

## Proactive Approach to Safe Automated Driving: Control of Evasive Maneuvers in Hazardous Scenarios

Gharavi, L.

**DOI**

[10.4233/uuid:7d7ed515-f870-4d7c-bfc2-6b72893b11c0](https://doi.org/10.4233/uuid:7d7ed515-f870-4d7c-bfc2-6b72893b11c0)

**Publication date**

2025

**Document Version**

Final published version

**Citation (APA)**

Gharavi, L. (2025). *Proactive Approach to Safe Automated Driving: Control of Evasive Maneuvers in Hazardous Scenarios*. [Dissertation (TU Delft), Delft University of Technology].  
<https://doi.org/10.4233/uuid:7d7ed515-f870-4d7c-bfc2-6b72893b11c0>

**Important note**

To cite this publication, please use the final published version (if applicable).  
Please check the document version above.

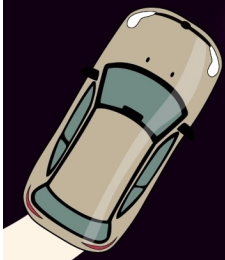
**Copyright**

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

**Takedown policy**

Please contact us and provide details if you believe this document breaches copyrights.  
We will remove access to the work immediately and investigate your claim.

# PROACTIVE APPROACH TO SAFE AUTOMATED DRIVING



*Leila Gharavi*

**PROACTIVE APPROACH TO SAFE AUTOMATED  
DRIVING: CONTROL OF EVASIVE MANEUVERS IN  
HAZARDOUS SCENARIOS**





# **PROACTIVE APPROACH TO SAFE AUTOMATED DRIVING: CONTROL OF EVASIVE MANEUVERS IN HAZARDOUS SCENARIOS**

## **Dissertation**

for the purpose of obtaining the degree of doctor  
at Delft University of Technology  
by the authority of the Rector Magnificus Prof.dr.ir. T.H.J.J. van der Hagen  
chair of the Board for Doctorates  
to be defended publicly on  
Monday 3 March 2025 at 10:00 o'clock

by

**Leila GHARAVI**

Master of Science in Mechanical Engineering,  
Amirkabir University of Technology, Iran  
born in Qom, Iran

This dissertation has been approved by

the promotors: Prof.dr.ir. B.H.K. De Schutter and Prof.dr.ir. S. Baldi

Composition of the doctoral committee:

Rector Magnificus,	chairman
Prof.dr.ir. S. Baldi,	Southeast University, China
Prof.dr.ir. B.H.K. De Schutter,	Delft University of Technology, Netherlands

*Independent members:*

Prof.dr. M. Corno,	Politecnico di Milano, Italy
Prof.dr. M. Johansson,	KTH Royal Institute of Technology, Sweden
Prof.dr. N. van de Wouw	Eindhoven University of Technology, Netherlands
Prof.dr. R.R. Negenborn,	Delft University of Technology, Netherlands
Dr. B. Shyrokau,	Delft University of Technology, Netherlands

*Reserve member:*

Prof.dr.ir. R. Happee,	Delft University of Technology, Netherlands
------------------------	---

This dissertation was supported by the Dutch Research Council (NWO-TTW) within the EVOLVE project (no. 18484) and the EU Horizon 2020 innovation and research program within the Marie Skłodowska-Curie OWHEEL project (no. 872907).



*Keywords:* Collision avoidance, Automated driving, Model predictive control, Hybrid systems, Evasive maneuvers

*Printed by:* Ridderprint, <https://www.ridderprint.nl>

*Cover:* Leila Gharavi

*Style:* TU Delft House Style, with modifications by Moritz Beller  
<https://github.com/Inventitech/phd-thesis-template>

The author set this thesis in  $\text{\LaTeX}$  using the Libertinus and Inconsolata fonts.

ISBN 978-94-6506-978-4

An electronic version of this dissertation is available at <https://repository.tudelft.nl> and <https://phd.leilagaravi.com>.

*Let's keep in mind that in real life the lines are fluid, are not easily drawn, and should not be rigidly maintained.*

— Gabor Maté, *The Myth of Normal*



# CONTENTS

<b>Summary</b>	<b>xi</b>
<b>Samenvatting</b>	<b>xiii</b>
<b>Acknowledgments</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The Concept of Hazard in Automated Driving . . . . .	1
1.2 Proactivity in Hazardous Scenarios. . . . .	3
1.3 Approximation: A Double-Edge Sword. . . . .	4
1.4 Contributions and Organization of this Thesis . . . . .	4
<b>I Hybridization Approach to Computational Efficiency</b>	<b>7</b>
<b>2 Iterative PWA Approximation via Cut-Based Domain Partitioning</b>	<b>9</b>
2.1 Introduction . . . . .	9
2.2 PWA Approximation. . . . .	11
2.2.1 Problem Formulation. . . . .	11
2.2.2 Parametric Definition of Regions. . . . .	12
2.2.3 Approximation Algorithm . . . . .	13
2.2.4 General Remarks. . . . .	15
2.3 Case Study: Vehicle Dynamics . . . . .	16
2.3.1 Nonlinear Vehicle Model . . . . .	16
2.3.2 Implementation and Results . . . . .	17
2.4 Conclusions . . . . .	19
<b>3 PWA Approximation of Multi-Dimensional Nonlinear Systems</b>	<b>21</b>
3.1 Introduction . . . . .	22
3.2 Problem Formulation. . . . .	23
3.3 Parametric Region Definition. . . . .	25
3.4 Cut-Based PWA Approximation . . . . .	29
3.5 Results and Discussion . . . . .	31
3.5.1 Case Study . . . . .	31
3.5.2 Comparison with Approaches in the Literature. . . . .	31
3.6 Conclusions . . . . .	33
<b>4 H4MPC: A Hybridization Toolbox for MPC</b>	<b>35</b>
4.1 Introduction . . . . .	35
4.2 Preliminaries. . . . .	36
4.2.1 Nonlinear Problem Description. . . . .	36
4.2.2 Approximation of the Nonlinear Problem . . . . .	37

4.2.3	Approximation Problem Formulation. . . . .	38
4.3	Toolbox Architecture. . . . .	39
4.3.1	Grid Generation Module . . . . .	39
4.3.2	Model Approximation Module . . . . .	41
4.3.3	Constraint Approximation Module. . . . .	41
4.4	Case Study . . . . .	41
4.4.1	Nonlinear Vehicle Model . . . . .	41
4.4.2	Approximation of the Vehicle Dynamics . . . . .	43
4.4.3	Approximation of the Kamm Circle Constraint. . . . .	43
4.5	Conclusions . . . . .	45
<b>5</b>	<b>Sensitivity Analysis for PWA Approximations of NLPs</b>	<b>47</b>
5.1	Introduction . . . . .	47
5.2	Preliminaries. . . . .	49
5.2.1	Representation of Continuous PWA Functions . . . . .	49
5.2.2	Sensitivity of the Convex Optimization Problem . . . . .	50
5.3	Continuous PWA Approximation of NLPs . . . . .	51
5.4	Theoretical Analysis . . . . .	52
5.5	Case Study . . . . .	55
5.5.1	Case 1: Finding the Confidence Radii. . . . .	56
5.5.2	Case 2: Finding the Approximation Criteria . . . . .	57
5.5.3	Application for NMPC Optimization . . . . .	57
5.6	Conclusions . . . . .	58
<b>II</b>	<b>Collision Avoidance in Automated Driving</b>	<b>61</b>
<b>6</b>	<b>Design and Numerical Analysis of Hybridization Benchmark for NMPC</b>	<b>63</b>
6.1	Introduction . . . . .	64
6.2	Background . . . . .	66
6.2.1	Model Approximation . . . . .	67
6.2.2	Constraint Approximation . . . . .	67
6.2.3	Relation to the State-of-the-Art. . . . .	68
6.3	Grid Generation . . . . .	69
6.4	Approximation Problem Formulation. . . . .	72
6.4.1	Model Approximation . . . . .	72
6.4.2	Constraint Approximation . . . . .	73
6.5	Model and Constraint Hybridization for Vehicle Control . . . . .	75
6.5.1	Nonlinear System Descriptions. . . . .	75
6.5.2	Grid Definition and Coverage . . . . .	77
6.5.3	Model Approximation Results . . . . .	79
6.5.4	Constraint Approximation and Validation . . . . .	81
6.6	Conclusions and Outlook. . . . .	84
6.6.1	Conclusions for Vehicle Control . . . . .	84
6.6.2	Generalized Hybridization Framework . . . . .	85
6.6.3	Next Steps and Future Work . . . . .	86

<b>7</b>	<b>Comparative Assessment of Hybridization in Vehicle Control</b>	<b>87</b>
7.1	Introduction . . . . .	88
7.2	Background . . . . .	90
7.2.1	Model and Constraint Hybridization . . . . .	90
7.2.2	MPC Optimization Problems . . . . .	91
7.3	Comparison Benchmark . . . . .	93
7.3.1	Prediction Model and Physics-Based Constraints . . . . .	93
7.3.2	Control Parameters. . . . .	94
7.3.3	MPC Controllers . . . . .	94
7.3.4	Reference Trajectory . . . . .	94
7.3.5	Driving Scenarios . . . . .	95
7.3.6	Solver Selection . . . . .	96
7.4	Simulation Results . . . . .	96
7.4.1	Ideal Case . . . . .	97
7.4.2	Friction Offset . . . . .	98
7.4.3	Friction Disturbance . . . . .	101
7.4.4	Analysis of Computational Performance . . . . .	102
7.4.5	Handling Limits . . . . .	103
7.5	IPG CarMaker Simulation . . . . .	104
7.6	Discussions and Outlook . . . . .	105
7.6.1	MILP vs. MIQCP . . . . .	105
7.6.2	Robustness to Friction Uncertainty . . . . .	105
7.6.3	Performance Close to Handling Limits . . . . .	106
7.6.4	Overall Computational Performance . . . . .	106
7.7	Conclusions . . . . .	106
<b>8</b>	<b>Efficient Response to Sudden Appearance of Static Obstacles</b>	<b>109</b>
8.1	Introduction . . . . .	110
8.2	Vehicle Model . . . . .	112
8.2.1	Kinematic Model . . . . .	114
8.2.2	Equations of Motion . . . . .	114
8.3	Proposed Control System. . . . .	115
8.4	Design of the Planning Strategy . . . . .	116
8.4.1	MSF Planner . . . . .	116
8.4.2	MPC Planner. . . . .	117
8.4.3	FBL . . . . .	119
8.5	System Implementation . . . . .	119
8.6	Results and Discussion . . . . .	120
8.6.1	Relaxed cases with $v_x = 5\text{m/s}$ . . . . .	120
8.6.2	Lateral assessment with $v_x = 6\text{m/s}$ . . . . .	121
8.6.3	Extreme cases with $v_x = 7\text{m/s}$ . . . . .	121
8.6.4	Comparison with MPC. . . . .	122
8.7	Conclusions . . . . .	123

<b>9</b>	<b>Proactive Collision Avoidance with Stochastic Obstacle Behavior</b>	<b>125</b>
9.1	Introduction . . . . .	126
9.1.1	Motion Planning Challenges in Different Scenarios. . . . .	126
9.1.2	Sources of Uncertainty in Highway Driving . . . . .	127
9.1.3	Computational Efficiency in Emergencies . . . . .	128
9.1.4	Contributions . . . . .	128
9.2	Problem Formulation. . . . .	129
9.3	MMPS Approximation . . . . .	129
9.4	Problem Reformulation and Solution Approach . . . . .	130
9.4.1	Obstacle Vehicle Model. . . . .	130
9.4.2	Hybrid Ego Vehicle Model . . . . .	132
9.4.3	Chance Constraints and Collision Risk Function . . . . .	134
9.4.4	SMPC Optimization Problem. . . . .	137
9.5	Simulations and Results . . . . .	138
9.5.1	Proactivity Assessment. . . . .	139
9.5.2	Assessment of Attainable Trajectories . . . . .	143
9.5.3	Comparison with IPG Motion Planner . . . . .	144
9.5.4	Performance Analysis and Discussion . . . . .	146
9.6	Conclusions . . . . .	149
<b>10</b>	<b>Conclusions</b>	<b>151</b>
10.1	Summary of Research Contributions . . . . .	151
10.2	Future Research Suggestions . . . . .	152
10.3	Outlook and Recommendations . . . . .	153
	<b>Bibliography</b>	<b>155</b>
	<b>Curriculum Vitæ</b>	<b>171</b>
	<b>List of Publications</b>	<b>173</b>



## SUMMARY

Emergency maneuvers on highways present one of the most complex challenges for automated driving. High speeds pushing the vehicle towards nonlinear regimes, coupled with the necessity of swift decision making, complicates the collision avoidance problem to the extent that even expert human drivers may struggle to safely avoid collisions.

