay function of the sliding surface.

### 4.3.1 Proof of the Controller Form

To ensure that the closed-loop system’s trajectories will be asymptotically stable during the reaching phase, Lyapunov’s direct method is used. As mentioned in Section 2.6, we need a function that describes the system’s energy, which is defined as:

\[ V(\bar{x}) = \frac{1}{2} \bar{x}^2 \quad (4.40) \]

Clearly the function defined above is positive definite, which means that the system initially has positive energy. To obtain the variation of the system’s energy, Eq. (4.40) is differentiated with respect to time, as shown below:

\[ \dot{V}(\bar{x}) = \frac{dV(\bar{x})}{d\bar{x}} \frac{d\bar{x}}{dt} = \dot{s}s \leq 0 \quad (4.41) \]

Replacing Eq. (4.36) into the derivative of the Lyapunov function, the following is obtained:

\[ \dot{V}(\bar{x}) = s(\ddot{x} - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \leq 0 \quad (4.42) \]

which can be further manipulated using Eq. (4.1):

\[ \dot{V}(\bar{x}) = s(\ddot{x} + u - u_{k-1} + \varepsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \leq 0 \quad (4.43) \]

Finally, replacing the control law by Eq. (4.39):

\[ \dot{V}(\bar{x}) = s(\ddot{x} + (-(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - \varepsilon(u) + u_{k-1} - \eta sgn(s)) - u_{k-1} + \varepsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \leq 0 \quad (4.44) \]
which can be simplified as:

\[ \dot{V}(\vec{x}) = s(-\eta \text{sgn}(s)) \leq 0 \tag{4.45} \]

Using the definition of the relay function, Eq. (3.7), the following equality holds true:

\[ \dot{V}(\vec{x}) = -\eta |s| \leq 0 \tag{4.46} \]

Since \( \eta \) can only assume positive values, the negative-definiteness of the Eq. (4.46) is guaranteed. Therefore, the closed-loop system is asymptotically stable and the control law form defined at Eq. (4.10) is correct since it satisfies the Lyapunov’s stability criteria.

### 4.3.2 Switching Gain

The control law defined by Eq. (4.39) is redefined as:

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda (\dot{x} - \dot{x}_d) - \hat{\epsilon}(u) + u_{k-1} - K \text{sgn}(s) \tag{4.47} \]

where \( \hat{\epsilon}(u) \) is the estimation of the control input error, described by Eq. (4.4), and \( K \) is the switching gain, which is required to ensure that the system trajectories are asymptotically stable during the reaching phase. Therefore, sliding condition must be satisfied:

\[ s\dot{s} \leq -\eta |s| \tag{4.48} \]

Performing the same procedures as before used to proof the controller form, but now with the updated control law defined by Eq. (4.47), the following equation is obtained:

\[ s(\epsilon(u) - \hat{\epsilon}(u) - K \text{sgn}(s)) \leq -\eta |s| \tag{4.49} \]

With the objective to the controller be the most conservative possible, i.e., be able to handle the most extreme case, the upper bound of the control input error is used:

\[ \epsilon(u) = (1 + \sigma_u)\hat{\epsilon}(u) \tag{4.50} \]

Replacing Eq. (4.50) into Eq. (4.49):

\[ s\left(\epsilon(u) - \hat{\epsilon}(u) + \sigma_u\hat{\epsilon}(u) - K \text{sgn}(s)\right) \leq -\eta |s| \tag{4.51} \]

which can be simplified to:

\[ -K |s| + s\sigma_u\hat{\epsilon}(u) \leq -\eta |s| \tag{4.52} \]

Re-arranging the equation above in terms of the switching gain:
\[ K|s| \geq \eta|s| + s\sigma_u \dot{\epsilon}(u) \]  

(4.53)

Again, to be most conservative possible, the absolute value of both sides of equation is used, which results into the following equation after dividing both sides by the absolute value of the sliding surface:

\[ K = |\sigma_u \dot{\epsilon}(u)| + \eta \]  

(4.54)

Replacing the estimation of the control input error by Eq. (4.4), the switching gain becomes:

\[ K = |\sigma_u(u_{k-2} - u_{k-1})| + \eta \]  

(4.55)

The control law can also be updated by replacing the estimation of the control input error by Eq. (4.4):

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - u_{k-2} + 2u_{k-1} - Ksgn(s) \]  

(4.56)

4.3.3 Boundary Layer

As shown in Sections 3.3.1 and 4.2.4, the sliding mode controller tends to produce a control signal that chatters, due to the high activity of the discontinuous term. In order to reduce the chattering, a smoothing boundary layer is inserted in the control law form, in place of the relay function. The procedure to add the boundary layer to the sliding mode controller can be seen with more details in Section 3.2. Hence, using Eq. (3.9), the control law becomes:

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - u_{k-2} + 2u_{k-1} - (K - \phi) sat \left( \frac{s}{\varphi} \right) \]  

(4.57)

Replacing the switching gain by Eq. (4.55), the control law is updated as:

\[ u = -(\ddot{x} - \ddot{x}_d) - \lambda(\dot{x} - \dot{x}_d) - u_{k-2} + 2u_{k-1} - (|\sigma_u(u_{k-2} - u_{k-1})| + \eta \\
- \phi) sat \left( \frac{s}{\varphi} \right) \]  

(4.58)

where the boundary layer dynamics are defined as:

\[ \dot{\varphi} = -\lambda \varphi + \sigma_u(u_{k-2} - u_{k-1}) + \eta \]  

(4.59)

with \( \varphi(0) = \eta/\lambda \).
4.3.4 Illustrative Examples

Two illustrative examples are presented next, one as a second-order linear system and the other one as a second-order nonlinear system. In both systems, the model-free sliding mode controller is implemented only with the smoothing boundary layer, defined by Eq. (4.57). Since the model-free sliding mode controller implemented with the relay function does not have acceptable tracking responses, it will be ignored.

4.3.4.1 Second-order linear system

Consider the following second-order linear mass-spring-damper model to be controlled:

\[ m\ddot{x} + c\dot{x} + kx = u \]  

(4.60)

where \( m \) is the mass of the system, \( c \) is the damping coefficient, \( k \) is the spring constant, \( u \) is the control input, \( \dot{x} \), \( \ddot{x} \) and \( x \) are the state measurement variables. For this example, the mass is set to 2 kg, the damping coefficient to 0.8 N/m/s and the spring constant to 2 N/m. The tracking problem is to track the reference signal defined as:

\[ x_d(t) = \sin\left(\frac{\pi}{2} t\right) \]  

(4.61)

Using the control law, Eq. (4.57), and the switching gain, Eq. (4.55), a Simulink model was built, as shown in Figures 4.45, 4.46, 4.47, 4.48, 4.49 and 4.50. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

Table 4.3: Controller parameters for a second-order linear system with unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>20</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Figure 4.45: Open-Loop system of the system described at Eq. (4.60)

Figure 4.46: Control law described at Eq. (4.57)

Figure 4.47: Desired Tracking block

Figure 4.48: Switching gain described at Eq. (4.55)
Figure 4.49: Sign or Sat. block

Figure 4.50: Boundary Layer block

Figure 4.51 shows that outstanding agreement is observed between the desired position and the measured position. The position tracking error, displayed by Figure 4.52, is less than $3 \times 10^{-7}$.

Figure 4.51: Position comparison for the linear second-order system example with boundary layer

Figure 4.52: Position tracking error for the linear second-order system example with boundary layer
Figure 4.53 shows that the control system is driving the velocity state onto the desired one with an outstanding performance. Figure 4.54 displays the velocity tracking error of the system and the error is less than 5e-07, which is negligible.

Figure 4.53: Velocity comparison for the linear second-order system example with boundary layer

Figure 4.54: Velocity tracking error for the linear second-order system example with boundary layer

Figure 4.55 shows that the agreement between the acceleration state and the reference signal is outstanding. Figure 4.56 displays the acceleration tracking error, which is less than 1e-04.

Figure 4.55: Acceleration comparison for the linear second-order system example with boundary layer

Figure 4.56: Acceleration tracking error for the linear second-order system example with boundary layer

Figure 4.57 displays the updated sliding condition of the closed-loop system. The sliding condition is satisfied at all times, since the sliding surface remains within the boundary layer. The control effort is smooth without any effect of chattering, as shown in Figure 4.58.
The model-free sliding mode controller achieved outstanding tracking performance for a second-order linear system. All system’s states are following the reference signal with minor errors. Since the controller was implemented with a smoothing boundary layer, the chattering was eliminated. The sliding condition was satisfied all simulation time, which proves that the closed-loop system is asymptotically stable.

### 4.3.4.2 Second-order nonlinear system

Consider the following second-order nonlinear mass-spring-damper model to be controlled:

\[
m\ddot{x} + c\dot{x} + kx^2 = u
\]  

(4.62)

where \(m\) is the mass of the system, \(c\) is the damping coefficient, \(k\) is the spring constant, \(u\) is the control input, \(\ddot{x}, \dot{x}\) and \(x\) are the state measurement variables. For this example, the mass is set to 2 kg, the damping coefficient to 0.8 N/m/s and the spring constant to 2 N/m. The tracking problem is to track the reference signal defined as:

\[
x_d(t) = \sin\left(\frac{\pi}{2} t\right)
\]

(4.63)

Using the control law, Eq. (4.57), and the switching gain, Eq. (4.55), a Simulink model was built, as shown in Figures 4.59, 4.60, 4.61, 4.62, 4.63 and 4.64. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:
Table 4.4: Controller parameters for a second-order nonlinear system with unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_u$</td>
<td>0.5</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>20</td>
</tr>
<tr>
<td>$\eta$</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Figure 4.59: Open-Loop system of the system described at Eq. (4.62)

Figure 4.60: Control law described at Eq. (4.57)
Figure 4.61: Desired Tracking block

Figure 4.62: Switching gain described at Eq. (4.55)

Figure 4.63: Sign or Sat. block

Figure 4.64: Boundary Layer block
Figure 4.65 shows that the agreement between the state position and the reference signal is outstanding. The error between the state position and the reference signal is less than $5e^{-07}$, as displayed by Figure 4.66.

Figure 4.65: Position comparison for the nonlinear second-order system example with boundary layer

Figure 4.66: Position tracking error for the nonlinear second-order system example with boundary layer

Figure 4.67 shows that near perfect tracking is achieved for the velocity state. Figure 4.68 displays the velocity tracking error of the system, which is less than $1e^{-06}$.

Figure 4.67: Velocity comparison for the nonlinear second-order system example with boundary layer

Figure 4.68: Velocity tracking error for the nonlinear second-order system example with boundary layer

Figure 4.69 displays the comparison between the acceleration state trajectory and the desired trajectory. The difference between the trajectories is negligible, with error less than $2e^{-04}$, as displayed by Figure 4.70.

Figure 4.69: Acceleration comparison for the nonlinear second-order system example with boundary layer

Figure 4.70: Acceleration tracking error for the nonlinear second-order system example with boundary layer
Figure 4.69: Acceleration comparison for the nonlinear second-order system example with boundary layer

Figure 4.70: Acceleration tracking error for the nonlinear second-order system example with boundary layer

Figure 4.71: Sliding condition for the nonlinear second-order system example with boundary layer

Figure 4.72: Control effort for the nonlinear second-order system example with boundary layer

The model-free sliding mode controller achieved outstanding tracking performance for a second-order nonlinear system. The same controller for the linear system was used, which proves that the controller is a model-free scheme, since the control law was not required to be updated or modified. The closed-loop system is asymptotically stable, since the sliding surface was satisfied during the
simulation times. The control effort has no chattering, once the controller was implemented with the smoothing boundary layer and thus is feasible to be implemented in a real-world system.
5 Model-Free SMC with non-unitary Control Input Gain

In this chapter, a model-free sliding mode controller is derived for systems with non-unitary control input gain. The model-free SMC method presented previously cannot be used if the control input gain is non-unitary, since the overall performance of the controller would be unsatisfactory. Therefore, a new scheme is required to handle system type with non-zero input influence.