Lack of sufficient and reliable data limits applicability of model-free and data-driven control approaches in hazardous scenarios, opening the door to model-based and optimization-based control approaches. However, the unknown behavior of other road users, the sensitivity of the handling limits (e.g., tire saturation) to road conditions, and the amplification of minor steering adjustments on the lateral trajectory due to high speed necessitate the incorporation of nonlinear models in the design. Such nonlinearities should be balanced with the increased complexity and the need for swift responses to hazard.

This thesis addresses the critical challenge of controlling evasive emergency maneuvers for automated driving on highways. As the complexity of the collision avoidance problem arises from its intrinsic connection to safety definitions and the need for computational efficiency, we investigate proactive solutions in the sense of computationally rapid and maximally safe responses, while avoiding unnecessary conservatism. This thesis is structured into two main parts.

In Part I, hybridization – i.e. approximation of nonlinear functions using hybrid systems formalism – is explored as a means to improve the computational efficiency. Chapter 2 introduces a novel PWA approximation technique utilizing parametric cut-based partitioning of domains, extending the applicability of the state-of-the-art methods to multi-dimensional systems. Chapter 3 introduces a more generalized PWA approximation method employing a hinging hyperplane formulation, offering greater flexibility compared to the 2-dimensional plane-based cutting strategy in domain partitioning. As a result, the generalized cut-based approach is able to obtain simpler PWA approximations of nonlinearities in comparison with the parametric cut-based approach for the same desired approximation error. In Chapter 4, we introduce H4MPC, a MATLAB-based open-source toolbox for hybridization of nonlinear control problems in automated driving, and in Chapter 5, the sensitivity of PWA approximation of nonlinear optimization problems with polytopic constraints is analyzed. This analysis can be used in two ways: finding the confidence radius, i.e. a bound on the distance between the approximated and original minima, for a given approximation error, as well as obtaining a required bound on the approximation error for a desired confidence radius.

Part II investigates the problem of proactive collision avoidance in emergency scenarios. Following the results of Part I, we define and provide a hybridization benchmark for nonlinear Model Predictive Control (MPC) in Chapter 6. Next, we extensively investigate different hybridization formulations for hybrid approximation of nonlinear models using the computationally tractable Max-Min-Plus-Scaling (MMPS) systems formalism, as well as nonlinear inequality constraints such as vehicle stability and time saturation limits. In

Chapter 7, this hybridization framework is assessed in a vehicle control benchmark in various driving scenarios such as friction uncertainty, showcasing the efficacy of hybridization in enhancing the computational speed, as well as improving the search across the decision space for the globally optimal solution. Chapter 8 addresses the challenge of real-time collision avoidance on a vehicle with limited computational resources after the sudden appearance of a static obstacle. More specifically, we propose an integrated approach to motion planning that combines efficient feed-forward planning signals to mitigate the feasibility issues of MPC. Finally in Chapter 9, we extend our focus to proactive collision avoidance in dynamic environments using a hybrid stochastic MPC framework for motion planning, validated through rigorous simulations with a high-fidelity vehicle model in IPG CarMaker.

In summary, by developing novel hybridization strategies, comprehensive analytical and numerical investigations, and introducing assessment benchmarks, this thesis tackles the challenges of controlling evasive maneuvers in hazardous scenarios. The newly developed methods throughout this thesis are validated using extensive cases studies and the effectiveness of hybridization in proactive response to danger is demonstrated through high-fidelity simulations as well as real-world experiments. Our recommendations for future work include enhancing robustness, more generalized and extensive sensitivity analysis, and improving the interpretability of control solutions to mimic human behavior, aiding in the acceptance of automated driving.

## SAMENVATTING

Noodmanoeuvres op snelwegen vormen een van de meest complexe uitdagingen voor geautomatiseerd rijden. Hoge snelheden, die het voertuig naar niet-lineaire regimes duwen, gecombineerd met de noodzaak van snelle besluitvorming, bemoeilijken het probleem van botsingvermijding zodanig dat zelfs ervaren menselijke bestuurders moeite kunnen hebben om op een veilige manier botsingen te vermijden.

Een gebrek aan voldoende betrouwbare gegevens beperkt de toepasbaarheid van model-vrije en datagestuurde regelmethoden in gevaarlijke scenarios, waardoor er een duidelijke nood is aan modelgebaseerde en optimalisatiegebaseerde regelmethoden. Het onbekende gedrag van andere weggebruikers, de gevoeligheid van de voertuiglimieten (bijvoorbeeld bandensaturatie) voor wegomstandigheden, en de versterking van kleine stuurcorrecties op de laterale trajectorie door de hoge snelheid, vereisen echter de opname van niet-lineaire modellen in het ontwerp. Dergelijke niet-lineariteiten moeten in balans worden gebracht met de toenemend complexiteit en de behoefte aan snelle reacties op gevaar.

Dit proefschrift behandelt de kritische uitdaging van het controleren van ontwijkende noodmanoeuvres voor geautomatiseerd rijden op snelwegen. Omdat de complexiteit van het botsingvermijdingsprobleem voortkomt uit het intrinsieke verband met veiligheids-definities en de noodzaak van computationele efficiëntie, onderzoeken we proactieve oplossingen in de zin van computationeel snelle en maximaal veilige reacties, terwijl onnodige conservativiteit wordt vermeden. Dit proefschrift is gestructureerd in twee hoofdonderdelen.

In Deel I wordt hybridisatie – dat wil zeggen de benadering van niet-lineaire functies met behulp van hybride systeemformalismen – verkend als een middel om de computationele efficiëntie te verbeteren. Hoofdstuk 2 introduceert een nieuwe PWA-benaderingstechniek die gebruik maakt van parametrische domeinpartitionering, waardoor de toepasbaarheid van de *state-of-the-art* methoden op multidimensionale systemen wordt uitgebreid. Hoofdstuk 3 introduceert een meer algemene PWA-benaderingsmethode die gebruik maakt van een *hinging-hyperplane* formulering, wat meer flexibiliteit biedt vergeleken met de tweedimensionale snijstrategie bij domeinpartitionering. Als resultaat kan de algemene snijbenadering eenvoudigere PWA-benaderingen van niet-lineariteiten verkrijgen in vergelijking met de parametrische snijbenadering bij dezelfde gewenste benaderingsfout. In Hoofdstuk 4 introduceren we H4MPC, een op MATLAB gebaseerde *open-source toolbox* voor de hybridisatie van niet-lineaire regelproblemen bij geautomatiseerd rijden, en in Hoofdstuk 5 wordt de gevoeligheid van PWA-benadering van niet-lineaire optimalisatieproblemen met polytoop-beperkingen geanalyseerd. Deze analyse kan op twee manieren worden gebruikt: voor het vinden van de betrouwbaarheidsradius, d.w.z. een grens op de afstand tussen de benaderde en originele minima, voor een gegeven benaderingsfout, evenals voor het verkrijgen van een vereiste grens op de benaderingsfout voor een gewenste betrouwbaarheidsradius.

Deel II onderzoekt het probleem van proactieve botsingvermijding in noodscenario's. Naar aanleiding van de resultaten van Deel I definiëren en bieden we een hybridisatiebenchmark voor niet-lineaire Model Predictive Control (MPC) in Hoofdstuk 6. Vervolgens onderzoeken we uitgebreid verschillende hybridisatieformuleringen voor hybride benadering van niet-lineaire modellen met behulp van het computationeel beheersbare Max-Min-Plus-Scaling (MMPS) systeemformalisme, evenals niet-lineaire ongelijkheidsbeperkingen zoals voertuigstabiliteit en tijdverzadigingslimieten. In Hoofdstuk 7 wordt dit hybridisatiekader beoordeeld in een voertuigregelbenchmark in verschillende rijscenario's, zoals wrijvingsonzekerheid, waarbij de effectiviteit van hybridisatie in het verbeteren van de computationele snelheid en het verbeteren van de zoektocht naar de globaal optimale oplossing wordt aangetoond. Hoofdstuk 8 behandelt de uitdaging van botsingvermijding in *real-time* met een voertuig met beperkte computationele middelen na het plotseling verschijnen van een statisch obstakel. Meer specifiek, we stellen een geïntegreerde aanpak voor bewegingsplanning die efficiënte *feed-forward* planningssignalen combineert om de *feasibility* problemen van MPC te verminderen. Ten slotte richten we ons in Hoofdstuk 9 op proactieve botsingvermijding in dynamische omgevingen met behulp van een hybride stochastisch MPC-kader voor bewegingsplanning, gevalideerd door middel van rigoureuze simulaties met een voertuigmodel van hoge nauwkeurigheid in IPG CarMaker.

Samenvattend, door het ontwikkelen van nieuwe hybridisatiestrategieën, uitgebreide analytische en numerieke onderzoeken, en het introduceren van beoordelingsbenchmarks, gaat dit proefschrift de uitdagingen aan van het beheersen van ontwijkende manoeuvres in gevaarlijke scenario's. De nieuw ontwikkelde methoden in dit proefschrift worden gevalideerd met behulp van uitgebreide case study's en de effectiviteit van hybridisatie in proactieve reacties op gevaar wordt aangetoond door middel van simulaties van hoge nauwkeurigheid evenals experimenten in de echte wereld. Onze aanbevelingen voor toekomstig werk omvatten het verbeteren van robuustheid, meer gegeneraliseerde en uitgebreide gevoeligheidsanalyse, en het verbeteren van de interpretatie van regeloplossingen om menselijk gedrag te imiteren, wat kan bijdragen aan de acceptatie van geautomatiseerd rijden.

# ACKNOWLEDGMENTS

*“We seek more and more privacy, and we feel more and more alienated when we get it.”*

— Philip Slater, *The Pursuit of Loneliness*

It’s hard to recognize the person I was at the start of my PhD journey. Reflecting on these four turbulent years feels almost surreal, a testament to how transformative the experience has been. Yet, none of us grow in isolation, and I want to take a moment to thank everyone who supported me along the way. Whether you realize it or not, you have contributed to this dissertation, and for that, I will always be grateful.

I would like to express my gratitude to my supervisors, **Bart** and **Simone**, for their guidance, flexibility, and open-mindedness during my research process. I am also thankful to the incredible colleagues I worked with closely. **Barys**, thank you for your support during EVOLVE user committee meetings and for arranging numerous opportunities that allowed me to learn and take on new challenges. **Laura**, **Azita** and **Ton**, I’m deeply grateful for our enriching discussions and brainstorming sessions.

I extend my heartfelt gratitude to my MSc students: **Roald**, your resilience and dedication were unforgettable; I’ll never forget the time we discretized a system into two million pieces! **Jelske**, your scientific curiosity and critical thinking are inspiring. **Pietro**, your attention to detail and perseverance in a new environment were remarkable. **Kenrick**, your determination and tireless effort to succeed left a lasting impression on me. During my dark days, meeting you all gave me the energy I needed to move forward and I’m proud of the incredible scientists you are today.

I’ll forever cherish the wonderful people I met and collaborated with during my research visit in the University of Tokyo. **Fujimoto-sensei**, I deeply appreciate the opportunity to visit your lab. **Binh-Minh-sensei**, working with you was a valuable experience, and spending Nowruz together made it even more special. Thank you **Hosomi-san** for offering help in every step of the way and **SaTona-san** for being the coolest driver ever! To **Mihoko-san**, **Hedaiko-san**, and the sweetest, **Michi-chan**, you all hold a special place in my heart and your warmth and kindness will always stay with me.

To my wonderful officemates—**Frederik**, **Max**, and **David**—thank you for being such a great support system. From your friendly greetings to sharing fun information, I enjoyed every moment with you. I appreciate how you always checked in on me and offered help when I needed it. Our office was a safe space, because of you.

A huge thank you to my other colleagues, the amazing crew of X: **Nuno**, **Shamangi**, **Olga**, **Neil**, **Kjeld**, and **Henri**. **Nuno**, your energy was contagious, and many dark days turned brighter with just an hour of spinning with you. **Shamangi**, your open-mindedness, courage, and authenticity are truly inspiring. **Neil**, thank you for occasionally reminding

me which leg was left and which was right in the dance class! **Olga**, I'll always treasure the wonderful conversations we shared. My gratitude also goes to **Deborah**, **Ismail**, **Simona**, **Kiarash** and everyone else who brought joy, warmth, and smiles to our shared experiences at X.

Communities thrive when we live by *praxis*—taking our values and ideals and putting them into action to create positive change. I am incredibly thankful for the people who contribute to this process with intention and care, making the world a better place for all of us. **Magda**, your unwavering support is invaluable. **Giedo**, I appreciate all the effort you put into improving the well-being of employees. **Mascha**, your thoughtful care for the PhDs, even when you are overwhelmingly busy, does not go unnoticed and is deeply appreciated. **Bo**, your dedication and kindness in your daily work are truly priceless. You have no idea how many tough days were turned around by simply interacting with you. **Ailishia** and **Mimi**, I hope you know how inspiring your projects are. **Kirsten**, thank you for your compassionate approach to your medical profession. **Mijntje** and **Nick**, thank you for your meaningful support and care in your practice. **Alba** and **Camilo**, I'm deeply grateful for the moments we shared. Your mindful, self-aware approach to art inspired me to grow and deepen my understanding of myself. And **Bart**, I would like to thank you again, this time as our department chair. Your management in critical times has made important and meaningful impact.

To **Emilio**, thank you for helping me find my first electric guitar and for our fun moments of non-verbal communication. **João**, your kindness radiates warmth and calmness, making every moment special. Thank you **Mahendra**, for your tasty curries and being up for any kind of fun activity. **Amin**, the reels you share are legendary, please keep on sending more! **Vicente**, “you’re the best!” I’m grateful for every trip we’ve taken together and the meaningful discussions we’ve had. **Changrui**, thank you for making complex derivations surprisingly fun. **Hasti**, **Mohammad**, **Fatemeh**, **Saman**, and **Nima**, thank you for the wonderful times we’ve shared together. You’ve all contributed something special to my journey, and I’m grateful for each of you.

To my other friends and colleagues—**Shengling**, **Anil**, **Lorenzo**, **Suad**, **Xiaoyu**, **Steven**, **Athina**, **Jan**, **Maolong**, **Arian**, **Kiana**, **Gianpietro**, **Tian**, **Dana**, **Micah**, **Kanghui**, **Sasan**, **Maria**, **Lotfi**, **Alessandro**, **Leonore**, **Ayda**, **Rayyan**, **Francesco**, **Sreeshma**, **Alberto**, **Ali**, **Caio**, **Maria**, and **Reza**—I owe a heartfelt thanks. Your presence made this difficult journey not just survivable, but meaningful.

I would also like to thank my parents, **Houra**, who inspires me to be courageous, and **Ali**, who motivates me pursue deeper understanding. I am also grateful to my brother, **Mahdi**, whose patience and commitment remind me to stay true to myself. Regardless of the distance between us, I love you all endlessly.

In the end, a special thank you to **Eva** for her mindfulness and bravery, and **Mohamad** for his inner strength and deep sense of integrity. I am so grateful for you both, and words cannot express how much I appreciate your presence in my life.

Leila  
Delft, January 2025

# 1

## INTRODUCTION

*As the saying goes, 'he who has a hammer sees everything as a nail'. If you approach a problem from a particular theoretical point of view, you will end up asking only certain questions and answering them in particular ways. You might be lucky, and the problem you are facing might be a 'nail' for which your 'hammer' is the most appropriate tool. But, more often than not, you will need to have an array of tools available to you.*

— Ha-Joon Chang, *Economics: The User's Guide*

Imagine the following scenario: you are driving on a highway, and the driver in front of you suddenly starts making erratic movements, possibly due to an unforeseen situation such as a stroke or even a collision. Another example is the unexpected appearance of an animal on the road, which is closely related to the well-known moose test in the context of automated driving. In such hazardous scenarios, an ideal driver would remain calm, maintain decision-making ability, and be able to perform a safe maneuver, balancing between steering and braking to avoid collision with the obstacle, as well as vehicle instability. In reality, however, drivers often panic, which leads to loss of focus and degraded performance. Automated driving systems can mitigate such risks, which is the core of the research in this thesis. To begin, it is important to define what constitutes a hazard.

### 1.1 THE CONCEPT OF HAZARD IN AUTOMATED DRIVING

Let us first distinguish between highway and urban driving scenarios, as each have their respective challenges and necessitate different research approaches. Urban driving involves navigating through complex traffic patterns, interacting with vulnerable participants such as pedestrians and cyclists, and managing frequent stops and starts. The frequent presence of traffic signals, crosswalks, and the need for constant vigilance for unexpected movements by pedestrians or cyclists add layers of difficulty that require comprehensive studies on human behavior, traffic flow, and safety measures [1–3]. These complexities demand a high level of situational awareness and quick decision-making from the driver.

In contrast, on highways the environment and the behavior of other road users are less complex. For instance, there are no junctions nor red lights, and other vehicles have fewer

behaviors such as decelerating, accelerating, or changing lanes. This reduced complexity might suggest that hazardous scenarios are less dangerous on highways than in urban areas. However, the opposite is true! In fact, highway driving involves its own intricate set of challenges, as evidenced by driver training programs where trainees must master specific urban driving scenarios before progressing to high-speed highway driving. For instance, in some countries e.g. Iran, new drivers are not permitted to drive on inter-city highways during their first year after obtaining their license.

On highways, the primary challenges stem from the high speeds, which significantly amplify the consequences of errors. As a slight disturbance in steering at high speeds can lead to severe instability and loss of control, the **margin for error is drastically reduced** and the **reaction time available to drivers is much shorter** than in urban driving scenarios. Moreover, **maneuvers options are limited** on highways; in urban environments, drivers can often stop completely to avoid obstacles, e.g. in case pedestrians or cyclists suddenly enter their path. However, on highways, it is either dangerous or even impossible to do the same due to the high speeds. For instance, if an animal suddenly appears on the road, the driver might not be able to stop in time or change lanes safely, potentially leading to a serious accident.

In such scenarios, human drivers often experience severe anxiety and impaired judgment, which can lead to suboptimal, if not drastic and dangerous, reactions. In contrast, an automated system does not panic or experience delays in decision-making. Therefore, automated driving can be a promising solution for human-based errors in hazardous scenarios, among other benefits e.g. enhanced mobility.

The concept of hazard plays a significant role in understanding the barrier towards high and full automation levels [4] defined by the Society of Automotive Engineers (SAE) [5]. The SAE classifies automation into six levels shown in Fig. 1.1, ranging from 0 to 5, with levels 4 and 5 respectively representing high and full autonomy. The term “automated driving” – in particular, Automated Driving Systems – is used to refer to levels 3 to 5 [5].

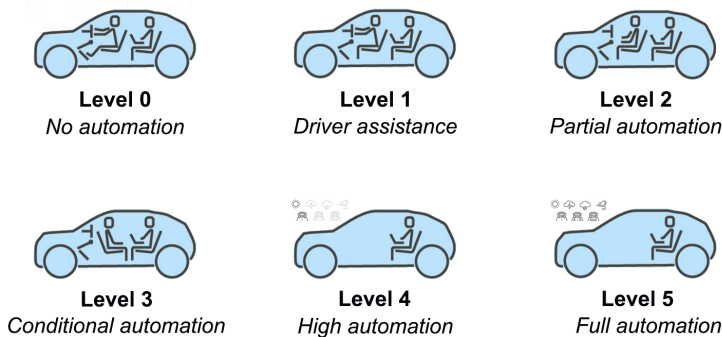


Figure 1.1: Vehicle automation levels defined by SAE, inspired by [6]

In level 3 of autonomy, similar to autopilot systems in airplanes, the automated system is responsible for perception and executing driving tasks [4]. However, the presence of a skilled driver in the vehicle is still required because the driver must take over the driving



task in emergencies [7]. This requirement is eliminated in level 4, meaning that the system must handle emergencies independently in specific operational design domains e.g. sunny weather and particular routes. Finally, a level 5 automated system functions as a skilled driver, and is capable of operating in a wide range of scenarios and various operational design domains without any human intervention. Thus, the primary barrier to reaching high and full automation levels in automated driving lies in the implementation of a driving system capable of handling hazards autonomously. This thesis will focus on these critical scenarios via a proactive approach, i.e. anticipating potential issues rather than simply reacting to them. We provide a detailed explanation of the concept of proactivity in the following section.

## 1.2 PROACTIVITY IN HAZARDOUS SCENARIOS

One of the main challenges in addressing hazards in the design of automated driving systems is the **lack of sufficient and reliable data** in such scenarios, which limits effective employment of data-driven and learning-based control design solutions. Moreover, a **rapidly changing environment** and **uncertain driving conditions** in real-world scenarios require high levels of adaptivity to change and robustness to disturbances for candidate solutions. Model Predictive Control (MPC), as evidenced by its increasing popularity in the field [8–12], is a promising approach for collision avoidance in hazardous scenarios, by systematically handling constraints within an online control optimization problem. As the name suggests, in MPC approach, a model of the system is used to predict its behavior over a finite horizon in the future time steps, which is then incorporated into the formulation of the control optimization problem.

The effective application of MPC for high levels of automated driving necessitates the use of **complex physics-based models** as many simplifying assumptions, such as low lateral accelerations or linear tire dynamics [13], are no longer valid when performing evasive maneuvers in high-speed hazardous scenarios. In this sense, the MPC optimization problem in hazardous scenarios may lead to long computation times, which is a critical limitation in these situations where **swift decision-making** is essential.

*A proactive control solution respects the entanglement of the aforementioned challenges by providing computationally fast and optimally safe trajectories and/or control braking and steering commands in response to an unforeseen dangerous situation.*

For instance, one may think of robust [14] and tube-based [15] MPC design approaches to tackle the uncertainty challenge. However, a robust approach has a limited ability to providing a safe solution in case of an unforeseen incident on the road. Moreover, in a hazardous scenario an overly-cautious and conservative control solution is in general not optimal and may even lead to the propagation of hazard in time. For example, braking behind a car with an unconscious driver keeps the ego vehicle in danger for a long period, while overtaking can help the system get out of the hazardous scenario as soon as possible. In this sense, a stochastic formulation of the uncertainties allows for a more proactive response to a hazardous situation on the road. In addition, even the most optimal overtaking trajectory becomes ineffective if it is computed too slowly, missing the critical window

of opportunity to avoid a collision. In summary, designing a proactive control strategy for automated driving requires consideration of the stochastic properties of the MPC optimization problem, alongside the need for computational efficiency. As the latter is the bottleneck in the field, the following section will focus on this aspect in greater depth.

### 1.3 APPROXIMATION: A DOUBLE-EDGE SWORD

The nonlinearity of the MPC optimization problem is the main source of long computation times. For instance, while solving a NonLinear Program (NLP) in general yield a local optimum, state-of-the-art solution methods for a Mixed-Integer Linear Program (MILP) can guarantee convergence to the global optimum.