Since the controller scheme is assumed to be model-free, the control input gain is defined as an unknown variable and can be time-varying or state-dependent. However, it is assumed the bounds of the control input gain are known, which is a realistic assumption.

The chapter is outlined as follows. Section 5.1 describes the system with a non-unitary control input gain used to derive a model-free sliding mode controller. Section 5.2 derives a model-free sliding mode controller for a first-order system. In Section 5.3, a model-free sliding mode controller is derived for a second-order system. In both last sections, linear and nonlinear examples are illustrated and results are discussed.

5.1. System Description

Consider a $n^{th}$-order single-input autonomous system. The following equality holds true for the system:

$$x^n = x^n + bu - bu_{k-1} - bu + bu_{k-1}$$

(5.1)

where $x^n$ represents the higher order state, $u$ is the control input, $u_{k-1}$ is the previous value of the control input and $b$ is the control input gain. The error between the control input and the previous control input is defined as:

$$
\epsilon(u) = u_{k-1} - u
$$

(5.2)

Hence, the system can be rewritten as:

$$x^n = x^n + bu - bu_{k-1} + b\epsilon(u)$$

(5.3)

The control input gain $b$ is considered to be unknown, but with known bounds, as defined in the equation below:

$$b_{low} \leq b \leq b_{up}$$

(5.4)
where $b_l$ is the lower bound of the control input gain and $b_u$ is the upper bound of the control input gain. In order to compute the control law, and to avoid an algebraic loop within the controller algorithm, an estimation of the control input error is necessary. The estimation of the control input error is defined as:

$$\hat{\varepsilon}(u) = u_{k-1} - u_{k-2}$$

(5.5)

where $u_{k-2}$ is the previous control input of the previous control input. Although the control input error is not known exactly, the error is assumed to be bounded as follows:

$$(1 - \sigma_l)\varepsilon(u) \leq \varepsilon(u) \leq (1 + \sigma_u)\hat{\varepsilon}(u)$$

(5.6)

where $\sigma_u$ is the upper bound and $\sigma_l$ is the lower bound of the control input estimation error. If the sampling time is high enough, the values of the error’s bounds will be near zero since the estimation error will be approximately equal to the actual error. Since the next steps to derive a model-free sliding mode controller scheme for systems with non-unitary control input gain depends on the order of the system, the derivation is divided in two sections. In the following section, the derivation for a first-order system is presented. Next, a model-free sliding mode controller is derived for a second-order system.

### 5.2. First-Order System

For a first-order system, Slotine and Li [15] defines the sliding surface as:

$$s = x - x_d + \lambda \int_0^t (x - x_d)dr$$

(5.7)

which takes into account $\lambda$, differently from Eq. (3.4). In order to obtain the control law, the sliding surface is differentiated with respect to time and the resulting equation is set to be equal to zero. The procedure is necessary in order for the system’s state trajectories to remain on the sliding surface once they reach it. Hence:

$$\dot{s} = \dot{x} - \dot{x}_d + \lambda (x - x_d) = 0$$

(5.8)

Replacing Eq. (5.1) into Eq. (5.8), the following is obtained:

$$\dot{x} + bu - bu_{k-1} + b\hat{\varepsilon}(u) - \dot{x}_d + \lambda (x - x_d) = 0$$

(5.9)

Re-arranging the equation above in terms of the control input $u$:
\[ u = -b^{-1}[\dot{x} - \dot{x}_d + \lambda(x - x_d)] + u_{k-1} - \varepsilon(u) \]  
\[ (5.10) \]

In order for the controller to achieve robustness against system uncertainties, a discontinuous term added in the control law, as shown in the previous chapters. The updated control law with the discontinuous term included is described below:

\[ u = b^{-1}[-(\dot{x} - \dot{x}_d) - \lambda(x - x_d) - \eta\text{sgn}(s)] + u_{k-1} - \varepsilon(u) \]  
\[ (5.11) \]

where \( \eta \) is a small positive constant and \( sgn(s) \) is the relay function of the sliding surface.

### 5.2.1 Proof of the Controller Form

To ensure that the closed-loop system’s trajectories are asymptotically stable during the reaching phase, Lyapunov’s direct method is applied. Thus, we define an equation describing the system’s energy, which is defined as:

\[ V(\vec{x}) = \frac{1}{2} s^2 \]  
\[ (5.12) \]

Clearly the function shown above is positive definite, which means that the system has initial positive energy. To obtain the energy rate of the system, Eq. (5.12) is differentiated with respect to time, which results into:

\[ \dot{V}(\vec{x}) = \frac{dV(\vec{x})}{d(\vec{x})} \frac{d\vec{x}}{dt} = \dot{s}s \leq 0 \]  
\[ (5.13) \]

Replacing Eq. (5.8) into the derivative of the Lyapunov’s candidate function results in:

\[ \dot{V}(\vec{x}) = s(\dot{x} - \dot{x}_d + \lambda(x - x_d)) \leq 0 \]  
\[ (5.14) \]

which can be further modified replacing Eq. (5.1) into it:

\[ \dot{V}(\vec{x}) = s(\dot{x} + bu - bu_{k-1} + b\varepsilon(u) - \dot{x}_d + \lambda(x - x_d)) \leq 0 \]  
\[ (5.15) \]

Replacing the control law defined by Eq. (5.11), the following is obtained:

\[ \dot{V}(\vec{x}) = s(\dot{x} + bu - bu_{k-1} + b\varepsilon(u) - \eta\text{sgn}(s) - bu_{k-1} + b\varepsilon(u) - \dot{x}_d + \lambda(x - x_d) - \lambda(x - x_d)) \leq 0 \]  
\[ (5.16) \]

which can be simplified as:

\[ \dot{V}(\vec{x}) = s(-\eta\text{sgn}(s)) \leq 0 \]  
\[ (5.17) \]
Since the relay function assumes a unitary negative value when the sliding surface is negative and a unitary positive value when the sliding surface is positive, the negative definiteness of Eq. (5.17) is assured, i.e.:

\[-\eta |s| \leq 0 \quad (5.18)\]

which is negative definite as \(\eta\) can only assume positive values. Therefore, the closed-loop system is asymptotically stable and the control law form defined in Eq. (5.11) is proved to be correct, as the Lyapunov’s criteria was satisfied.

### 5.2.2 Switching Gain

The control law, defined at Eq. (5.11), is updated as:

\[ u = \hat{b}^{-1}[-(\dot{x} - \dot{x_d}) - \lambda(x - x_d) - K \text{sgn}(s)] + u_{k-1} - \hat{\varepsilon}(u) \quad (5.19) \]

where \(\hat{\varepsilon}(u)\) is the estimation of the control input error, described in Eq. (4.4), \(K\) is the switching gain required to ensure that the state trajectories are asymptotically stable during the reaching phase and \(\hat{b}\) is the estimation of the control input gain. The control input gain is estimated by:

\[ \hat{b} = \sqrt{b_{up}b_{low}} \quad (5.20) \]

In order to simplify several equations that will be used later, an auxiliary variable, \(\beta\), is defined:

\[ \beta = b\hat{b}^{-1} = \frac{b_{up}}{\sqrt{b_{low}}} \quad (5.21) \]

To the closed-loop system be asymptotically stable, the sliding condition defined in Eq. (3.5) must be satisfied, i.e.:

\[ s\hat{s} \leq -\eta |s| \quad (5.22) \]

By performing the same procedures as before to proof the controller form, but using the updated control law defined in Eq. (5.19), the following is obtained:

\[ s \left( (\ddot{x} - \ddot{x_d})(1 - b\hat{b}^{-1}) + \lambda(x - x_d)(1 - b\hat{b}^{-1}) - b\hat{b}^{-1}K\text{sgn}(s) + b(\varepsilon(u) - \hat{\varepsilon}(u)) \right) \leq -\eta |s| \quad (5.23) \]

To be most conservative possible, the upper bound of the control input error is used:
\[ \varepsilon(u) = (1 + \sigma_u)\ddot{\varepsilon}(u) \] (5.24)

Replacing into Eq. (5.23):
\[ s \left( (\dot{x} - \dot{x}_d)(1 - b\hat{b}^{-1}) + \lambda(x - x_d)(1 - b\hat{b}^{-1}) - b\hat{b}^{-1}K\text{sgn}(s) + b\sigma_u\dot{\varepsilon}(u) \right) \leq -\eta|s| \] (5.25)

The equation described above can be re-arranged in terms of the switching gain \( K \), as shown below:
\[ s \left( (\dot{x} - \dot{x}_d)(1 - b\hat{b}^{-1}) + \lambda(x - x_d)(1 - b\hat{b}^{-1}) + b\sigma_u\dot{\varepsilon}(u) \right) + \eta|s| \leq b\hat{b}^{-1}K|s| \] (5.26)

Dividing both sides by \( b\hat{b}^{-1} \):
\[ s \left( (\dot{x} - \dot{x}_d)(\hat{b}b^{-1} - \hat{b}b^{-1}b\hat{b}^{-1}) + \lambda(x - x_d)(\hat{b}b^{-1} - \hat{b}b^{-1}b\hat{b}^{-1}) + \hat{b}b^{-1}b\sigma_u\dot{\varepsilon}(u) \right) + \hat{b}b^{-1}\eta|s| \leq K|s| \] (5.27)

which can be simplified as:
\[ s \left( (\dot{x} - \dot{x}_d)(\hat{b}b^{-1} - 1) + \lambda(x - x_d)(\hat{b}b^{-1} - 1) + \hat{b}\sigma_u\dot{\varepsilon}(u) \right) + \hat{b}b^{-1}\eta|s| \leq K|s| \] (5.28)

Since the equality of Eq. (5.28) states that the switching gain \( K \) must be equal or greater to the left side of the equation, the absolute value is applied in both sides in order to the controller be able to handle the most extreme case. Thus, after dividing both sides by the absolute value of the sliding surface, \(|s|\), the following result is obtained:
\[ K = |\dot{x} - \dot{x}_d| |\hat{b}b^{-1} - 1| + \lambda|x - x_d| |\hat{b}b^{-1} - 1| + |\hat{b}\sigma_u\dot{\varepsilon}(u)| + \hat{b}b^{-1}\eta \] (5.29)

The switching gain can be updated as shown below, by replacing \( \hat{b}b^{-1} \) by Eq. (5.21) and the estimation of the control input gain by Eq. (4.4):
\[ K = |\dot{x} - \dot{x}_d| |\beta - 1| + \lambda|x - x_d| |\beta - 1| + |\hat{b}\sigma_u(u_{k-2} - u_{k-1})| + \beta\eta \] (5.30)

If the control input again is set to be unitary, the switching gain becomes the identical switching gain derived for a first-order system with unitary control input gain, defined by Eq. (4.26). The control law can be also updated using Eq. (4.4):
\[ u = \hat{b}^{-1}[-(\dot{x} - \dot{x}_d) - \lambda(x - x_d) - Ks\text{gn}(s)] + 2u_{k-1} - u_{k-2} \] (5.31)
5.2.3 Boundary Layer

As shown previously, the sliding mode controller introduces a high frequency signal to the closed-loop system, known as chattering, due to the discontinuous term added in the control law. In order to reduce the chattering effect, a smoothing boundary layer is included in the control law form in place of the relay function. The procedure to add the boundary layer to the control law can be seen with more details in Section 3.2. Thus, the control law becomes:

\[
u = \hat{b}^{-1} \left[ -(\dot{x} - \dot{x}_d) - \lambda (x - x_d) - K - \phi \right] sat \left( \frac{s}{\phi} \right) + 2u_{k-1} - u_{k-2}
\] (5.32)

Replacing the switching gain by Eq. (5.30) into the equation above:

\[
u = \hat{b}^{-1} \left[ -(\dot{x} - \dot{x}_d) - \lambda (x - x_d) - (|\dot{x} - \dot{x}_d|/1 - |x - x_d|/1) + \lambda |x - x_d|/1 - \right]
\]
\[+ |\hat{b} \sigma_u (u_{k-2} - u_{k-1})| + \beta/|\dot{x} - \dot{x}_d|/1 - \] sat \left( \frac{s}{\phi} \right) + 2u_{k-1} - u_{k-2}
\] (5.33)

where the boundary layer dynamics are defined as:

\[
\dot{\phi} = -\lambda \phi + |\dot{x} - \dot{x}_d|/1 - + \lambda |x - x_d|/1 - + |\hat{b} \sigma_u (u_{k-2} - u_{k-1})| + \beta/|\dot{x} - \dot{x}_d|/1 - \eta
\] (5.34)

with \(\phi(0) = \eta/\lambda\).