Linearization can effectively represent the local behavior of the system, but often fails to capture critical nonlinear phenomena such as saturation of the tire forces. As a result, avoiding the nonlinearity by e.g. successive online linearization [16] or grid-based linearization [17] of the nonlinearity in hazardous scenarios does not allow capturing the full dynamics of systems such as tires operating in nonlinear regions, where the interaction between longitudinal and lateral dynamics further complicates the behavior.

With these limitations of linearization-based techniques for our research problem, it is evident that a sophisticated approximation approach is required for the vehicle control problem in hazardous scenarios. This approximation must be meticulously defined within a comprehensive framework to ensure a well-balanced trade-off between computation time and accuracy, allowing for informed decision-making in the control design process. In this context, hybridization emerges as a particularly promising approach.

Hybridization involves approximating nonlinearities using hybrid systems modeling frameworks [18]. While the literature provides examples of this hybridization for computational efficiency [19–21], developing a comprehensive framework necessitates a more thorough analysis and synthesis. This development is achieved by analysis of various approximation methods and parameters while keeping an eye on the dynamic nature of the problem and its critical aspects including the optimization problem and its number of decision variables, specific application requirements, numerical stability, and computational performance analysis, especially in systems with multi-dimensional domains. In line with our primary research objective to propose a proactive control solution for automated driving on highways, this thesis includes the examination of hybridization techniques for automated driving, as well as assessment of their impact on the proactivity of the resulting MPC formulation.

### 1.4 CONTRIBUTIONS AND ORGANIZATION OF THIS THESIS

This thesis addresses the control of evasive maneuvers in hazardous scenarios. In summary, the contributions of the thesis are

- presenting novel iterative cut-based PieceWise Affine (PWA) approximation approaches for a flexible approximation of multi-dimensional nonlinear systems while ensuring adherence to user-defined error tolerances (Chapters 2 and 3),
- introducing H4MPC, a MATLAB-based hybridization toolbox, offering a user-friendly

interface to formulate and solve optimization problems for approximating nonlinearities for MPC optimization problems (Chapter 4),

- analyzing the sensitivity of PWA approximations to obtain quantitative bounds on the distance between the original and the approximated local minima of a NLP with polytopic constraints (Chapter 5),
- developing a novel hybridization method for MPC optimization problems using the Max-Min-Plus-Scaling (MMPS) formalism along with a comparative assessment benchmark to evaluate the resulting computational performance (Chapters 6 and 7),
- investigating real-time implementation of MPC for emergency collision avoidance after sudden appearance of a static obstacle with extensive discussion of experimental insights (Chapter 8), and
- proposing a proactive Stochastic Model Predictive Control (SMPC) approach for collision avoidance in highway scenarios (Chapter 9).

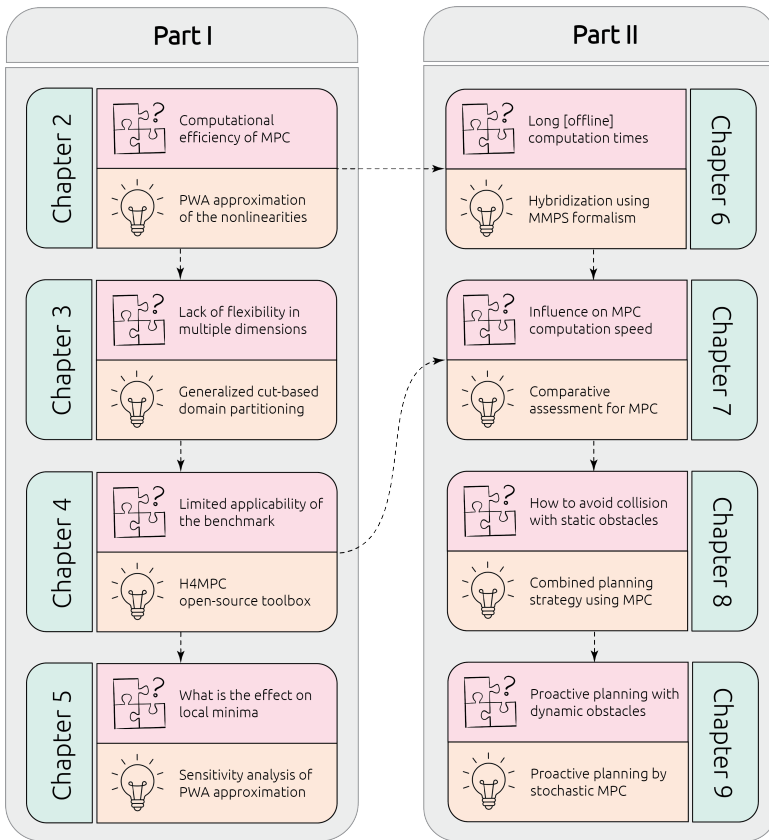


Figure 1.2: Structure of this thesis

The structure of this thesis is designed to provide a comprehensive exploration and application of hybridization techniques, as well as applying the developed frameworks to the control problem at hand by categorizing the chapters in two parts. In Part I, we focus on the concept of hybridization, providing solid theoretical foundation and a clear understanding of how hybridization can be effectively implemented. In Part II, we shift our focus to the practical application of these hybridization frameworks in various aspects of the control problem in hazardous scenarios to demonstrate how the theoretical principles discussed in Part I can be translated into practical solutions, addressing real-world challenges in automated driving.

Fig. 1.2 illustrates the thesis structure. Each chapter proposes solutions to specific questions within our main research problem. The connection between different chapters is indicated by dashed arrowhead lines.

# I

## HYBRIDIZATION APPROACH TO COMPUTATIONAL EFFICIENCY



## 2

# ITERATIVE PWA APPROXIMATION VIA CUT-BASED DOMAIN PARTITIONING

*“Science is not a theory of reality, but a method of inquiry.”*

— Donald D. Hoffman, *The Case Against Reality*

*PieceWise Affine (PWA) approximations are widely used among hybrid modeling frameworks as a way to increase computational efficiency in nonlinear control and optimization problems. A variety of approaches to construct PWA approximations have been proposed, most of which are tailored to specific application areas by using some prior knowledge of the system in their assumptions and/or steps. In this chapter, a parametric method is proposed to identify PWA approximations of nonlinear systems, without any prior knowledge of their dynamics or application requirements. The algorithm defines the regions parametrically using hyperplanes to cut the domain, and increases the number of regions iteratively until a user-defined error tolerance criterion is met. General remarks are given on the algorithm’s implementation and a case study is provided to illustrate its application to vehicle dynamics.*

## 2.1 INTRODUCTION

The literature on hybrid systems provides analysis and control synthesis methodology for systems featuring interacting continuous and discrete dynamics. To do so, a variety of modeling frameworks have been proposed for hybrid systems, as well as proof of their equivalence [23], such as PWA, Mixed-Logical-Dynamical (MLD), Max-Min-Plus-Scaling (MMPS), and linear complementary systems [18]. Several of these frameworks have been extensively studied, including for control [24] and reachability analysis [25] of MMPS

This chapter has been published in IFAC-PapersOnLine, the proceedings of IFAC World Congress [22].

systems, model predictive control design [26] and its explicit solution in some cases [27] for MLD, as well as for continuous PWA systems [28].

Among all the hybrid modeling frameworks, PWA systems have received extensive attention due to their simple, yet clear, formulation of the hybrid nature of the system behavior (i.e. explicit representation of different dynamics and their activation criteria). For example, the performance of discrete-time PWA systems [29], their stability criteria in presence of uncertainty [30], their periodic solutions [31], and bifurcation phenomena [32] were analyzed.

The PWA formalism is not only applied in domains where the hybrid nature of the system is important, but it has also been extensively utilized in a wide range of problems to increase computational efficiency, such as modeling prostate-specific antigen levels [33], water motion in sewer networks [34], or cornering behavior in vehicles [35]. In some cases, PWA approximation of a nonlinear model facilitates reduction of the nonlinear control optimization problem into a mixed-integer programming one, while still capturing the complexity of the nonlinear behavior.

There are two main aspects to the problem of finding a PWA approximation: optimal partitioning of the state space into regions, and finding the optimal affine approximation in each one. The shape and the number of the regions influence computational complexity, the accuracy, and potential numerical issues of the final form. A higher number of regions improves accuracy and reduces the error bound, but leads to computationally more complex control problems. In addition, the shape and edge of the regions are of importance as the optimization problem is most likely to encounter numerical problems, if e.g. regions have redundant edges or gaps exist between them.

In some applications, a proper partitioning strategy is known based on heuristics or physics-based knowledge of the system [20, 36]. In such cases, finding local affine approximations is more straightforward and can be achieved using least-squares or other regression methods. However, a generic PWA-approximation optimization problem is combined, i.e. both regions and local approximations are decision variables. Some techniques have been proposed to tackle challenges due to the combined nature of the problem, like partitioning the domain based on the variations of the nonlinear function [37], learning-based PWA system identification using recursive adaptive control laws [38] and online observer-based identifiers [39], or clustering approaches, either based on convex relaxation of sparse optimization problems [40] or incorporating fuzzy-based outlier rejection and k-means method [41].

To date, many of the developed techniques, either explicitly or implicitly, limit the application to low dimensions or a bound on the number of local dynamics/modes [42], and many require some prior knowledge of the PWA approximation to be found. e.g. by employing some heuristic clustering steps [43]. Evidently, the effectiveness of the method depends upon the application area and its requirements; the cited papers have successfully found computationally efficient PWA models for their respective systems. However, to the best of our knowledge, no method has been proposed that addresses generic PWA approximation of a system, without taking specific dimensions, applications, or assumptions into account.

In this chapter, we propose a novel iterative algorithm to find PWA approximations of nonlinear systems satisfying a user-defined error tolerance. Our proposed approach solves



combined optimization problems in each iteration where parametric hyperplanes are used to cut the domain into different regions. This results in parametric definition of regions, which are then directly optimized as a subset of the decision variables. As the algorithm assumes no prior knowledge of the system, it can be implemented for discrete-time and continuous-time dynamics, as well as event-driven and time-driven dynamics, in a wide range of application areas. In any case, the algorithm can still be simplified, curtailed, or easily modified if any information on the system is available. Details of the algorithm and parametric region definition are described in Section 2.2, accompanied by general remarks on various steps and considerations. The algorithm is then tested using a nonlinear vehicle model as a case study in Section 2.3. Finally, concluding remarks and suggestions for future work are given in Section 2.4.

## 2.2 PWA APPROXIMATION

### 2.2.1 PROBLEM FORMULATION

Consider a given nonlinear system with its dynamics expressed in the generic form  $\dot{s} = F(s, u)$ , where  $s \in \mathbb{R}^n$  and  $u \in \mathbb{R}^m$  respectively represent the state and input vectors and  $F : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^n$  is the nonlinear function to be approximated. Without loss of generality, the augmented state vector  $x = [s^T u^T]^T$  is used to define  $F(x) := F(s, u)$  since the approximated function will be selected to be affine in both the state and the input. The augmented domain is assumed to be bounded and will be defined as  $\text{dom}(F) = \mathcal{D} \subset \mathbb{R}^{m+n}$ .

The proposed approach approximates the nonlinear function  $F$  by a PWA function  $f$  defined as

$$x \in C_p \implies f(x) = f_p(x), \quad f_p(x) = A_p x + B_p, \quad (2.1)$$

with  $p \in \{1, 2, \dots, P\}$ , where  $P$  is the number of regions, each defined by polytope  $C_p \subseteq \mathbb{R}^{m+n}$  with

$$C_p \neq \emptyset, \quad \forall p \in \{1, \dots, P\}, \quad (2.2a)$$

$$\text{int}(C_p) \cap \text{int}(C_q) = \emptyset, \quad \forall p, q \in \{1, \dots, P\}, p \neq q, \quad (2.2b)$$

$$\bigcup_{p=1}^P C_p = \mathcal{D}, \quad \forall p, q \in \{1, \dots, P\}, p \neq q, \quad (2.2c)$$

to form a partition of  $\mathcal{D}$ , with  $\text{int}(C_p)$  denoting the interior of region  $C_p$ . By defining the border hyperplanes  $L_{p,q} \subset \mathbb{R}^{m+n-1}$  as

$$L_{p,q} = C_p \cap C_q, \quad \forall p, q \in \{1, 2, \dots, P\}, p \neq q, \quad (2.3)$$

the set of border hyperplanes forming boundaries of the region  $C_p$  are represented by the set

$$\mathcal{L}_p = \{L_{p,q} \mid q \in \{1, \dots, P\} \wedge q \neq p\}.$$

For a fixed  $P$ , both the regions  $C_p$  and the corresponding local affine approximations  $f_p$  are obtained by finding the optimal values of the matrices  $A_p$  and  $B_p$ , as well as the set  $\mathcal{L}_p$  so as to minimize the squared approximation error. This is implemented by solving the

optimization problem

$$\min_{A_p \in \mathcal{A}, B_p \in \mathcal{B}, \mathcal{L}_p \in \mathcal{L}} \int_D \frac{\|F(x) - f(x)\|_2^2}{\|F(x)\|_2^2 + \epsilon} dx, \quad \text{s.t. (2.1)–(2.2b),} \quad (2.4)$$

where  $\mathcal{A}$ ,  $\mathcal{B}$ , and  $\mathcal{L}$  represent the sets containing  $A_p$ ,  $B_p$ , and  $\mathcal{L}_p$ , respectively. The term  $\|F(x)\|_2^2$  in the denominator is introduced such that the cost values represent the relative error and the added scalar  $\epsilon > 0$  prevents division by very small values where  $\|F(x)\|_2 \approx 0$ .

### 2.2.2 PARAMETRIC DEFINITION OF REGIONS

Without loss of generality,  $m+n$  is assumed to be an even number as  $m+n = 2d$ , and the states are paired in couples as  $(x_i, x_j)$  to form 2-dimensional subspaces. The corresponding pairs  $(i, j)$  are collected in the set  $\Omega$  and the local domains  $\mathcal{D}_{i,j} \subset \mathbb{R}^2$ , are defined as

$$x \in \mathcal{D} \implies \begin{bmatrix} x_i & x_j \end{bmatrix}^T \in \mathcal{D}_{i,j}, \quad \forall (i, j) \in \Omega.$$

**Remark 2.1.** It should be noted that this assumption will not pose any restrictions on the method since for an odd  $(m+n)$  value, the cutting procedure can be easily implemented on the unpaired single dimension as an axis.

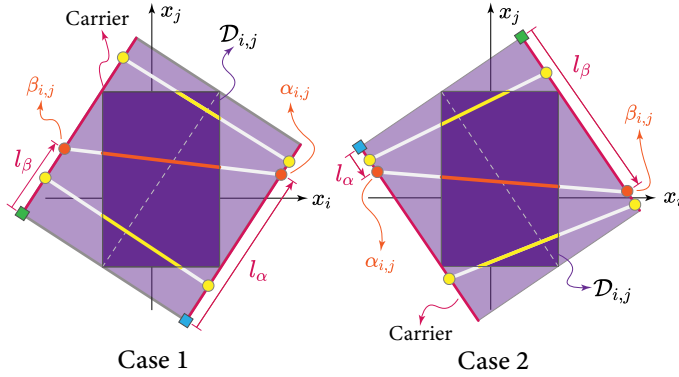


Figure 2.1: Parametric definition of cutting the domain; two cases are proposed to cover all the cutting angles within the local domain.

After pairing the states, the regions are defined by cutting  $\mathcal{D}$  perpendicular to the  $(x_i, x_j)$  planes as shown in Fig. 2.1. Since the region boundaries are to be optimized, the place of the cuts needs to be defined parametrically. To do so, two carrier lines are introduced on opposite sides of  $\mathcal{D}_{i,j}$ , on which points  $\alpha_{i,j}$  and  $\beta_{i,j}$  can slide. As an example, Fig. 2.1 shows three points (in yellow and orange) sliding on the carriers, where the lines connecting the pairs of  $(\alpha_{i,j}, \beta_{i,j})$  are used to cut the domain  $\mathcal{D}$  perpendicular to  $\mathcal{D}_{i,j}$ .

**Remark 2.2.** The  $\alpha_{i,j}$  values should be increasing and the same holds for the  $\beta_{i,j}$  values, since otherwise the corresponding cuts collide.

**Remark 2.3.** Given  $l_\alpha$  and  $l_\beta$ , the location of the points  $\alpha_{i,j}$  and  $\beta_{i,j}$  on carriers parallel to the diagonal of  $\mathcal{D}$  can be obtained as

$$\begin{aligned} x_{i\alpha} &= x_{i\min} + X_i \sin^2 \phi + l_\alpha \cos \phi, \\ x_{j\alpha} &= x_{j\min} - X_i \sin \phi \cos \phi + l_\alpha \sin \phi, \\ x_{i\beta} &= x_{i\min} - X_j \sin \phi \cos \phi + l_\beta \cos \phi, \\ x_{j\beta} &= x_{j\min} + X_j \cos^2 \phi \cos \phi + l_\beta \sin \phi, \end{aligned} \quad (2.5)$$

where the domain parameters  $X$ ,  $\phi$ , and  $x_{\min}$  associated with the  $i$  and  $j$  axes are shown in Fig. 2.2.

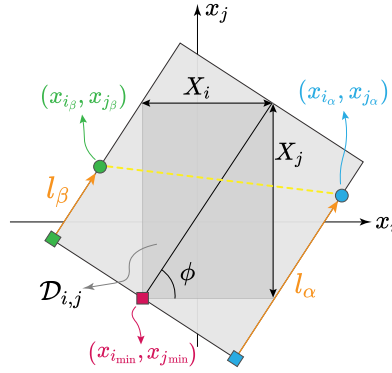


Figure 2.2: A schematic view of the connection among various domain parameters and their relation to the decision variables, i.e. domain cuts.

To cover all possible cutting angles, the two cases in Fig. 2.1 should be investigated separately with different carriers. Note that requiring two cases stems from the  $(x_i, x_j)$  plane being 2-dimensional. For a rectangular  $\mathcal{D}_{i,j}$  (e.g. due to bound constraints), or a parallelogram, it is convenient to define the carriers for  $\alpha_{i,j}$  and  $\beta_{i,j}$  points parallel to one of the diagonals in each case. Nevertheless, this concept can be easily extended for applications with other  $\mathcal{D}_{i,j}$  forms by circumscribing a parallelogram to  $\mathcal{D}_{i,j}$  and defining the carriers parallel to the diagonals for each case. The numbers of cuts perpendicular to each  $(x_i, x_j)$  plane, is denoted by  $n_{c_{i,j}}$  and it is equal to the number of  $\alpha_{i,j}$  and  $\beta_{i,j}$  points sliding on the carriers.

### 2.2.3 APPROXIMATION ALGORITHM

As the optimal number of regions is not known a priori, our proposed algorithm tackles PWA approximation through an iterative loop given in Algorithm 1. The vector  $n_c$  contains the number of cuts  $n_{c_{i,j}}$  with indices  $i$  and  $j$  such that  $(x_i, x_j) \in \Omega$ . By getting  $n_c$  as input and solving (2.4) for a fixed  $P$  as

$$P = \prod_{(i,j) \in \Omega} (n_{c_{i,j}} + 1),$$

the function  $\text{reg\_optimization}(\cdot)$  finds the optimal affine approximations and their corresponding regions simultaneously. During each iteration,  $\text{reg\_optimization}(\cdot)$  returns both the minimum objective  $J^*$  and its corresponding optimal decision variables

$$v^* = (\mathcal{A}^*, \mathcal{B}^*, \mathcal{L}^*),$$

as output. The asterisk indicates the optimal value of the variable, and the border hyper-planes are defined using the position of the  $\alpha_{i,j}$  and  $\beta_{i,j}$  points as

$$\mathcal{L} = \{(l_{\alpha_k}, l_{\beta_k}) \mid \kappa \in \{1, 2, \dots, d\}, k \in \{1, 2, \dots, (n_c)_\kappa\}\}.$$

It should be noted that (2.4) is a nonlinear optimization problem. Therefore, the function  $\text{reg\_optimization}(\cdot)$  can either use a global search solver such as genetic algorithm or particle swarm, or gradient-based approaches with multiple starting points. In both cases, the best objective value would be the lowest value among the minima obtained in each trial.

---

**Algorithm 1** Iterative cut-based PWA approximation

---

```

cond ← true
 $n_c \leftarrow 0_{1 \times d}$ 
iter ← 0
while cond do
    iter ← iter + 1
     $n_{\text{cuts}} \leftarrow 1_{d \times 1} \times n_c + I_{d \times d}$ 
    for  $i \in \{1, 2, \dots, d\}$  do
         $n_{\text{in}} \leftarrow d\text{-th row of } n_{\text{cuts}}$ 
         $[\text{err}(i), \text{sol}(i)] \leftarrow \text{reg\_optimization}(n_{\text{in}})$ 
    end for
     $d_{\text{best}} \leftarrow \arg \min_i (\text{err}(i))$ 
     $n_c \leftarrow d_{\text{best}}\text{-th row of } n_{\text{cuts}}$ 
    if  $\min(\text{err}) \leq \text{tol}_{\text{err}}$  then
        cond ← false
        return  $n_c, \text{sol}(d_{\text{best}})$ 
    else if  $\prod_{q=1}^d n_c(q) \geq N_{\text{reg\_max}}$  then
        cond ← false
        return print('Exceeded  $N_{\text{reg}}$ ')
    else if iter  $\geq \text{iter}_{\text{max}}$  then
        cond ← false
        return print('Exceeded  $\text{iter}_{\text{max}}$ ')
    end if
end while

```

---

However, the number of regions may not be sufficient to approximate the nonlinear function within a particular error bound. In that case, more cuts should be introduced to partition  $\mathcal{D}$ . To do so, the designed loops runs as follows to investigate different scenarios: in each iteration,  $\text{reg\_optimization}(\cdot)$  is solved for  $d$  cases, in which only one element in

$n_c$  is increased by 1, and the  $n_c$  with the lowest objective is selected as the best cutting strategy for the next iteration. The algorithm stops when reaching objective values below the error tolerance  $\text{tol}_{\text{err}}$ . To avoid an infinite loop, the procedure can also be stopped by passing maximum bounds on the number of iterations or the number of regions.

**Example 2.1.** For  $\mathcal{D} \subset \mathbb{R}^4$ ,  $d = 2$ , and  $\Omega = \{(1,2), (3,4)\}$ , the algorithm starts by setting  $n_c = \begin{bmatrix} 0 & 0 \end{bmatrix}$ , which means no cutting.

In the first iteration,  $\text{reg\_optimization}(\cdot)$  is called twice, finding the best approximations for  $n_c = \begin{bmatrix} 1 & 0 \end{bmatrix}$  and  $n_c = \begin{bmatrix} 0 & 1 \end{bmatrix}$  which correspond respectively to making only one cut perpendicular to  $\mathcal{D}_{1,2}$ , and only one cut perpendicular to  $\mathcal{D}_{3,4}$ .

If  $n_c = \begin{bmatrix} 1 & 0 \end{bmatrix}$  gives a lower objective, but fails to satisfy the error tolerance, the next iteration starts with  $n_c = \begin{bmatrix} 1 & 0 \end{bmatrix}$ , and two cases  $n_c = \begin{bmatrix} 2 & 0 \end{bmatrix}$  and  $n_c = \begin{bmatrix} 1 & 1 \end{bmatrix}$  are investigated. In other words, if one cut on  $\mathcal{D}_{1,2}$  is considered a successful cutting strategy, the next step is to improve the result by adding more cuts to it as a baseline.