5.2.4 Illustrative Examples

Two illustrative examples are presented next, one as a linear first-order system and the other one as a nonlinear first-order system. In both examples, the model-free sliding mode controller is only implemented with a smoothing boundary layer.

5.2.4.1 First-order linear system

Suppose the following first-order linear model with non-unitary control input gain:

\[
\dot{x} + 5x = bu
\] (5.35)

where \(u\) is the control input, \(b\) is the control input gain, and, \(\dot{x}\) and \(x\) are the state measurement variables of the system. The tracking problem is to track the reference signal defined as:

\[
x_d(t) = \sin \left( \frac{\pi}{2} t \right)
\] (5.36)
The control input gain is set to be time-varying, and it is bounded between 1 and 5, as shown below:

![Figure 5.1: Control input gain variation over time](image)

Using the control law, Eq. (5.33), and the switching gain, Eq. (5.30), a Simulink model was built as shown in Figures 5.2, 5.3, 5.4, 5.5, 5.6, 5.7 and 5.8. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_u$</td>
<td>0.5</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>20</td>
</tr>
<tr>
<td>$\eta$</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Figure 5.2: Open-Loop system of the model described at Eq. (5.35)

Figure 5.3: Control law described at Eq. (5.33)

Figure 5.4: Desired Tracking block
Figure 5.5: Switching gain described at Eq. (5.30)

Figure 5.6: Sign or Sat. block

Figure 5.7: Boundary Layer block

Figure 5.8: Control input gain block
Figure 5.9 shows that the agreement between the position measurement and the position reference is outstanding. Figure 5.10 displays the position tracking error of the closed-loop system, which is less than 1.5e-06.

![Position Comparison](image1)

**Figure 5.9:** Position comparison for the first-order linear example with non-unitary control input gain

![Position tracking error](image2)

**Figure 5.10:** Position tracking error for the first-order linear example with non-unitary control input gain

Figure 5.11 shows that near perfect tracking is achieved for the velocity state. The error is less than 2.5e-04, as displayed by Figure 5.12.

![Velocity Comparison](image3)

**Figure 5.11:** Velocity comparison for the first-order linear example with non-unitary control input gain

![Velocity tracking error](image4)

**Figure 5.12:** Velocity tracking error for the first-order linear example with non-unitary control input gain

Figure 5.13 displays the sliding condition of the closed-loop system. The sliding surface remains inside the boundary layer all simulation time, which implies that the sliding condition is satisfied. The control effort, displayed by Figure 5.14, is smooth since the control effort chattering was eliminated due the utilization of the smoothing boundary layer.

![Sliding condition](image5)

**Figure 5.13:** Sliding condition of the closed-loop system

![Control effort](image6)

**Figure 5.14:** Control effort with non-unitary control input gain
The model-free sliding mode controller achieved excellent tracking responses for linear systems with non-unitary control input gain. The difference between the state’s trajectories and the reference signal is minimal. The control effort chattering was eliminated and the sliding condition was satisfied, which implies that the closed-loop system is asymptotically stable.

### 5.2.4.2 First-order nonlinear system

Suppose the following first-order nonlinear model with non-unitary control input gain:

\[ \dot{x} + 5x^2 = bu \]  

where \( u \) is the control input, \( b \) is the control input gain, and \( \dot{x} \) and \( x \) are the state measurement variables of the system. The tracking problem is to track the reference signal defined as:

\[ x_d(t) = \sin \left( \frac{\pi}{2} t \right) \]  

The identical time-varying control input gain, used in the previous example and shown in Figure 5.1, is used in the nonlinear example. Using the control law defined by Eq. (5.33) and the switching gain defined by Eq. (5.30), a Simulink model was built as shown in Figures 5.15, 5.16, 5.17, 5.18, 5.19, 5.20 and 5.21. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:
Table 5.2: Controller parameters for a first-order nonlinear system with non-unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_u$</td>
<td>0.5</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>20</td>
</tr>
<tr>
<td>$\eta$</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Figure 5.15: Open-Loop system of the model described at Eq. (5.38)

Figure 5.16: Control law described at Eq. (5.33)

Figure 5.17: Desired Tracking block
Figure 5.18: Switching gain described at Eq. (5.30)

Figure 5.19: Sign or Sat. block

Figure 5.20: Boundary Layer block

Figure 5.21: Control input gain block
Figure 5.22 shows that the difference between the position trajectory and the reference signal is minimal. The position tracking error, displayed by Figure 5.23, is less than 2e-06.

Figure 5.22: Position comparison for the first-order nonlinear example with non-unitary control input gain

Figure 5.23: Position tracking error for the first-order nonlinear example with non-unitary control input gain

Figure 5.24 displays the velocity trajectory comparison between the velocity state and the reference signal. Near perfect tracking is obtained, with errors less than 3.5e-04, as displayed by Figure 5.25.

Figure 5.24: Velocity comparison for the first-order nonlinear example with non-unitary control input gain

Figure 5.25: Velocity tracking error for the first-order nonlinear example with non-unitary control input gain

Figure 5.26 displays the sliding condition of the closed-loop system. The sliding surface remains inside the boundary layer all simulation time. Thus, the sliding condition is satisfied. Figure 5.27 displays the control effort, which is smooth without any chattering.

Figure 5.26: Sliding condition for the first-order nonlinear example with non-unitary control input gain

Figure 5.27: Control effort for the first-order nonlinear example with non-unitary control input gain
The model-free sliding mode controller presented excellent performance for a first-order nonlinear system with non-unitary control input. The tracking performance is outstanding, since the system’s states are tracking the reference signal smoothly with minor errors. The same controller was used for the linear and nonlinear examples, which proves that this controller only relies in the previous control inputs, state measurements and system’s order knowledge. As the sliding condition was satisfied, the closed-loop system is asymptotically stable.

### 5.3. Second-Order System

For a second-order system, the following sliding surface is obtained using Eq. (3.4):

\[ s = \dot{x} - \dot{x}_d + \lambda(x - x_d) \quad (5.40) \]

The control law is derived by differentiating the sliding surface and setting the resulting equation to be equal to zero. In that manner, the system’s trajectories remain inside the sliding surface once they reach it. Therefore:

\[ \dot{s} = \ddot{x} - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d) = 0 \quad (5.41) \]

Replacing Eq. (5.1) into the derivative of the sliding surface:

\[ \dot{s} = \ddot{x} + bu - bu_{k-1} + b\epsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d) = 0 \quad (5.42) \]

Re-arranging the equation above in terms of the control input \( u \):

\[ bu = -\ddot{x} + bu_{k-1} - b\epsilon(u) + \ddot{x}_d - \lambda(\dot{x} - \dot{x}_d) \quad (5.43) \]
After dividing both sides of the equation by the control input gain, the following control law is obtained:

\[ u = b^{-1}[-\lambda(\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d)] + u_{k-1} - \varepsilon(u) \]  

(5.44)

A discontinuous term is added to the control law in order to the control system be able to drive the system’s trajectories onto the sliding surface in the presence of uncertainties:

\[ u = b^{-1}[-\lambda(\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d) - \eta \text{sgn}(s)] + u_{k-1} - \varepsilon(u) \]  

(5.45)

where \( \eta \) is a small positive constant and \( \text{sgn}(s) \) is the relay function of the sliding surface.

### 5.3.1 Proof of the Controller Form

In order to assure that the closed-loop system trajectories are asymptotically stable during the reaching phase, the Lyapunov’s direct method is used. The following equation is used as the Lyapunov function:

\[ V(\vec{x}) = \frac{1}{2} s^2 \]  

(5.46)

The equation defined above represents the system’s energy. Since it is positive definite, the system has initially positive energy. The system’s energy rate is derived by differentiating Eq. (5.46) with respect to time:

\[ \dot{V}(\vec{x}) = \frac{dV(\vec{x})}{d\vec{x}} \cdot \frac{d\vec{x}}{dt} = \dot{s}s \leq 0 \]  

(5.47)

Replacing Eq. (5.41) into the derivative of the Lyapunov function:

\[ \dot{V}(\vec{x}) = s(\ddot{x} - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \leq 0 \]  

(5.48)

which can be further manipulated by using Eq. (5.1):

\[ \dot{V}(\vec{x}) = s(\ddot{x} + bu - bu_{k-1} + b\varepsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \]  

(5.49)

Finally, replacing the control law by Eq. (5.45):

\[ \dot{V}(\vec{x}) = s(\ddot{x} + (-bb^{-1}(\ddot{x} - \ddot{x}_d) - bb^{-1}\lambda(\dot{x} - \dot{x}_d) - b\varepsilon(u) + bu_{k-1} - bb^{-1}\eta \text{sgn}(s)) - bu_{k-1} + b\varepsilon(u) - \ddot{x}_d + \lambda(\dot{x} - \dot{x}_d)) \leq 0 \]  

(5.50)

which can be simplified into:
\[
\dot{V}(\bar{x}) = s(-\eta \text{sgn}(s)) \leq 0 \tag{5.51}
\]

Using the definition of the relay function described by Eq. (3.7), the following equation holds true for the derivative of the Lyapunov function:

\[
\dot{V}(\bar{x}) = -\eta |s| \leq 0 \tag{5.52}
\]

Since \(\eta\) can only assume positive values, the equality described at Eq. (5.52) is satisfied. Thus, the closed-loop system is asymptotically stable and the control law defined at Eq. (5.45) is proved to be correctly derived, since the Lyapunov’s stability criteria was satisfied.