### 2.2.4 GENERAL REMARKS

The power of Algorithm 1 stems from neither posing limits on system dimensions nor assuming a required number of regions. Moreover, as the approximation problem (2.4) can be solved by gridding the domain, our proposed method can also be applied in cases where the analytical form of the nonlinear model is not available. For instance, training measurement data sets can also be used to find a fitted PWA approximation using Algorithm 1. In addition, some general notes should be made:

**State-based vs. full-state partitioning:** PWA approximation of  $F : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^n$  is done by running Algorithm 1 independently for each of the  $n$  states. This leads to the cuts and subsequently regions that are independently defined and evaluated for each component of  $\dot{s}$ . If this is not convenient for certain applications and it is desired to have the same regions for all the elements of  $\dot{s}$ , Algorithm 1 can be used in the same fashion or modified by changing the objective in (2.4) as

$$\min_{A_p \in \mathcal{A}, B_p \in \mathcal{B}, L_p \in \mathcal{L}} \int_{\mathcal{D}} \frac{\|W(F(x) - f(x))\|_2^2 + 1}{\|F(x)\|_2^2} dx, \quad \text{s.t. (2.1)–(2.2b),}$$

where  $W$  is a weight matrix.

**Selection of a single case:** Implementing the proposed approach is completed by running Algorithm 1 for cases 1 and 2 in parallel and choosing the best result. However, one of the two cases may always be showing better results from the first iteration. To avoid unnecessary computation in such instances, the cases can be tested and compared by running the first iteration of Algorithm 1, identifying the better case (i.e. with a lower objective), and then implementing Algorithm 1 only for that case.

**Pairing the states:** Pairing the states as  $\Omega$  can be done arbitrarily. Prior knowledge of the system and/or its application may suggest that specific states should be paired. Nevertheless, the pairing can be also done by testing different combinations of  $\Omega$  through one iteration, as was proposed for evaluating cases 1 and 2.

**Cases with unbounded domain:** The proposed algorithm assumes the domains ( $\mathcal{D}$  and subsequently  $\mathcal{D}_{i,j}$ ) to be bounded. In case of an unbounded domain, a subset of the regions  $\mathcal{C}_p$  need to be defined unbounded as well. This will not affect the decision variables in (2.4) as the cutting planes are optimized, not the regions' boundaries. However, the objective in (2.4) approaches infinity across an unbounded domain. To avoid this, a sufficiently large bounded subset of the unbounded domain  $\mathcal{D}$  can be used to find the PWA approximation using our algorithm. The result can then be directly used to approximate the behavior in the original domain.

**Obtaining the cutting hyperplanes:** The matrix form of the border hyperplanes obtained from (2.3) can be constructed by extending the definition of the cuts. Using (2.5), a cut  $L$  is defined by

$$L := x_j = \left( \frac{x_{j\alpha} - x_{j\beta}}{x_{i\alpha} - x_{i\beta}} \right) x_i + \left( x_{j\alpha} - x_{i\alpha} \frac{x_{j\alpha} - x_{j\beta}}{x_{i\alpha} - x_{i\beta}} \right).$$

As each pair of cuts from different  $\mathcal{D}_{i,j}$  are perpendicular, the resulting cutting hyperplanes in  $\mathcal{D}$  can be directly combined in a generic matrix form

$$L_{p,q} := Hx + h = 0.$$

## 2.3 CASE STUDY: VEHICLE DYNAMICS

In this section, Algorithm 1 is used to find a PWA approximation of a nonlinear model of vehicle dynamics, integrating the coupled longitudinal and lateral dynamics in a single-track configuration, and considering linear tire forces. The model and implementation of the proposed approach are explained in the following sections.

### 2.3.1 NONLINEAR VEHICLE MODEL

A single-track representation of the vehicle is shown in Fig. 2.3. With the system variables and parameters respectively defined in Tables 2.1 and 2.2, the nonlinear vehicle model is described by the following equations:

$$\dot{v}_x = \frac{1}{m} [F_{xf} \cos \delta - F_{yf} \sin \delta + F_{xr}] + v_y r, \quad (2.6a)$$

$$\dot{v}_y = \frac{1}{m} [F_{xf} \sin \delta + F_{yf} \cos \delta + F_{yr}] - v_x r, \quad (2.6b)$$

$$\dot{r} = \frac{1}{I_{zz}} [F_{xf} \sin \delta l_f + F_{yf} \cos \delta l_f - F_{yr} l_r], \quad (2.6c)$$

and the lateral forces are given by the linear tire model

$$F_{yf} = C_{\alpha_f} \alpha_f, \quad F_{yr} = C_{\alpha_r} \alpha_r,$$

where the slip angles are obtained by

$$\alpha_f = \delta - \tan^{-1} \left( \frac{v_y + l_f r}{v_x} \right), \quad \alpha_r = \tan^{-1} \left( \frac{v_y - l_r r}{v_x} \right).$$

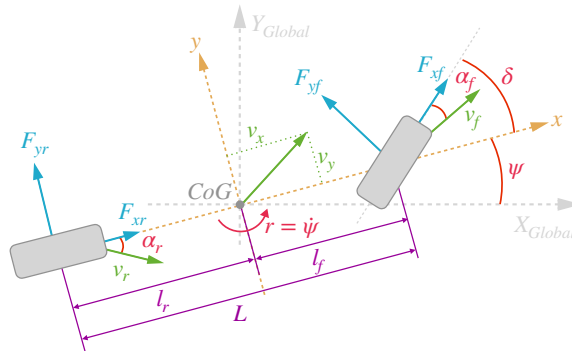


Figure 2.3: Configuration of the single-track vehicle model

Table 2.1: System variables

Variable	Definition	Unit
$v_x$	Longitudinal velocity	m/s
$v_y$	Lateral velocity	m/s
$\psi$	Yaw angle	rad
$r$	Yaw rate	rad/s
$\delta$	Steering angle (road)	rad
$F_{xf}$	Longitudinal force on the front axis	N
$F_{xr}$	Longitudinal force on the rear axis	N
$F_{yf}$	Lateral force on the front axis	N
$F_{yr}$	Lateral force on the rear axis	N
$F_{zf}$	Normal load on the front axis	N
$F_{zr}$	Normal load on the rear axis	N
$\alpha_f$	Front slip angle	rad
$\alpha_r$	Rear slip angle	rad

Table 2.2: System parameters

Parameter	Definition	Value	Unit
$m$	Vehicle mass	1970	kg
$I_{zz}$	Inertia moment about z-axis	3498	kg/m <sup>2</sup>
$l_f$	CoG* to front axis distance	1.4778	m
$l_r$	CoG to rear axis distance	1.4102	m
$C_{\alpha_f}$	Front cornering stiffness	126784	N
$C_{\alpha_r}$	Front cornering stiffness	213983	N

\*Center of Gravity

### 2.3.2 IMPLEMENTATION AND RESULTS

Considering system dynamics in (2.6a) to (2.6c), Algorithm 1 is used to find PWA approximation of  $\dot{v}_x$ ,  $\dot{v}_y$ , and  $\dot{r}$  independently. MATLAB's Optimization toolbox is used to implement the algorithm using lsqnonline for 10 starting points. The system is simulated during an evasive double lane-change maneuver and the axes corresponding to the augmented state

vector  $x = [v_x \ v_y \ r \ F_{xf} \ F_{xr} \ \delta]^T$  are paired as

$$\Omega = \{(v_x, r), (v_y, \delta), (F_{xf}, F_{xr})\},$$

which results from our physics-based knowledge of the system states, their dimensions, and their order of magnitude. Comparing the first iterations of cases 1 and 2 showed that case 2 gives lower objectives when cutting  $\mathcal{D}$  perpendicular to  $\mathcal{D}_{v_x, r}$  and  $\mathcal{D}_{v_y, \delta}$ , while case 1 is the better one to define cuts on  $\mathcal{D}_{F_{xf}, F_{xr}}$ .

2

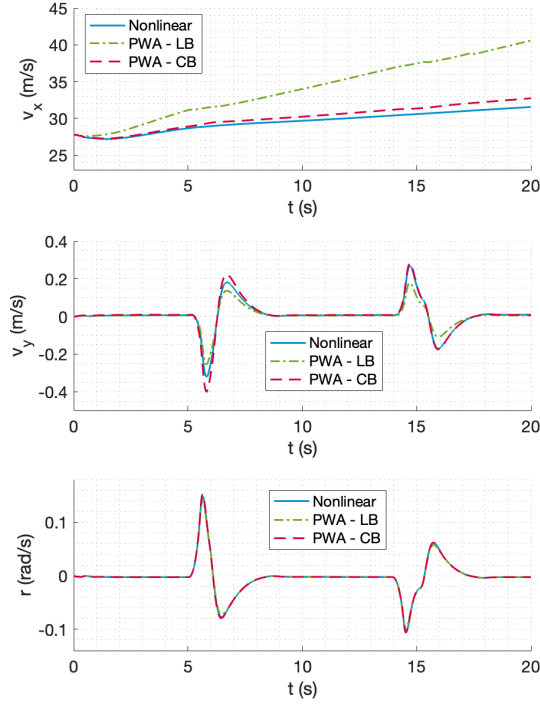


Figure 2.4: Open-loop simulation of an evasive double lane-change maneuver using nonlinear vehicle model and two PWA approximations: LB and CB approaches

The solution time depends on the number of regions due to an subsequent increase in the number of decision variables. The algorithm was run for different error tolerances using the DelftBlue supercomputer [44] with every iteration for the number of regions between 2 to 10 taking on average 435 minutes.

The approximations obtained for  $\text{tol}_{\text{err}}$  values in Table 2.3 using our proposed cut-based algorithm (CB), and the Lebesgue PWA approximation (LB) approach proposed by [37], have been compared with the nonlinear system for the open-loop system simulation in Fig. 2.4. In the LB approach, the domain is partitioned perpendicular to each axis and based on variation of the nonlinear function's gradient; this results in hypercubic regions. However, the CB approach cuts the domain perpendicular to 2-dimensional subspaces which leads to polytopic regions. The same tolerances were selected for both algorithms



for fair comparison, and they converged to the number cuts  $n_c$  defined as

$$n_c = \left[ n_{c(v_{x,r})}, n_{c(v_{y,\delta})}, n_{c(F_{xf}, F_{xt})} \right].$$

The total number of regions  $N$  is listed as well in Table 2.3. Fig. 2.4 shows that the CB approach provides a more accurate approximation of the model, and its good performance is better seen in  $\dot{v}_x$  which has a higher degree of nonlinearity where CB gives a better approximation while introducing a smaller number of regions.

Table 2.3: The number of cuts at convergence for case study instances with different error tolerance values

Instance		$\dot{v}_x$	$\dot{v}_y$	$\dot{r}$
	tol <sub>err</sub>	0.30	0.10	0.05
PWA – LB	$n_c$	[1,3,0]	[0,0,0]	[0,0,0]
	$N$	8	1	1
PWA – CB	$n_c$	[0,3,0]	[1,0,0]	[1,0,0]
	$N$	4	2	2

## 2.4 CONCLUSIONS

In this chapter, an iterative algorithm for PWA approximation of nonlinear systems was proposed assuming no prior knowledge of the application area. By using a cut-based parametric definition of the regions in the optimization problem, the algorithm aims at finding an optimal partitioning of the domain into polytopic regions and the corresponding local affine approximations, simultaneously. This combined optimization problem is solved in each iteration for several cases of adding new cuts whereas the number of cuts is increased in each iteration until a user-specified error tolerance is reached. The algorithm is implemented on a nonlinear vehicle model as a case study where different error tolerances were selected for each state and the results were compared to another PWA approximation approach from the literature, where similar to our proposed algorithm, the regions are included parametrically in the decision variables of the combined optimization problem. The comparison shows that our approach gives more a accurate approximation of the nonlinear system, in some cases with fewer number of regions.

In future work, the current algorithm can be improved along two lines. First, the iteration law can be enhanced for faster convergence to the optimal number of regions while avoiding introduction of extra and/or redundant cuts. For instance, instead of increasing the number of cuts in each iteration by one, more cuts can be introduced based on the difference of the objective functions between the last two iterations. Second, adjustments or additions to the algorithm structure can be introduced for applications where discontinuity is problematic, to either avoid discontinuity on the region borders in the obtained PWA approximation, or to circumvent its undesired consequences (e.g. in switching analysis or control synthesis) by defining auxiliary affine dynamics or switching rules along the borders. Moreover, on the application level we aim at investigating the performance of our proposed approximation method on a wider variety of test cases, i.e. driving scenarios.