### 5.3.2 Switching Gain

The control law defined in Eq. (5.45) is updated as:

\[
u = \hat{b}^{-1}[-\lambda(\ddot{x} - \ddot{x}_d) - (\dot{x} - \dot{x}_d) - K \text{sgn}(s)] + u_{k-1} - \hat{\epsilon}(u) \tag{5.53}
\]

where \(\hat{\epsilon}(u)\) is the estimation of the control input error, defined in Eq. (4.4), \(\hat{b}\) is the estimation of the input matrix, defined in Eq. (5.20), and \(K\) is the switching gain to be found. The switching gain is required to ensure that the state trajectories are asymptotically stable during the reaching phase. The sliding condition must be satisfied:

\[
s\dot{s} \leq \eta |s| \tag{5.54}
\]

Performing the same procedures used to proof the controller form, but now with the updated control law defined by Eq. (5.53), the following equation is obtained:

\[
s\big[(\ddot{x} - \ddot{x}_d) - b\hat{b}^{-1}(\ddot{x} - \ddot{x}) + \lambda(\dot{x} - \dot{x}_d) - b\hat{b}^{-1}\lambda(\dot{x} - \dot{x}_d) - b\hat{\epsilon}(u) + b\epsilon(u)
\]

\[
- b\hat{b}^{-1}K \text{sgn}(s)\big] \leq -\eta |s| \tag{5.55}
\]

To be most conservative possible, the upper bound of the control input error is used:

\[
\epsilon(u) = (1 + \sigma_u)\hat{\epsilon}(u) \tag{5.56}
\]

Replacing Eq. (5.56) into Eq. (5.55):

\[
s\big[(\ddot{x} - \ddot{x}_d) - b\hat{b}^{-1}(\ddot{x} - \ddot{x}_d) + \lambda(\dot{x} - \dot{x}_d) - b\hat{b}^{-1}\lambda(\dot{x} - \dot{x}_d) + b\sigma_u\hat{\epsilon}(u)
\]

\[
- b\hat{b}^{-1}K \text{sgn}(s)\big] \leq -\eta |s| \tag{5.57}
\]

which can be re-arranged as:
\[ s[(\ddot{x} - \ddot{x}_d)(1 - b\hat{b}^{-1}) + \lambda(\dot{x} - \dot{x}_d)(1 - b\hat{b}^{-1}) + b\sigma_u\dot{\varepsilon}(u) - b\hat{b}^{-1}K\text{sgn}(s)] \leq -\eta|s| \] (5.58)

Writing in terms of the switching gain \( K \):

\[ b\hat{b}^{-1}K|s| \geq s[(\ddot{x} - \ddot{x}_d)(1 - b\hat{b}^{-1}) + \lambda(\dot{x} - \dot{x}_d)(1 - b\hat{b}^{-1}) + b\sigma_u\dot{\varepsilon}(u)] + \eta|s| \] (5.59)

Dividing both sides by \( b\hat{b}^{-1} \):

\[ K|s| \geq s[(\ddot{x} - \ddot{x}_d)(\hat{b}b^{-1} - \hat{b}b^{-1}b\hat{b}^{-1}) + \lambda(\dot{x} - \dot{x}_d)(\hat{b}b^{-1} - \hat{b}b^{-1}b\hat{b}^{-1}) + \hat{b}b^{-1}b\sigma_u\dot{\varepsilon}(u)] + \hat{b}b^{-1}\eta|s| \] (5.60)

Finally, after some simplifications:

\[ K|s| \geq s[(\ddot{x} - \ddot{x}_d)(\hat{b}b^{-1} - 1) + \lambda(\dot{x} - \dot{x}_d)(\hat{b}b^{-1} - 1) + \hat{b}\sigma_u\dot{\varepsilon}(u)] + \hat{b}b^{-1}\eta|s| \] (5.61)

As the equality of the equation defined above states greater or equal, the absolute value is used in both sides of the equation in order to the controller be conservative as possible, which enables the controller to handle the most extreme case. Thus, after dividing both sides of the equation by the absolute value of the sliding surface, the following result is obtained:

\[ K = |\hat{b}b^{-1} - 1||\ddot{x} - \ddot{x}_d| + |\hat{b}b^{-1} - 1|\lambda|\dot{x} - \dot{x}_d| + |\hat{b}\sigma_u\dot{\varepsilon}(u)| + \hat{b}b^{-1}\eta \] (5.62)

The equation above can be simplified by replacing Eq. (5.21) and Eq. (4.4) into it:

\[ K = |\beta - 1||\ddot{x} - \ddot{x}_d| + |\beta - 1|\lambda|\dot{x} - \dot{x}_d| + |\hat{b}\sigma_u(u_{k-2} - u_{k-1})| + \beta\eta \] (5.63)

If the control input gain is set to be unitary, the switching gain becomes identically the same as the one defined by Eq. (4.55), which is the switching gain derived for a second-order system with unitary control input gain. The control law can also be updated by replacing the estimation of the control input error by Eq. (4.4):

\[ u = \hat{b}^{-1}[-\lambda(\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d) - K\text{sgn}(s)] + 2u_{k-1} - u_{k-2} \] (5.64)

### 5.3.3 Boundary Layer

As shown in Chapters 3 and 4, when the relay function is used as discontinuous term in the control law form of the sliding mode controller, the control effort tends to chatter, which is unacceptable. To reduce, or remove, the chattering, a smoothing boundary layer is added to the control law. The
procedure to add the smoothing boundary layer is shown with more details in Section 3.2. Thus, the control law with boundary layer is updated as:

\[
u = \hat{b}^{-1} \left[ -\lambda (\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d) - (K - \dot{\phi}) \text{sat} \left( \frac{S}{\phi} \right) \right] + 2u_{k-1} - u_{k-2} \tag{5.65} \]

Replacing the switching gain by Eq. (5.63):

\[
u = \hat{b}^{-1} \left[ -\lambda (\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d) - (|\beta - 1||\dddot{x} - \dddot{x}_d| + |\beta - 1|\lambda|\dddot{x} - \dddot{x}_d| + |\hat{b}\sigma_u (u_{k-2} - u_{k-1})| + \beta \eta \right] \tag{5.66} \]

\[
- \phi \text{sat} \left( \frac{S}{\phi} \right) \right] + 2u_{k-1} - u_{k-2}
\]

where the boundary layer dynamics are defined as:

\[
\dot{\phi} = -\lambda \phi + |\beta - 1||\dddot{x} - \dddot{x}_d| + |\beta - 1|\lambda|\dddot{x} - \dddot{x}_d| + |\hat{b}\sigma_u (u_{k-2} - u_{k-1})| + \beta \eta \tag{5.67} \]

with \( \phi(0) = \eta/\lambda \).

5.3.4 Illustrative Examples

To test the model-free sliding mode controller, two illustrative examples are presented. One as a linear and the other one as a nonlinear system, both second-order systems with non-unitary control input gain. The model-free sliding mode controller is only implemented with a smoothing boundary layer.

5.3.4.1 Second-order linear system

Suppose that the following second-order linear mass-spring-damper model with non-unitary control input gain is to be controlled:

\[m\ddot{x} + c\dot{x} + kx = bu \tag{5.68} \]

where \(m\) is the mass of the system, \(c\) is the damping coefficient, \(k\) is the spring constant, \(u\) is the control input, \(b\) is the control input gain, \(\ddot{x}\), \(\dot{x}\) and \(x\) are the state measurement variables. For this example, the mass is set to 2 kg, the damping coefficient to 0.8 N/m/s and the spring constant to 2 N/m. The tracking problem is to track the reference signal defined as:
\[
x_d(t) = \sin\left(\frac{\pi}{2} t\right)
\]  

(5.69)

The control input gain is considered time-varying, bounded between 1 and 5, as shown below:

Figure 5.28: Control input gain variation over time

Therefore, the variables regarding the control input gain, required to compute the control law and the switching gain, are defined as:

\[
\hat{b} = \sqrt{b_ub_l} = \sqrt{5(1)} = \sqrt{5} \\
\beta = \frac{b_u}{b_l} = \sqrt{\frac{5}{1}} = \sqrt{5}
\]  

(5.70)

Using the control law, Eq. (5.66), and the switching gain, Eq. (5.63), a Simulink model was built, as shown in Figures 5.29, 5.30, 5.31, 5.32, 5.34 and 5.35. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

Table 5.3: Controller parameters for a second-order linear system with non-unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\sigma_u)</td>
<td>0.5</td>
</tr>
<tr>
<td>(\lambda)</td>
<td>20</td>
</tr>
<tr>
<td>(\eta)</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Figure 5.29: Open-Loop system of the model described at Eq. (5.68)

Figure 5.30: Control law described at Eq. (5.66)

Figure 5.31: Desired tracking block
Figure 5.32: Switching gain described at Eq. (5.63)

Figure 5.33: Control input gain block

Figure 5.34: Sign or Sat. block

Figure 5.35: Boundary Layer block
Figure 5.36 shows that the position state’s trajectory tracks the desired position measurement almost perfectly. Figure 5.37 displays the position tracking error of the closed-loop system, which is less than 2.5e-07.

Figure 5.38 shows that the control system is driving the velocity state’s trajectory into the desired trajectory with outstanding performance. Figure 5.39 displays the velocity tracking error of the closed-loop system and the error is less than 4.5e-07, which is negligible.

Figure 5.40 shows the comparison between the acceleration state’s trajectory and the reference signal. The difference is minimal, with errors less than 1e-04, as displayed by Figure 5.41.
Figure 5.40: Acceleration comparison for the second-order linear example with non-unitary control input gain

Figure 5.41: Acceleration tracking error for the second-order linear example with non-unitary control input gain

Figure 5.42 displays the sliding condition of the closed-loop system. The sliding condition is satisfied all simulation time, since the sliding surface remains inside the boundary layer. The control effort is smooth without any effect of chattering, as displayed by Figure 5.43.

Figure 5.42: Sliding condition for the second-order linear example with non-unitary control input gain

Figure 5.43: Control effort for the second-order linear example with non-unitary control input gain

The model-free sliding mode controller obtained outstanding response when applied to a second-order linear system with non-unitary control input gain. The system’s states are tracking the reference signal with negligible errors. The control effort is smooth, since the chattering was eliminated. The closed-loop system is asymptotically stable, as the sliding conditions was satisfied all simulation time.


5.3.4.2 Second-order nonlinear system

Suppose the following second-order nonlinear mass-spring-damper model with non-unitary control input gain is to be controlled:

\[ m \ddot{x} + c \dot{x} + kx^2 = bu \]  

(5.71)

where \( m \) is the mass of the system, \( c \) is the damping coefficient, \( k \) is the spring constant, \( u \) is the control input, \( b \) is the control input gain, \( \ddot{x}, \dot{x} \) and \( x \) are the state measurement variables. For this example, the mass is set to 2 kg, the damping coefficient to 0.8 N/m/s and the spring constant to 2 N/m. The tracking problem is to track the reference signal defined as:

\[ x_d(t) = \sin\left(\frac{\pi}{2} t\right) \]  

(5.72)

The identical time-varying control input gain, used in the previous linear example and shown in Figure 5.28, is used next. Using the control law, Eq. (5.66), and the switching gain, Eq. (5.63), a Simulink model was built, as shown in Figures 5.44, 5.45, 5.46, 5.47, 5.48, 5.49 and 5.50. A sampling time of 0.0001 seconds with ode5 (Dormand-Prince) as solver was implemented for 30 seconds. The controller parameters are defined as follows:

Table 5.4: Controller parameters for a second-order nonlinear system with non-unitary control input gain

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>20</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Figure 5.44: Open-Loop system of the model described at Eq. (5.71)

Figure 5.45: Control law described at Eq. (5.66)

Figure 5.46: Desired Tracking block
Figure 5.47: Switching gain described at Eq. (5.63)

Figure 5.48: Sign or Sat. block

Figure 5.49: Boundary Layer block

Figure 5.50: Control input gain block
Figure 5.51 displays the comparison between the position state’s trajectory and the reference signal. The agreement between the trajectories is excellent. Figure 5.52 displays the position tracking error the closed-loop system, which is less than 2.5e-07.

Figure 5.51: Position comparison for the second-order nonlinear example with non-unitary control input gain  
Figure 5.52: Position tracking error for the second-order nonlinear example with non-unitary control input gain

Figure 5.53 shows that near perfect tracking is achieved for the velocity state. The error is minimal, with values less than, as displayed by Figure 5.54.

Figure 5.53: Velocity comparison for the second-order nonlinear example with non-unitary control input gain  
Figure 5.54: Velocity tracking error for the second-order nonlinear example with non-unitary control input gain

Figure 5.55 shows that the acceleration state’s trajectory is tracking the reference signal with outstanding performance. The error between the acceleration state’s trajectory and the desired one is less than 1e-04, as shown in Figure 5.56.