## 3

## 3

# PWA APPROXIMATION OF MULTI-DIMENSIONAL NONLINEAR SYSTEMS

*"Behold me - I am a Line, the longest in Lineland, over six inches of Space".*

*"Of Length," I ventured to suggest.*

*"Fool", said he, "Space is Length."*

— Edwin Abbott, *Flatland: A Romance of Many Dimensions*

*PieceWise Affine (PWA) approximations for nonlinear functions have been extensively used for tractable, computationally efficient control of nonlinear systems. However, reaching a desired approximation accuracy without prior information about the behavior of the nonlinear systems remains a challenge in the function approximation and control literature. As the name suggests, PWA approximation aims at approximating a nonlinear function or system by dividing the domain into multiple subregions where the nonlinear function or dynamics is approximated locally by an affine function also called local mode. Without prior knowledge of the form of the nonlinearity, the required number of modes, the locations of the subregions, and the local approximations need to be optimized simultaneously, which becomes highly complex for large-scale systems with multi-dimensional nonlinear functions. This chapter introduces a novel approach for PWA approximation of multi-dimensional nonlinear systems, utilizing a hinging hyperplane formalism for cut-based partitioning of the domain. The complexity of the PWA approximation is iteratively increased until reaching the desired accuracy level. Further, the tractable cut definitions allow for different forms of subregions, as well as the ability to impose continuity constraints on the PWA approximation. The methodology is explained via multiple examples and its performance is compared to two existing approaches through case studies, showcasing its efficacy.*

---

This chapter has been submitted to Automatica.

### 3.1 INTRODUCTION

PWA systems are a class of hybrid modeling frameworks where the dynamics is expressed by multiple subsystems, i.e. local modes, that are affine functions of states and inputs and are active on a partition of the domain region, i.e. subregions [45]. PWA approximation of nonlinear functions has long been used in diverse applications, contributing to enhanced modeling power [46], improved computational efficiency [47], identification of explicit control laws [48] or serving as a descriptor for neural networks in machine learning [49]. Moreover, complexity of PWA approximations [50] and their verification processes [51] has been examined in the literature.

Approximating a nonlinear function by a PWA form is rather straightforward if the subregions are known a priori, as the problem boils down to determining local affine approximations on each subregion. The knowledge of the subregions may arise from the knowledge of different regimes (refer to the tire model in [52]) or from the knowledge of different equilibria (refer to the chaos model in [53]). However, in numerous applications, the difficulty arises when we lack prior information about the location of the subregions and the quantity of local modes to reach a particular approximation accuracy.

The conceptualization of PWA approximation as an optimization problem becomes notably intricate when dealing with multi-dimensional nonlinear functions. Even for known, yet multi-dimensional models such as in resistor networks [54], analytical solution of the PWA approximation problem may be elusive and the optimization problem should better be formulated for a set of points sampled from the domain [55]. This idea resembles PWA approximation approaches learned through experimental data [56]. A question arise about how to sample the points. A trivial solution is taking as many subregions as data points [45]; however, this approach easily leads to overfitting. Therefore, for the optimization problem to become well-posed, the number of local modes is often fixed while minimizing approximation error [45] or its expectation [57].

Various methods have been used to formulate the PWA approximation problem [45, 58, 59]. A common formulation is a bi-level optimization problem [60] that can be recast into a mixed-integer program [61], or solved in a recursive manner [62–65]. While recursive solutions are fast and can be used for online PWA approximation [39], they are often limited in handling multi-dimensional systems and are most effective when the form of the subregions is partially known and just needs to be refined [38]. For instance, in [66] more vertices are iteratively added to the subregions for improved accuracy, but the solver needs to be properly initialized.

Bi-level optimization arises because PWA approximation essentially has two key aspects: establishing a partitioning strategy to divide the domain into subregions, and finding the local affine approximations. A popular partitioning strategy is clustering of the mesh points [40, 64, 67, 68], which can be sensitive to the mesh quality [65]. Despite the efforts to reduce the sensitivity to the cluster boundary and outliers [41], the performance of clustering-based partitioning degrades for multi-dimensional nonlinear functions. Some formulations use a specific shape for the partition, e.g. using hyper-rectangular subregions for digital systems [69], using the function gradient [37], which is only applicable for uni-dimensional domains, or simplicial representation [70], which is applicable for low-dimensional domains [58]. Conversely, the hinging hyperplanes formalism, where the function is defined as a sum of hinging functions, e.g. min and max, of parameterized

hyperplanes, have proven to be an efficient and tractable formulation for partitioning of multi-dimensional domains [71].

If the approximation problem is solved offline, the main goal is to converge to the most accurate PWA approximation with minimal complexity. This essentially requires a partitioning strategy that is flexible enough to divide the domain in various ways and consequently, allow for finding a PWA approximation of a desired accuracy via defining the lowest possible number of subregions. Additionally, an appropriate methodology should be able to yield a continuous PWA approximation, which was handled by using min and max operators in [72]. However, using max and min operators does not allow for a discontinuous form. In this sense, using the continuity constraint from [54] is more suitable for more flexibility.

In this chapter, we present a novel approach for PWA approximation of multi-dimensional nonlinear systems using cut-based partitioning of the domain via a hinging hyperplane formalism. The complexity level of the PWA form can be iteratively increased if the intended level of complexity is either unknown or not restricted. In our earlier work [73], we have tackled the PWA approximation of multi-dimensional nonlinear systems with no prior knowledge. In this chapter, we generalize the cutting definitions and the formulation of the optimization problem, enhancing the flexibility of the partitioning strategy. These extensions enable evaluating cuts that were not possible with our prior approach, and allow for a tractable implementation of continuity constraints in the approximation problem. In summary, our novel PWA approximation approach improves upon the state-of-the-art methods by (1) flexible definition of subregions using a generalized hinging hyperplane formalism to allow for finding PWA approximations with fewer number of subregions for a given approximation error tolerance, and (2) tractable bi-level formulation of the optimization problem to facilitate modifications based on applications e.g. imposing the continuity constraint on the PWA approximation. The efficacy of these extensions will be shown via illustrative examples. Further, we compare our method to the approach in [65] where a recursive solution of the PWA approximation problem helps reducing the complexity and we show that our optimization formulation and cut-based partitioning allows for convergence to the same accuracy level with fewer number of subregions.

The structure of the chapter is as follows: Section 3.2 covers the main definitions and the formulation of the approximation problem. The novel cut-based partitioning strategy is explained in Section 3.3 using examples for clarity, and Section 3.4 presents the resulting optimization problem and the solution procedure. Case studies and comparisons are described in Section 3.5, while Section 3.6 concludes this chapter.

## 3.2 PROBLEM FORMULATION

Let us consider a nonlinear system with dynamics

$$\dot{s} = F(s, u),$$

where  $s \in \mathbb{R}^n$  and  $u \in \mathbb{R}^m$  respectively represent the state and input vectors and  $F : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^n$  is the nonlinear function to be approximated. Without loss of generality, the augmented state vector  $x = [s^T \ u^T]^T$  is used to define  $F(x) := F(s, u)$  since the approximated function will be selected to be affine in both the state and the input. The augmented domain is

assumed to be bounded and is defined as  $\text{dom}(F) = \mathcal{D} \subset \mathbb{R}^{n+m}$ . For brevity of the expressions, hereafter we will use  $d = n + m$  as the dimension of the domain.

**Definition 3.1** (Domain  $\mathcal{D}$ ). *The domain  $\mathcal{D} \subset \mathbb{R}^d$  is defined by the scalar boundary function  $g : \mathbb{R}^d \rightarrow \mathbb{R}$  as*

$$\mathcal{D} := \{x \in \mathbb{R}^d \mid 0 \leq g(x) \leq 1\}.$$

**Remark 3.1.** *We use the normalized form  $0 \leq g \leq 1$  instead of  $g \geq 0$  to avoid numerical issues in solving the approximation optimization problem.*

3

The proposed approach approximates the nonlinear function  $F$  by a PWA function  $f$  defined as

$$x \in C_p \implies f(x) = f_p(x), \quad f_p(x) = J_p x + K_p, \quad (3.1)$$

with  $p \in \{1, 2, \dots, P\}$ , where the matrices  $J_p \in \mathbb{R}^{n \times d}$  and the vectors  $K_p \in \mathbb{R}^n$  describe in total  $P$  local affine modes, each defined on a polytopic subregion  $C_p \subset \mathbb{R}^d$  such that the polytopes form a partition of  $\mathcal{D}$ , i.e. the subregions are nonempty,

$$C_p \neq \emptyset, \quad \forall p \in \{1, \dots, P\} \quad (3.2a)$$

they are non-overlapping,

$$\text{int}(C_p) \cap \text{int}(C_q) = \emptyset, \quad \forall p, q \in \{1, \dots, P\}, \quad p \neq q \quad (3.2b)$$

and their union covers the entire domain,

$$\bigcup_{p=1}^P C_p = \mathcal{D}, \quad (3.2c)$$

with  $\text{int}(C_p)$  denoting the interior of region  $C_p$ .

**Definition 3.2** (Region set  $\mathcal{R}$ ). *The region set  $\mathcal{R}$  is the ordered set collecting the partition (i.e. the subregions) as*

$$\mathcal{R} := \{C_1, C_2, \dots, C_P\}.$$

For a fixed  $P$ , the region set  $\mathcal{R}$  and the corresponding local affine approximations  $f_p$  are obtained simultaneously via solving the optimization problem

$$\min_{\mathcal{J}, \mathcal{K}, \mathcal{R}} \int_{\mathcal{D}} \frac{\|F(x) - f(x)\|_2^2}{\|F(x)\|_2^2 + \epsilon} dx, \quad (3.3)$$

$$\text{subject to} \quad (3.1) - (3.2), \quad (3.4)$$

to minimize the squared approximation error where  $\mathcal{J}$  and  $\mathcal{K}$  represent the ordered sets containing  $J_p$  and  $K_p$ , respectively. The term  $\|F(x)\|_2^2$  in the denominator is introduced such that the cost values represent the relative error and the added scalar  $\epsilon > 0$  prevents division by very small values where  $\|F(x)\|_2 \approx 0$ .

### 3.3 PARAMETRIC REGION DEFINITION

We partition  $D$  into  $P$  subregions by cutting it using  $(d-1)$ -dimensional hyperplanes.

**Definition 3.3** (Cutting hyperplane  $H_i$ ). *The  $i$ -th cutting hyperplane  $H_i$  is an affine subspace of  $\mathbb{R}^d$  defined as*

$$H_i := \{x \mid h_i^T x - 1 = 0\},$$

for  $i \in \{1, \dots, n_c\}$  where  $n_c$  represents the number of cuts.

**Definition 3.4** (Cut arrangement  $\mathcal{H}$ ). *The cut arrangement  $\mathcal{H}$  is the arrangement of  $n_c$  cutting hyperplanes defined by the set*

$$\mathcal{H} = \{H_1, H_2, \dots, H_{n_c}\}.$$

**Remark 3.2.** *In principle, the number of subregions generated by cutting  $\mathbb{R}^d$  via the arrangement  $\mathcal{H}$  can be calculated using Zaslavsky's theorem [74] provided that all the possible 0- to  $(d-1)$ -dimensional intersections of the hyperplanes in  $\mathcal{H}$  are obtained. As a more computationally-efficient approach, here we fix the number of cutting hyperplanes and numerically obtain the region set  $\mathcal{R}$  within  $D$  by investigating the existence of the possible subregions created by  $\mathcal{H}$  without counting the 0- to  $(d-1)$ -dimensional intersections. As a result,  $P$  is not fixed a priori.*

To define each cutting hyperplane, we generate  $d$  points in  $\mathbb{R}^d$  and find the hyperplane passing through them as shown in Fig. 3.1. These points are defined on the surface of an enclosing hypersphere  $S$  in  $D$  to ensure they are linearly-independent.