Figure 5.55: Acceleration comparison for the second-order nonlinear example with non-unitary control input gain  
Figure 5.56: Acceleration tracking error for the second-order nonlinear example with non-unitary control input gain
Figure 5.55: Acceleration comparison for the second-order nonlinear example with non-unitary control input gain

Figure 5.56: Acceleration tracking error for the second-order nonlinear example with non-unitary control input gain

Figure 5.57 displays the updated sliding condition of the closed-loop system. The sliding surface remains within the boundary layer all simulation time. Therefore, the sliding condition is satisfied. The control effort, shown in Figure 5.58, has no chattering and the response is smooth.

Figure 5.57: Sliding condition for the second-order nonlinear example with non-unitary control input gain

Figure 5.58: Control effort for the second-order nonlinear example with non-unitary control input gain

When implemented to second-order nonlinear systems with non-unitary control input gain, the model-free sliding mode controller obtained outstanding tracking performance. The tracking response is excellent, where nearly perfect tracking was obtained for all system’s states. The control effort has no chattering, due the utilization of the smoothing boundary layer. The closed-loop system is asymptotically stable, since the sliding condition was satisfied all simulation time.
The model-free characteristic of the controller is illustrated, since the same controller used in the linear example was used for the nonlinear system.
6 State Measurement Noise

In this chapter, the model-free sliding mode controller is tested under the presence of system disturbances and uncertainties. All systems, which were used previously as illustrative examples, were considered to be “perfect” models, without disturbances and uncertainties, with the exception of the cases with the non-unitary control input gain, which was considered to an unknown and time-varying variable, characterizing an uncertainty.

According to the results of the previous chapter simulations, the model-free SMC method showed to be capable to handle system uncertainties, such as the variation of the control input gain. In real physical systems, a common problem that occurs is the noise, which is characterized as system disturbance. However, the noise effects cannot be handled by the model-free sliding mode controller if the controller parameters are chosen arbitrarily, since the tracking performance of the controller becomes unacceptable. Therefore, a method to select the controller parameters is required and proposed in this chapter. Since the model-free sliding mode control is solely based on state measurements and previous control inputs, the only possible source of noise is the one generated by the sensors used to sense the system’s states values, known as state measurement noise.

The chapter is outlined as follows: Section 6.1 briefly describes the noise that is added in the simulations and how to obtain its probabilistic properties. Section 6.2 shows the results of implementing the model-free sliding mode controller into a system with noise if the parameters are chosen freely. In Section 6.3, a method to select the controller parameters is proposed and two examples are illustrated to test the method.

6.1 Noise Properties and Characterization

Noise is a fundamental variable to be considered when designing a control system, since it typically limits the overall performance of the closed-loop system. Thus, the problem of noise reduction has attracted a considerable amount of attention over the past decades and numerous different techniques were developed. A commonly used technique, for example, is the utilization of low-pass filters, which simply cut high frequency signals.
In order to implement a controller into a physical system, electronic devices, which are used as sensors and actuators, are required. Unfortunately, any electronic device have multiple sources of noise. Many of these sources are related to the device type and the manufacturing quality, such as the “flicker” noise.

As any stochastic process, noise cannot be eliminated, just reduced. Every random process is characterized by probabilistic properties such as: variance, probability distribution function and probability density function, which are usually given by the manufacturer datasheet. The spectral density function is frequency related and it is usually measured as \( \frac{W}{\sqrt{\text{Hz}}} \). This property can also be measured in \( \frac{V}{\sqrt{\text{Hz}}} \), since the power in a resistive element is proportional to the square of the voltage across it. As an accelerometer is used as reference to select the noise to be used in the simulations, the spectral density is given by \( \frac{g}{\sqrt{\text{Hz}}} \), where \( g \) is the g-force unit (acceleration unit).

Another important noise characteristic is the quadratic mean, or root mean square (RMS). The quadratic mean is related to the noise probability density function \( \rho_n \) and can be approximated by the following:

\[
y_{\text{rms}} = \rho_n(\sqrt{B_\omega})
\]

(6.1)

where \( B_\omega \) is the bandwidth of the system. The quadratic mean, by the other hand, is required to compute the noise variance. According to Papoulis and Unnikrishna [16], the variance is related to the quadratic mean by:

\[
y_{\text{rms}}^2 = \bar{y}^2 + \sigma_n^2
\]

(6.2)

where \( \bar{y} \) is the mean of the noise and \( \sigma_n^2 \) is the noise’s variance. If the mean of the noise is zero, which generally is true, the variance will be exactly the value of the quadratic mean.

Lastly, another noise characteristic needs to be defined and it is a purely probabilistic variable, which is the peak to peak noise value. For example, if the user defines that the peak to peak noise value is six times the standard deviation \( 6(\sigma_n) \), the probability that the noise is within \( 6(\sigma_n) \) is 99.7%, considering a Gaussian distributed noise with zero mean. Others values can be used as well and different peak to peak noise values can be obtained, by using a standard normal distribution table, for a Gaussian distributed noise. However, since noise is a stochastic process, it is impossible to guarantee a peak to peak value which has the probability of 100% to be within the bounds.
To obtain the noise properties required to simulate the noise in MATLAB, the accelerometer ADXL206 from Analog Devices is used as reference. According to the manufacturer datasheet [17], the noise generated by this device is uncorrelated and has Gaussian distribution with the following probability density function and bandwidth:

$$\rho_n = 110\mu g \sqrt{Hz}$$

$$B_\omega = 1.6kHz$$

Thus, the square root mean and the standard deviation are computed by using Eq. (6.2) and Eq. (6.1), which results into:

$$\gamma_{rms} = 0.0044$$

$$\sigma_n = 0.0044$$

(6.4)

With those values determined, it is possible to simulate the noise using MATLAB. A 30 seconds sample of the noise can be seen in Figure 6.1. Examining the noise sample, indeed the noise peak to peak value remains inside \(6(\sigma_n)\) practically all simulation time.

![Gaussian Noise d(t)](image)

Figure 6.1: Noise sample

### 6.2. Results in the Model-Free SMC with Arbitrarily Assigned Parameters

For the next simulation, a noise, which probabilistic properties are defined by Eq. (6.4), is added to the states’ measurements of the model-free sliding mode controller used for a second-order nonlinear system with non-unitary control input gain, as illustrated in Section 5.3.4.2. The same controller parameters and Simulink models were used.
Figure 6.2 shows that the position state trajectory is tracking the reference signal with acceptable performance. The position tracking error, displayed by Figure 6.3, is less than 2e-02, clearly due to the state measurement noise.

Figure 6.2: Position comparison for the second-order linear example with non-unitary control input gain and measurement noise  
Figure 6.3: Position tracking error for the second-order linear example with non-unitary control input gain and measurement noise

Figure 6.4 shows that the velocity state is in a good agreement with the desired velocity. Figure 6.5 displays the velocity tracking error of the system, with values less than 2.5e-02.

Figure 6.4: Velocity comparison for the second-order linear example with non-unitary control input gain and measurement noise  
Figure 6.5: Velocity tracking error for the second-order linear example with non-unitary control input gain and measurement noise

Figure 6.6 shows the acceleration comparison between the acceleration state and the reference signal. The difference between the trajectories is unacceptable. The acceleration tracking error is huge, as displayed by Figure 6.7.

Figure 6.6: Acceleration comparison for the second-order linear example with non-unitary control input gain and measurement noise  
Figure 6.7: Acceleration tracking error for the second-order linear example with non-unitary control input gain and measurement noise

97
Figure 6.6: Acceleration comparison for the second-order linear example with non-unitary control input gain and measurement noise

Figure 6.7: Acceleration tracking error for the second-order linear example with non-unitary control input gain and measurement noise

Figure 6.8 displays the updated sliding condition of the closed-loop system. Clearly, the sliding condition is not satisfied, which is not admissible. The control effort displayed by Figure 6.9 is noisy and the magnitude has increased, compared to the control effort of the same control system implemented without measurement noise, as shown by Figure 5.58.

Figure 6.8: Sliding condition for the second-order linear example with non-unitary control input gain and measurement noise

Figure 6.9: Control effort for the second-order linear example with non-unitary control input gain and measurement noise

The model-free sliding mode controller did not achieve acceptable responses in the presence of state measurement noise. The sliding condition was not satisfied, therefore, nothing about the closed-loop stability can be concluded. The position and velocity states presented an acceptable
tracking response. However, the tracking performance of the acceleration state is completely unacceptable. The control effort became noisy and increased its magnitude.

In order to explain the unacceptable tracking response of the acceleration state that occurred in the previous simulation, and sliding condition transgression, we need to analyze the control law obtained for that system, Eq. (5.66):

\[ \hat{u} = \tilde{b}^{-1} \left[ -\lambda (\dot{x} - \dot{x}_d) - (\ddot{x} - \ddot{x}_d) 
- (|\beta - 1| |\ddot{x} - \ddot{x}_d| + |\beta - 1| |\dot{x} - \dot{x}_d| + |\tilde{b}\sigma_u(u_{k-2} - u_{k-1})| + \beta \eta 
- \phi) \text{sat} \left( \frac{S}{\phi} \right) \right] + 2u_{k-1} - u_{k-2} \]  

(6.5)

According to the equation above, \( \lambda \) actuate as a gain for the difference between the derivative of the actual state and the derivative of the desired one and is also inversely proportional to the size of the boundary layer, since, if \( \lambda \) is increased, the size of the boundary layer decreases. If the boundary layer size decreases, the controller becomes more aggressive, since more control effort is required to maintain the system states within a smaller boundary layer (better precision). In fact, if \( \lambda \) is chosen to be extremely big, the control effort chattering may appear even with a smoothing boundary layer, since a great amount of switching will occur in order to the control system be able to maintain the system’s states inside the boundary layer, which will behave almost as a relay function.

Also, the sliding surface is directly proportional to the difference between the actual system’s states and the desired ones, as shown in Eq. (3.4). Therefore, it is not reasonable to reduce the size of the boundary layer, by increasing \( \lambda \), in a manner that it will become smaller than the peak to peak value of the noise, considering that the noise cannot be eliminated. Therefore, if the boundary layer is chosen to be smaller than the peak to peak value of the noise, the sliding condition will not be satisfied, since the sliding surface will transgress the boundary layer.

The issue of arbitrarily assign the controller parameters was not noticed when the control system was implemented without considering measurement noise because the controller was able to reduce the difference between the actual states and the desired ones, even if \( \lambda \) was extremely large. However, when noise is inserted into the system, it is not possible to eliminate the difference, which causes the controller be more aggressive, resulting an increased magnitude of the control
effort, since the controller tries unsuccessfully to reduce the difference between the system trajectories and the reference signal.

Lastly, another issue that explains the poor tracking performance of the acceleration state is the algebraic loop that exists in the model-free sliding mode controller control law. The algebraic loop happens because the higher order state is directly fed by itself through the control input. Thus, if the control input amplifies the noise, the higher order state will also become noisy, which explains the tracking performance regarding the acceleration state, which is the higher order state.

6.3. Controller Parameters Selection Method

Only the controller’s parameters need to be modified to reduce the effects of state measurement noise on the model-free sliding mode controller. Hence, a method to choose those parameters is required. To ensure asymptotic stability of the closed-loop system, the sliding condition must be satisfied, i.e., the sliding surface must be within the boundary layer. In order to make the derivation simpler, the initial condition of the boundary layer is used:

\[ \varphi \approx \varphi(0) = \frac{\eta}{\lambda} \tag{6.6} \]

where \( \eta \) is strictly a small positive number. Also, the sliding surface, Eq. (3.4), is defined as:

\[ s = \left( \frac{d}{dt} + \lambda \right)^{n-1} \tilde{x} \tag{6.7} \]

where \( n \) is the order of the system and \( \tilde{x} \) is the difference between the actual state and the desired one. As we are interested in the gain that \( \lambda \) gives to the difference between the actual states and the reference signal, the sliding surface can be approached by:

\[ s \approx \lambda^{n-1}(\tilde{x}) \approx \lambda^{n-1}(x - x_d + \gamma) \approx \lambda^{n-1}V_{pp} \tag{6.8} \]

where \( \gamma \) is the state measurement noise and \( V_{pp} \) is the peak to peak value of the noise. Thus, in order to the sliding surface remain inside the boundary layer, the following equation must be satisfied:

\[ \lambda^{n-1}V_{pp} \leq \frac{\eta}{\lambda} \tag{6.9} \]

which can be simplified into:
\[ \lambda^n \leq \frac{\eta}{V_{pp}} \]  \hspace{1cm} (6.10)

or:

\[ \lambda \leq \left( \frac{\eta}{V_{pp}} \right)^{\frac{1}{n}} \]  \hspace{1cm} (6.11)

The \(\eta\) value also has to be modified in order to the sliding condition be satisfied at initial time. The following equation is proposed to select the updated \(\eta\) value:

\[ \eta_n = \eta + \left( \frac{\sigma_n}{\eta} \right) \left( \frac{\lambda}{2} \right) \]  \hspace{1cm} (6.12)

where \(\sigma_n\) is the noise variance and \(\eta_n\) is the updated \(\eta\) value.

### 6.4. Illustrative Examples

In order to test the method to select the controller’s parameters in presence of state measurement noise, two illustrative examples are presented. The first example simulates the exact control system described in Section 4.2.4.1.1, a first-order linear system with unitary control input gain, and the second example illustrates the control system used in Section 5.3.4.2, a second-order nonlinear system with non-unitary control input gain. However, with noise added to the state measurements. The noise included is characterized by the probabilistic properties defined in Eq. (6.4).

#### 6.4.1 First-order linear system with unitary control input gain

For a first-order system, and considering the noise defined in Eq. (6.4), \(\lambda\) is updated by using Eq. (6.11), which results into:

\[ \lambda \leq \left( \frac{0.1}{0.0044(6)} \right)^{\frac{1}{n}} \leq 3.78 \]  \hspace{1cm} (6.13)

There is a tradeoff that must be considered when choosing \(\lambda\). According to the equation above, if \(\lambda\) is less than 3.78, it is guaranteed that the sliding condition will be satisfied and the closed-loop system will be asymptotically stable. However, nothing about the tracking performance is said. If \(\lambda\) is chosen to be near the upper bound, \(\lambda\) will actuate as a gain for the noise, which can result into an unacceptable tracking performance of the higher order state, since the control input will become noisy. By the other hand, if \(\lambda\) is too small, the boundary layer will expand and tracking precision...
will be lost. After several simulations, the “optimal” value selected for \( \lambda \) is 1, which satisfies Eq. (6.13). Then, using Eq. (6.12), the updated \( \eta \) becomes:

\[
\eta = 0.1 + \left( \frac{0.0044}{0.1} \right) \left( \frac{1}{2} \right) = 0.122
\]  

(6.14)

The exact Simulink diagrams shown in Section 4.2.4.1.1 is used, with the difference that measurement noise was inserted and the controller parameters are updated as follows:

Table 6.1: Controller parameters for a first-order linear system with measurement noise

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>1</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.122</td>
</tr>
</tbody>
</table>

Figure 6.10 displays the position comparison of the closed-loop system. The agreement between the state position trajectory and the reference signal is acceptable, with errors less than 2e-02, as displayed by Figure 6.11.

Figure 6.10: Position comparison for the first-order linear example with measurement noise  
Figure 6.11: Position tracking error for the first-order linear example with measurement noise

Figure 6.12 shows that the tracking performance of the velocity state is good. Figure 6.13 displays the velocity tracking error of the system, which is less than 2e-02.
Figure 6.12: Velocity comparison for the first-order linear example with measurement noise

Figure 6.13: Velocity tracking error for the first-order linear example with measurement noise

Figure 6.14 shows that the sliding condition is satisfied all simulation time. The control effort, displayed by Figure 6.15, is smooth without any chattering.

Figure 6.14: Sliding condition for the first-order linear example with measurement noise

Figure 6.15: Control effort for the first-order linear example with measurement noise

Even though tracking precision was lost, compared to the results of the same control system without measurement noise, the model-free sliding mode controller achieved an acceptable tracking performance for all system’s states. The problem regarding the higher order state was solved and the control effort is smooth. The boundary layer expanded, which means that less tracking precision is guaranteed. However, the asymptotic stability of the closed-loop system is guaranteed, since the sliding condition was satisfied.
6.4.2 Second-order nonlinear system with non-unitary control input gain

For a second-order system, using Eq. (6.11) and Eq. (6.12), \( \lambda \) is updated as:

\[
\lambda \leq \left( \frac{0.1}{0.0044(6)} \right)^{\frac{1}{2}} \leq 1.95
\] (6.15)

For the same reasons mentioned in the previous example, \( \lambda \) is set to 1, which satisfies the equation above. The exact control system built in Simulink shown in Section 5.3.4.2 is used, with the difference that measurement noise is inserted and the controller parameters are updated as:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_u )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>1</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.122</td>
</tr>
</tbody>
</table>

Table 6.2: Controller parameters for a second-order nonlinear system with measurement noise

Figure 6.16 shows that the state position is in a good agreement with the desired trajectory. The position tracking error is less than 2e-02, as displayed by Figure 6.17.

Figure 6.19 displays the velocity comparison of the closed-loop system. The velocity state is tracking the reference signal with good performance, with errors less than 2e-02, as shown by Figure 6.18.
Figure 6.18: Velocity comparison for the second-order nonlinear example with non-unitary control input gain, measurement noise and parameters updated

Figure 6.19: Velocity tracking error for the second-order nonlinear example with non-unitary control input gain, measurement noise and parameters updated

Figure 6.20 shows that the acceleration state is tracking the reference signal with acceptable errors. The response is more acceptable than the one obtained by arbitrarily assigning the controller parameters, as shown in Figure 6.6. The error is less than 4e-02, as displayed by Figure 6.21.

Figure 6.20: Acceleration comparison for the second-order nonlinear example with non-unitary control input gain, measurement noise and parameters updated

Figure 6.21: Acceleration tracking error for the second-order nonlinear example with non-unitary control input gain, measurement noise and parameters updated

Figure 6.22 shows that the sliding condition is satisfied all simulation time. The control effort, displayed by Figure 6.23, is smooth, which is better in performance compared to the control effort obtained in Figure 6.9, when the controller’s parameters were arbitrarily assigned.
It was shown that the problems regarding the state measurement noise are solved by selecting the correct controller’s parameters. The model-free sliding mode controller obtained good overall performance for a second-order nonlinear system with non-unitary control input gain and state measurement noise. Therefore, the robustness against system uncertainties and disturbances was observed in the model-free sliding mode controller.

The problems mentioned in Section 6.2, where a noisy control effort was obtained and the higher order state was tracking the reference signal poorly, were solved. The higher order state is now tracking the reference signal with acceptable performance without any noisy characteristic. The other system’s states also obtained a good tracking response. The sliding condition is now satisfied, which implies that the closed-loop system is asymptotic stable. Lastly, the control effort is smooth, without any chattering or noise.
7 Conclusions

A model-free sliding mode controller, which is solely based on previous control inputs, system state measurements and on the knowledge of the system’s order, was successfully derived in this work. It was shown that the controller is indeed a model-free scheme, as the system model was not used to derive the control law. The controller was tested on first and second-order systems, linear and nonlinear, and outstanding tracking performance was observed and closed-loop asymptotic stability was shown regardless of the system under study. A new system approach was developed, where a new parameter, defined as estimation of the control input error, was used and a more precise controller was obtained. Comparing the simulation results with the ones obtained by other researches, for the same control systems used as illustrative examples, the SMC developed in this work achieved better tracking response and required less control effort.

The first case examined in this work was for systems with unitary control input gain. At a first step, the model-free sliding mode controller was implemented with a relay function as the discontinuous term. As expected, the controller did not achieve acceptable responses, since the relay function introduces a high frequency signal, due the high activity of the control effort switching required to maintain the system trajectories inside the sliding surface. However, the controller managed to drive the system trajectories onto the desired ones, with very poor performance for the higher order state, and the closed-loop system was asymptotic stable, since the sliding condition was satisfied in all simulation time. In order to reduce control effort chattering, a smoothing boundary layer was used in place of the relay function. In that manner, the model-free SMC obtained an outstanding tracking performance and all system trajectories were in excellent agreement with the general reference signal. The control effort chattering was eliminated and the condition for closed-loop asymptotic stability was satisfied, since the sliding surface remained inside the boundary layer.

The second case addressed the derivation of a model-free SMC for systems with non-unitary control input gain. In order to keep the model-free characteristic of the SMC, it was assumed that the control input gain was a time-varying unknown variable. However, the bounds of the control input gain were considered to be known, which is a reasonable assumption. The model-free SMC was implemented with the smoothing boundary layer, since the one implemented with the relay function obtained unacceptable overall performance. The control input bounds were presumed to
be considerably large, in order to test the controller for extreme cases. Still, the control system managed to drive the system trajectories onto desired ones with excellent precision. The control input magnitude increased, since more control effort was required to deal with the system uncertainty introduced by the variation of the control input gain. No control effort chattering was observed and the closed-loop system achieved asymptotic stability, since the sliding surface was maintained within the boundary layer in every case.

The state measurement noise problem was the last case examined. It was shown that the user cannot arbitrarily assign the model-free SMC parameters, since the parameters actuate as a gain for the difference between the system trajectories and the reference signal. Thus, if state measurement noise is added to the control system, the SMC parameters will actuate as a gain for the noise resulting into a noisy control effort. Consequently, since the higher order state is directly fed by the control input, the higher order state tracking performance will become noisy as well, resulting in unacceptable tracking response. In addition, the sliding condition is not satisfied if the parameters are freely defined, therefore, the closed-loop asymptotic stability is not guaranteed. Thus, a new method to choose the SMC parameters was proposed. The model-free SMC implemented with those updated parameters achieved more acceptable results. The problem with the higher order state becoming noisy was eliminated and the closed-loop system asymptotic stability condition was satisfied. Compared to the other examples without state measurement noise, the boundary layer expanded, implying a loss in tracking precision. However, it is not reasonable to assure higher precision if measurement noise is considered, by the fact that noise is a stochastic process and cannot be eliminated. The magnitude of the control effort also increased, since the control system attempts, unsuccessfully, to reduce the difference between the system state’s trajectories and the reference signal in the presence of noise. Still, the method proved to be robust and implementable in the presence of state measurement noise.

7.1. Future Work

There are several ways to improve the model-free sliding mode controller derived in this work. The first suggestion is to solve the algebraic loop contained in the controller algorithm. Even though the algebraic loop effects wouldn’t appear when the controller is implemented in a real physical system, it resulted into a large tracking error in the start of the simulations, which was quickly reduced by the controller.
As mentioned by Sen et al. [7], the bounds of system uncertainties are usually overestimated which yields excessive control effort. Therefore, to reduce the control effort, the control input estimation error and the control input gain bounds could be estimated using techniques such as online parameter estimation. In addition, the parameter $\lambda$ could be made a time-varying variable and should be optimally updated in order to satisfy the sliding condition regardless of the system under study characteristics.

The measurement noise problem could be also reduced by integrating the controller with filters or other techniques to reduce noise. The model-free SMC could be also extended to multi-input systems, output tracking, other class of systems, and tested in systems with saturated actuators, time delays, minimum phase, and other particular system characteristics.

7.2. Applications

The model-free SMC scheme proposed in this work can be applied in any system as long some requirements are met. The system must be single-input, autonomous and in companion form. For systems with order higher than two, the same procedure to derive the model-free SMC for first and second-order systems can be used, however, the sliding surface must be updated accordingly, which will result into a different control law and switching gain.

The main advantage of this method is that it does not require any knowledge of the system model, only the order of the system and the bounds of the control input gain, which can be estimated. This is quite useful if a system needs more than one controller, such as an Unmanned Aircraft System (UAS). In that manner, the implementation is quite simple and this scheme guarantees a robust and excellent tracking performance.
Bibliography


Appendix A

The MATLAB codes that were used for each system are displayed next. The Simulink diagrams are shown in their respective sections.

Appendix A1 - First-order linear/nonlinear system with relay function MATLAB code

```matlab
%% Model Free Sliding Mode Controller - 1st order - Signum function
clear all;
warning('on');
%% I) Define the controller parameters:
% Define lambda
lambda=20;
% Define upper bound of the error estimation: (between 0 and 1):
su=0.5;
% Define eta;
eta=0.1;
% Define the x_desired (A*sin(w*t+phase)):
% Frequency of the x_desired:
w_xd=pi/2;
% Amplitude of the x_desired:
a_xd=1;
% Phase of the x_desired:
p_xd=0;
% Define the initial conditions for the x_desired:
x0=0;
%% Run the Simulation:
% Define tf (simulation time)
tf=30;
% [0] for nonlinear and [1] for linear system:
st=0;
if st==1
    sim ModelFree_SMC_1storder_sgn_linear_withupdatedsurface
else
    sim ModelFree_SMC_1storder_sgn_nonlinear_withupdatedsurface
end;
%% Plot the results:
% Plot the state trajectories
figure(1);
set(gca,'FontSize',22)
plot(tout,x,'k-','Linewidth',1);
hold on;
plot(tout,xd,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)', 'x_d(t)');
grid on;
hold off;
% axis([-inf inf -1.2 1.2])
% Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot,'k-','Linewidth',1);
hold on;
plot(tout,xd_dot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Velocity Comparison');
legend('xdot(t)', 'x_ddot(t)');
grid on;
% axis([-inf inf -2 2])
hold off;
% Plot the position tracking error
figure(3);
set(gca,'FontSize',22)
plot(tout,x-xd,'b','Linewidth',1);
```

112
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
axis([-inf inf -3e-5 3e-5]);
legend('e_p');
grid on;

%Plot the velocity tracking error
figure(4);
set(gca,'FontSize',22)
plot(tout,xdot-xd_dot,'r','Linewidth',1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
legend('e_v');
axis([-Inf inf -3e-1 3e-1]);
grid on;
hold on;
hold off;

%Plot the sliding condition
figure(5);
set(gca,'FontSize',22)
plot(tout,-eta.*abs(s),'m--','Linewidth',1)
xlabel('Time(s)');
ylabel('s.*sdot-t & -eta|s|');
title('s*sdot<-eta*|s|');
axis([-inf inf -5e-6 3e-6]);
grid on;
hold off;

%Plot the switching gain:
figure(6);
set(gca,'FontSize',22)
plot(tout,K,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

%Plot the control effort:
figure(7);
set(gca,'FontSize',22)
plot(tout,u,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;

Appendix A2 - First-order linear/nonlinear system with smoothing boundary layer

MATLAB code

%% Model Free Sliding Mode Controller - 1st order - Smoothing Boundary Layer
clc;
clear all;
warning('on');

%% I) Define the controller parameters:
Define lambda
lambda=20;

%Define upper bound of the error estimation: (between 0 and 1):
su=0.5;

%Define eta;
eta=0.1;

% Define the x_desired (A*sin(w*t+phase)):
Frequency of the x_desired:
w_xd=pi/2;
Amplitude of the x_desired:
a_xd=1;
Phase of the x_desired:
p_xd=0;
% Define the initial conditions for the x_desired:
x0=0;
% Run the Simulation:
% Define tf (simulation time):
tf=30;
% [0] for nonlinear and [1] for linear system:
st=0;
if st==1
  sim ModelFree_SMC_1storder_WBL_linear_withupdatedsurface
else
  sim ModelFree_SMC_1storder_WBL_nonlinear_withupdatedsurface
end
% Plot the results:
% Plot the state trajectories
figure(1);
set(gca,'FontSize',22)
plot(tout,x,'k-','Linewidth',1);
hold on;
plot(tout,xd,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)', 'x_d(t)');
grid on;
hold off;
axis([-inf inf -1.2 1.2]);
% Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot,'k-','Linewidth',1);
hold on;
plot(tout,xd_dot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Velocity Comparison');
legend('xdot(t)', 'x_ddot(t)');
grid on;
axis([-inf inf -1.7 1.7]);
hold off;
% Plot the position tracking error
figure(3);
set(gca,'FontSize',22)
plot(tout,x-xd,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
axis([-inf inf -2e-6 2e-6]);
legend('e_p');
grid on;
% Plot the velocity tracking error
figure(4);
set(gca,'FontSize',22)
plot(tout,xdot-xd_dot,'r','Linewidth',1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
axis([-inf inf -3e-4 3e-4]);
legend('e_v');
grid on;
% Plot the sliding condition
figure(5)
set(gca,'FontSize',22)
plot(tout,abs(phi),'r');
hold on;
plot(tout,-1*abs(phi),'b');
plot(tout,s,'k');
xlabel('Time(s)');
title('Boundary Layer');
legend('phi', '-phi', 's');
grid on;
hold off;
Appendix A3 - Second-order linear/nonlinear system with smoothing boundary layer

MATLAB code

```matlab
% Model Free Sliding Mode Controller - 2nd order - Boundary Layer - linear
clc;
clear all;
warning('on');

% I) Define the controller parameters:
% Define lambda
lambda=20;
% Define upper bound of the error estimation: (between 0 and 1):
su=0.5;
% Define eta;
eta=0.1;

% Define the x_desired (A*sin(w*t+phase)):
% Frequency of the x_desired:
w_xd=pi/2;
% Amplitude of the x_desired:
a_xd=1;
% Phase of the x_desired:
p_xd=0;
% Define the initial conditions for the x_desired:
x0=0;
xdot0=pi/2;
% Define the constants of the open loop system:
c=0.8;
k=2;
m=2;
% Run the Simulation:
% Define tf:
tf=30; %Simulation Time
% [0] for nonlinear and [1] for linear system:
st=0;
if st==0
    sim ModelFree_SMC_2ndorder_WBL_linear;
else
    sim ModelFree_SMC_2ndorder_wbl_nonlinear;
end

% Plot the results:
% Plot the state trajectories
figure(1);
set(gca, 'FontSize',22)
plot(tout,x(:,1),'k-','Linewidth',1);
hold on;
plot(tout,xd,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)', 'x_d(t)');
```
grid on;
hold off;
axis([-inf inf -1.2 1.2]);

%Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Velocity Comparison');
legend('xdot(t)', 'x_ddot(t)');
grid on;
axis([-inf inf -1.7 1.7]);
hold off;

%Plot the second derivative of the state trajectories:
figure(3);
set(gca,'FontSize',22)
plot(tout,xdotdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dotdot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdotdot(t)');
title('Acceleration Comparison');
legend('xdotdot(t)', 'x_dddot(t)');
grid on;
axis([-inf inf -2.8 2.8]);
hold off;

%Plot the position tracking error
figure(4);
set(gca,'FontSize',22)
plot(tout,x(:)-xd,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('|e_p(t)|');
title('Position tracking error');
legend('|e_p|');
grid on;

%Plot the velocity tracking error
figure(5);
set(gca,'FontSize',22)
plot(tout,xdot(:)-xd_dot,'r','Linewidth',1);
xlabel('Time(s)');
ylabel('|e_v(t)|');
title('Velocity tracking error');
legend('|e_v|');
grid on;
axis([-inf inf -5e-7 5e-7]);

%Plot the acceleration tracking error:
figure(6);
set(gca,'FontSize',22)
plot(tout,xdotdot(:)-xd_dotdot,'k','Linewidth',1);
xlabel('Time(s)');
ylabel('|e_a(t)|');
title('Acceleration tracking error');
axis([-inf inf -1e-4 1e-4]);
legend('|e_a|');
grid on;

%Plot sliding condition
figure(7)
set(gca,'FontSize',22)
plot(tout,abs(phi(:)),'r');
hold on;
plot(tout,-1*abs(phi(:)),'b');
plot(tout,s(:),'k');
xlabel('Time(s)');
title('Boundary Layer');
legend('phi',''-phi','s');
grid on;
hold off;

%Plot the switching gain:
figure(8);
set(gca,'FontSize',22)
plot(tout,K(:,1),b,'Linewidth',1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

%Plot the control effort:
figure(9);
plot(tout,u(:,1),b,'Linewidth',1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;

Appendix A4 - First-order linear/nonlinear system with smoothing boundary layer and non-unitary control input gain MATLAB code

```matlab
%% Model Free Sliding Mode Controller - 1st order - WBL - With B MATRIX
clc;
clear all;
warning('on');

%% I) Define the controller parameters:
% Input matrix:
% Variable b:
b_up=5; %Upper bound
b_low=1; %Lower bound
bhat=sqrt(b_up*b_low); %estimation of b
beta=sqrt((b_up)/(b_low));
% Define lambda
lambda=20;
% Define upper bound of the error estimation: (between 0 and 1):
su=0.5;
% Define eta;
eta=0.1;

%% Define the x_desired (A*sin(w*t+phase)):
% Frequency of the x_desired:
w_xd=pi/2;
% Amplitude of the x_desired:
a_xd=1;
% Phase of the x_desired:
p_xd=0;
% Define the initial conditions for the x_desired:
x0=0;
% Define system parameters
b=bhat;
% Define if the parameters will be constant or if they will change with time (1 for non-constant and 0 for constant);
b_switch=1;

%% Run the Simulation:
% Define tf (simulation time):
tf=30;
% [0] for nonlinear or [1] for linear
st=1
if st==1;
sim ModelFree_SMC_1storder_Winputmatrix_WBL_linear_wus;
else
sim ModelFree_SMC_1storder_Winputmatrix_WBL_nonlinear_wus;
end;

%% Plot the results:
% Plot the state trajectories
figure(1);
set(gca,'FontSize',22)
plot(tout,x,'k-', 'Linewidth',1);
hold on;
plot(tout,xd,'m--', 'Linewidth',2);
```
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)', 'x_d(t)');
grid on;
hold off;
axis([-inf inf -1.2 1.2]);

% Plot the derivative of the state trajectories
figure(2);
set(gca, 'FontSize', 22)
plot(tout, xdot, 'k-', 'Linewidth', 1);
hold on;
plot(tout, xd_dot, 'm--', 'Linewidth', 2);
xlabel('Time(s)');
ylabel('xdot(t)');
legend('xdot(t)', 'x_ddot(t)');
grid on;
axis([-inf inf -1.7 1.7]);
hold off;

% Plot the position tracking error
figure(4);
set(gca, 'FontSize', 22)
plot(tout, x-xd, 'b', 'Linewidth', 1);
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
axis([-inf inf -2.5e-6 2.5e-6]);
legend('e_p');
grid on;

% Plot the velocity tracking error
figure(5);
set(gca, 'FontSize', 22)
plot(tout, xdot-xd_dot, 'r', 'Linewidth', 1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
axis([-inf inf -3.5e-4 3.5e-4]);
legend('e_v');
grid on;

% Plot the sliding condition
figure(6)
set(gca, 'FontSize', 22)
plot(tout, abs(phi), 'r');
hold on;
plot(tout, -1*abs(phi), 'b');
plot(tout, s, 'k');
xlabel('Time(s)');
ylabel('phi');
title('Boundary Layer');
legend('phi', '-phi', 's');
grid on;
hold off;

% Plot the switching gain
figure(8);
set(gca, 'FontSize', 22)
plot(tout, K, 'b', 'Linewidth', 1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

% Plot the control effort
figure(9);
set(gca, 'FontSize', 22)
plot(tout, u, 'b', 'Linewidth', 1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;

% Plot the Input Matrix b
Appendix A5 - Second-order linear/nonlinear system with smoothing boundary layer and non-unitary control input gain MATLAB code

```matlab
figure(10);
set(gca,'FontSize',22)
plot(tout,b_s,'r--','Linewidth',2);
xlabel('Time(s)');
ylabel('b(t)');
legend('b');
grid on;
title('Variation of Input Matrix');
```

```
%% Model Free Sliding Mode Controller - 2nd order - WBL - With B MATRIX
clc;
clear all;
warning('on');

%% I) Define the controller:
%Input matrix:
%Variable b:
b_up=5; %Upper bound
b_low=1; %Lower bound
bhat=sqrt(b_up*b_low); %estimation of b
beta=sqrt((b_up)/(b_low));

%Define lambda
lambda=20;
%Define eta:
eta=0.1;
%Define upper bound of the error estimation: (between 0 and 1):
su=0.5;
% Define the x_desired (A*sin(w*t+phase)):
%Frequency of the x_desired:
w_xd=pi/2;
%Amplitude of the x_desired:
a_xd=1;
%Phase of the x_desired:
p_xd=0;
%Define the initial conditions for the x_desired:
x0=0;
xdot0=pi/2;
% Define the constants of the "actual" system:
c=0.8;
k=2;
m=2;
b=bhat;
% Define if the parameters will be constant or if they will change with
%time (1 for non-constant and 0 for constant):
b_switch=1;
% Run the Simulation:
%Define tf:
tf=30; %Simulation Time
%[0] for nonlinear or [1] for linear
st=1
if st==1;
sim ModelFree_SMC_2ndorder_Winputmatrix_WBL_linear;
else
sim ModelFree_SMC_2ndorder_Winputmatrix_WBL_nonlinear;
end
%Plot the results:
%Plot the state trajectories
figure(1);  
set(gca,'FontSize',22)
plot(tout,x(:,1),'k-','Linewidth',1);
hold on;
plot(tout,xd,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)','x_d(t)');
grid on;
```
hold off;
axis([[-inf inf -1.2 1.2]]);
%Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Velocity Comparison');
legend('xdot(t)','x_ddot(t)');
grid on;
axis([[-inf inf -1.7 1.7]]);
hold off;

%Plot the second derivative of the state trajectories:
figure(3);
set(gca,'FontSize',22)
plot(tout,xdotdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dotdot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Acceleration Comparison');
legend('xdotdot(t)','x_dddot(t)');
grid on;
axis([[-inf inf -2.8 2.8]]);
hold off;
%Plot the position tracking error
figure(4);
set(gca,'FontSize',22)
plot(tout,x(:)-xd,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
legend('e_p');
axis([[-inf inf -3e-7 3e-7]]);
grid on;

%Plot the velocity tracking error
figure(5);
set(gca,'FontSize',22)
plot(tout,xdot(:)-xd_dot,'r','Linewidth',1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
legend('e_v');
grid on;
axis([[-inf inf -5e-7 5e-7]]);

%Plot the acceleration tracking error:
figure(6);
set(gca,'FontSize',22)
plot(tout,xdotdot(:)-xd_dotdot,'k','Linewidth',1);
xlabel('Time(s)');
ylabel('e_a(t)');
title('Acceleration tracking error');
axis([[-inf inf -1.2e-4 1.2e-4]]);
legend('e_a');
grid on;

%Plot sliding condition
figure(7)
set(gca,'FontSize',22)
plot(tout,abs(phi(:)),'r');
hold on;
plot(tout,-1*abs(phi(:)),'b');
plot(tout,s(:),'k');
xlabel('Time(s)');
title('Boundary Layer');
legend('phi','-phi','s');
grid on;
hold off;
Appendix A6 - First-order linear system with smoothing boundary layer and measurement noise MATLAB code

```matlab
%% Model Free Sliding Mode Controller - 1st order - Smoothing Boundary Layer - With Noise
clear all;
warning('on');

%% I) Define the controller parameters:
lambda=3;

%% Define the x_desired (A*sin(w*t+phase)):
%Frequency of the x_desired:
w_xd=p/2;
%Amplitude of the x_desired:
a_xd=1;
%Phase of the x_desired:
p_xd=0;

%% Define noise parameters
seed1=round(100*randn(1));
seed2=round(100*randn(1));
std=0.0044; %Standard deviation
tau=0.1+lambda*(std)/0.2; %New eta

%% Run the Simulation:
%Define tf (Simulation Time):
tf=30;

sim ModelFree_SMC_1storder_WBL_linear_withnoise

%% Plot the results:
%%Plot the state trajectories
figure(1);
hold on;
plot(tout,x(:,k--), 'Linewidth',1);
plot(tout,xd(:,m--), 'Linewidth',2);
```

```matlab
%% Plot the switching gain:
figure(8);
set(gca,'FontSize',22)
plot(tout,K','b','Linewidth',1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

%% Plot the control effort:
figure(9);
set(gca,'FontSize',22)
plot(tout,u','b','Linewidth',1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;

%% Plot the Input Matrix b
figure(10);
set(gca,'FontSize',22)
plot(tout,b_s,'r--','Linewidth',2);
xlabel('Time(s)');
ylabel('b(t)');
title('Variation of Input Matrix');
```
title('Position Comparison');
legend('x(t)', 'x_d(t)');
grid on;
hold off;
axis([-inf inf -1.2 1.2]);

%Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot,'k-', 'Linewidth',1);
hold on;
plot(tout,xd_dot,'m--', 'Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Velocity Comparison');
legend('xdot(t)', 'x_ddot(t)');
grid on;
axis([-inf inf -1.7 1.7]);
hold off;

%Plot the position tracking error
figure(3);
set(gca,'FontSize',22)
plot(tout,x-xd,'b', 'Linewidth',1);
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
axis([-inf inf -2.2e-2 2.2e-2]);
legend('e_p');
grid on;

%Plot the velocity tracking error
figure(4);
set(gca,'FontSize',22)
plot(tout,xdot-xd_dot,'r', 'Linewidth',1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
axis([-inf inf -2.2e-2 2.2e-2]);
legend('e_v');
grid on;

%Plot the sliding condition
figure(5)
set(gca,'FontSize',22)
plot(tout,abs(phi),'r');
hold on;
plot(tout,-1*abs(phi),'b');
plot(tout,a,'k');
xlabel('Time(s)');
title('Boundary Layer');
legend('phi', '-phi', 's');
grid on;
hold off;

%Plot the switching gain:
figure(6);
set(gca,'FontSize',22)
plot(tout,K,'b', 'Linewidth',1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

%Plot the control effort:
figure(7);
set(gca,'FontSize',22)
plot(tout,u,'b', 'Linewidth',1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;
Appendix A7 - Second-order nonlinear system with non-unitary control input gain, smoothing boundary layer and measurement noise MATLAB code

```matlab
% Model Free Sliding Mode Controller - 2nd order - WBL - With B MATRIX - With Noise
clc;
clear all;
warning('on');
% I) Define the controller:
%Input matrix variables:
%Variable b:
b_up=5; %Upper bound
b_low=1; %Lower bound
bhat=sqrt(b_up*b_low); %estimation of b
beta=sqrt((b_up)/(b_low));
%Define lambda
lambda=1; %%%0.8 - 4
%Define eta;
eta=0.1;
%Define upper bound of the error estimation: (between 0 and 1):
su=0;
% Define the x_desired (A*sin(w*t+phase)):
%Frequency of the x_desired:
w_xd=pi/2;
%Amplitude of the x_desired:
a_xd=1;
%Phase of the x_desired:
p_xd=0;
%Define the initial conditions for the x_desired:
x0=0;
xdot0=pi/2;
% Define the constants of the "actual" system:
c=0.8;
k=2;
m=2;
b=bhat;
% Define if the parameters will be constant or if they will change with
% time (1 for non-constant and 0 for constant);
b_switch=1;
% Define noise parameters
seed1=round(100*randn(1));
seed2=round(100*randn(1));
seed3=round(100*randn(1));
std=0.0044; %standard deviation
eta=0.1+lambda*(std)/0.2; %New eta
% Run the Simulation:
%Define tf (simulation time):
tf=30;
sim ModelFree_SMC_2ndorder_Winputmatrix_WBL_withnoise_simulink;
% Plot the results:
%Plot the state trajectories
figure(1);
set(gca,'FontSize',22)
plot(tout,x(:),'k-','Linewidth',1);
hold on;
plot(tout,xd,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('x(t)');
title('Position Comparison');
legend('x(t)','x_d(t)');
grid on;
hold off;
axis([-inf inf -1.2 1.2]);
%Plot the derivative of the state trajectories
figure(2);
set(gca,'FontSize',22)
plot(tout,xdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
```

123
title('Velocity Comparison');
legend('xdot(t)', 'x_ddot(t)');
grid on;
axis([-inf inf -1.7 1.7]);
hold off;

figure(3);
set(gca,'FontSize',22)
plot(tout,xdotdot(:),'k-','Linewidth',1);
hold on;
plot(tout,xd_dotdot,'m--','Linewidth',2);
xlabel('Time(s)');
ylabel('xdot(t)');
title('Acceleration Comparison');
legend('xdotdot(t)', 'x_ddotdot(t)');
grid on;
axis([-inf inf -2.8 2.8]);
hold off;

figure(4);
set(gca,'FontSize',22)
plot(tout,x(:)-xd,'b','Linewidth',1);
xlabel('Time(s)');
ylabel('e_p(t)');
title('Position tracking error');
legend('e_p');
axis([-inf inf -2.2e-2 2.2e-2]);
grid on;

figure(5);
set(gca,'FontSize',22)
plot(tout,xdot(:)-xd_dot,'r','Linewidth',1);
xlabel('Time(s)');
ylabel('e_v(t)');
title('Velocity tracking error');
legend('e_v');
axis([-inf inf -2.2e-2 2.2e-2]);
grid on;

figure(6);
set(gca,'FontSize',22)
plot(tout,xdotdot(:)-xd_dotdot,'k','Linewidth',1);
xlabel('Time(s)');
ylabel('e_a(t)');
title('Acceleration tracking error');
axis([-inf inf -4e-2 4e-2]);
legend('e_a');
grid on;

figure(7)
set(gca,'FontSize',22)
plot(tout,abs(phi(:)),'r');
hold on;
plot(tout,-1*abs(phi(:)),'b');
plot(tout,s(:),'k');
xlabel('Time(s)');
title('Boundary Layer');
legend('phi', '-phi', 's');
grid on;
hold off;

figure(8);
set(gca,'FontSize',22)
plot(tout,K(:),'b','Linewidth',1);
xlabel('Time(s)');
ylabel('K(t)');
title('Switching Gain');
legend('K');
grid on;

figure(9);
set(gca,'FontSize',22)
plot(tout,u(:,),'b','Linewidth',1);
xlabel('Time(s)');
ylabel('u(t)');
title('Control Effort');
legend('u');
grid on;

figure(10);
set(gca,'FontSize',22)
plot(tout,b_s,'r--','Linewidth',2);
xlabel('Time(s)');
ylabel('b(t)');
legend('b');
grid on;
title('Variation of Input Matrix');