**Definition 3.5** (Enclosing hypersphere  $S$ ). *The enclosing hypersphere  $S$  is the smallest  $d$ -dimensional hypersphere enclosing  $D$  defined by*

$$S := \{x \mid \|x\|_2^2 - \rho^2 = 0\},$$

where the constant  $\rho$  is the radius of the enclosing hypersphere.

**Remark 3.3.** *Without loss of generality, one can always define a coordinate shift for the domain so that the center of the hypersphere is located at the origin. This allows to simplify the expressions and mathematical manipulations.*

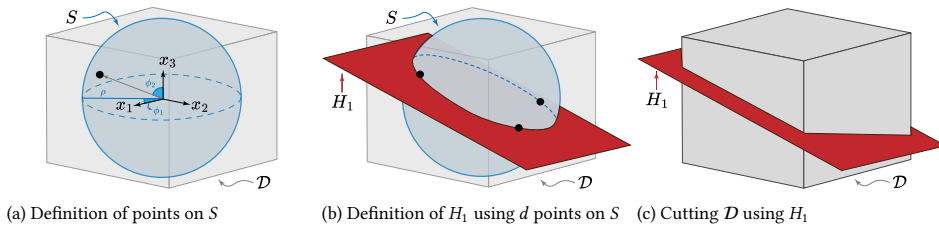


Figure 3.1: A schematic view cutting the domain using a hypersphere.

To locate the points on  $S$ , we use spherical coordinates ( $r = \rho, \phi_1, \phi_2, \dots, \phi_{d-1}$ ) to parameterize the locations using spatial angles

$$\phi_1, \phi_2, \dots, \phi_{d-2} \in [0, \pi], \quad \phi_{d-1} \in [0, 2\pi].$$

Figure 3.1a shows this concept for  $d = 3$ . The Cartesian coordinates of the  $k$ -th point  $x_k = [x_{1,k}, \dots, x_{d,k}]^T$  can then be obtained by [75]

$$x_{j,k} = \rho \cos \phi_{j,k} \prod_{v=1}^{j-1} \sin \phi_{v,k}, \quad \forall j \in \{1, \dots, d-1\}, \quad (3.5a)$$

$$x_{d,k} = \rho \prod_{v=1}^{d-1} \sin \phi_{v,k}, \quad (3.5b)$$

where  $x_{j,k}$  is the  $j^{\text{th}}$  component of the  $k^{\text{th}}$  point. Then, to find the hyperplane  $H_i$  passing by  $d$  points, we need to solve

$$X_i h_i = \mathbf{1}_{d \times 1} \implies h_i = X_i^{-1} \mathbf{1}_{d \times 1}, \quad (3.6)$$

where

$$X_i = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,d} \\ x_{2,1} & x_{2,2} & \dots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{d,1} & x_{d,2} & \dots & x_{d,d} \end{bmatrix}.$$

Figure 3.1b illustrates the generation of one hyperplane  $H_1$  for  $d = 3$  and how the domain is cut into two partitions in Fig. 3.1c. For more cuts, we then need to proceed analogously for all  $n_c$  cuts, obtaining  $\mathcal{H}$  as

$$\mathcal{H} = \{\mathbf{1}_{1 \times d} X_i^{-T} x = 1\}, \quad \forall i \in \{1, \dots, n_c\}. \quad (3.7)$$

Since each hyperplane divides the domain into two half-spaces, we define the map  $\sigma$  from the domain  $\mathcal{D}$  to the  $n_c \times 1$  Boolean vector  $\sigma$  as

$$\sigma_i(x) = \begin{cases} 0 & \text{if } h_i^T x < 1 \\ 1 & \text{if } h_i^T x \geq 1 \end{cases}, \quad \forall i \in \{1, \dots, n_c\}, \quad (3.8)$$

to indicate which side of the hyperplane  $H_i$  the point  $x$  lies on. Since the subregions are also located on one side of each hyperplane, there exist at most  $2^{n_c}$  possible partitions that can be stored in an  $n_c \times 2^{n_c}$  matrix. However, to avoid unnecessary usage of memory, we suggest generating  $\sigma$  vectors by investigating the binary vectors corresponding to integer numbers from 0 to  $2^{n_c} - 1$  without storing all of them in a very large matrix. We use the prune-and-search paradigm [76] by solving  $2^{n_c}$  linear programs to check the feasibility of each combination

$$\min_x \quad 1, \quad (3.9)$$

$$\text{subject to} \quad x \in \mathcal{D}, \quad (3.10)$$

$$h_i^T x < 1 \quad \text{if } \sigma_i = 0, i \in \{1, \dots, n_c\}, \quad (3.11)$$

$$h_i^T x \geq 1 \quad \text{if } \sigma_i = 1, i \in \{1, \dots, n_c\}, \quad (3.12)$$



where  $\sigma \in \mathbb{R}^{n_c}$  is the binary representation of the integer  $j \in \{0, \dots, 2^{n_c} - 1\}$  in each linear program. The  $\sigma$  vectors corresponding to feasible problems are then stored in the feasibility matrix  $\Sigma \in \mathbb{R}^{n_c \times P}$ . This procedure is implemented by Algorithm 2.

---

**Algorithm 2** Find feasibility matrix of a cut arrangement as  $\Sigma = \text{chambers}(\mathcal{H}, \mathcal{D})$

---

**Require:**  $\mathcal{H}, \mathcal{D}$

$n_c \leftarrow |\mathcal{H}|$   $\triangleright |\cdot|$  denotes cardinality

$\sigma \leftarrow 0_{n_c \times 1}$

$\Sigma \leftarrow 0_{n_c \times 0}$

$\triangleright 0_{n_c \times 0}$  is an empty matrix

**for**  $l \in \{0, 1, \dots, 2^{n_c} - 1\}$  **do**

$\sigma \xleftarrow{\text{binary vector}} l$

**if** (3.9)-(3.12) feasible for  $(\sigma, \mathcal{H})$  **then**

$\Sigma \leftarrow \begin{bmatrix} \Sigma & \sigma \end{bmatrix}$

**end if**

**end for**

**return**  $\Sigma$

---

**Example 3.1.** Consider the 3-dimensional hypercube domain  $\mathcal{D}$  shown in Fig. 3.2 and two cut arrangements  $\mathcal{H}_a$  and  $\mathcal{H}_b$  respectively shown in Figures 3.2a and 3.2b. We have

$$n_c = 2 \implies l \in \{0, 1, 2, 3\} \xrightarrow{\text{binary}} \{00, 01, 10, 11\}.$$

Problem (3.9)-(3.12) is not feasible for  $\sigma = 01$  for  $\mathcal{H}_a$  as there is no region within  $\mathcal{D}$  lying above  $H_1$  and below  $H_2$ . However, (3.9)-(3.12) is feasible for  $\sigma \in \{00, 10, 11\}$ . Therefore, the feasibility matrices for  $\mathcal{H}_a$  and  $\mathcal{H}_b$  are

$$\Sigma_a = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad \Sigma_b = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

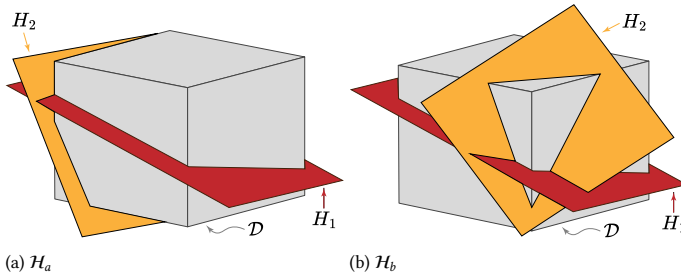


Figure 3.2: A schematic view of two cut arrangements in Example 1.

In the combination geometry literature, a similar concept is used but in a set of tuples called the oriented metroid. For more details see [77].

Next, we aim at finding the neighboring subregions to identify the hyperplane they share as their boundary. We define the adjacency matrix  $A$  to store this information.

**Definition 3.6** (Adjacency Matrix  $A$ ). *The adjacency matrix  $A \in \mathbb{R}^{P \times P}$  represents the neighboring subregions within the region set  $\mathcal{R}$  as follows:*

$$A_{p,q} = \begin{cases} i & \text{if } C_p \cap C_q = H_i \\ 0 & \text{otherwise} \end{cases}.$$

The adjacency matrix is symmetric by definition.

Note that the adjacent subregions share their  $\sigma$  vector, except for only one element, which is the element corresponding to their boundary hyperplane. Thus, the adjacency matrix is constructed by investigating the columns in  $\Sigma$ . Since each column in  $\Sigma$  is a binary vector, two columns differ only in one element if and only if their subtraction contains only one  $\pm 1$  element, i.e.

$$\|\Sigma_{:,p} - \Sigma_{:,q}\|_1 = 1.$$

Each column of  $\Sigma$  represents a subregion and how it relates to the hyperplanes in  $\mathcal{H}$ . However, defining each region is done only by evaluating its boundaries and not all the cutting hyperplanes to avoid redundancy. Since each column (or row) of  $A$  corresponds to one of the elements in  $\mathcal{R}$ , the subregion  $p$  can now be formulated as follows:

$$C_p = \left\{ x \in \mathcal{D}, \forall i > 0, A_{p,i} = i \mid (-1)^{\Sigma_{i,p}} h_i^T x \leq (-1)^{\Sigma_{i,p}} \right\}. \quad (3.13)$$

The subregions are then stored in the region set  $\mathcal{R}$ . Algorithm 3 describes the procedure of obtaining  $\mathcal{R}$  from  $\mathcal{H}$  and  $\Sigma$  using the adjacency matrix  $A$ .

**Example 3.2.** Consider the 3-dimensional domain shown in Fig. 3.3 as

$$\mathcal{D} := \{ x \in \mathbb{R}^3 \mid x_j \in [-2, 2], j \in \{1, 2, 3\} \},$$

with the cut arrangement  $\mathcal{H} = \{H_1, H_2, H_3\}$  shown in Fig. 3.3a where

$$H_1 := \{ x \in \mathbb{R}^3 \mid -x_1 + 2x_2 + 5x_3 = 1 \},$$

$$H_2 := \{ x \in \mathbb{R}^3 \mid 0.1x_1 - 0.5x_2 - 0.2x_3 = 1 \},$$

$$H_3 := \{ x \in \mathbb{R}^3 \mid -x_1 + x_2 = 1 \}.$$

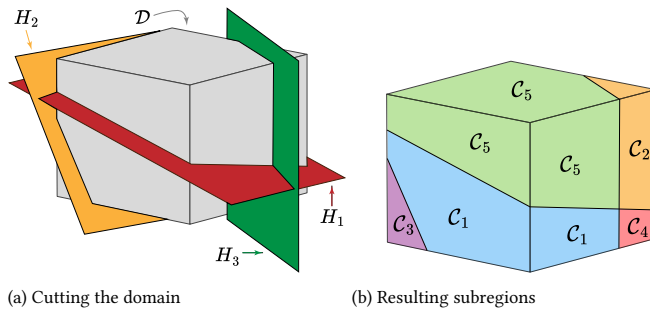


Figure 3.3: Illustration of the cut arrangement and the resulting subregions in Example 2.

---

**Algorithm 3** Find subregions of a cut arrangement as  $(\mathcal{R}, A, P) = \text{regions}(\mathcal{H}, \Sigma)$ 


---

**Require:**  $\mathcal{H}, \Sigma$ 

```

 $n_c \leftarrow |\mathcal{H}|$   $\triangleright |\cdot| := \text{cardinality}$ 
 $P \leftarrow$  number of columns in  $\Sigma$ 
 $A \leftarrow 0_{P \times P}$ 
 $\mathcal{R} \leftarrow \emptyset$ 
for  $p \in \{1, \dots, P\}$  do
  for  $q \in \{p+1, \dots, P\}$  do
     $\delta \leftarrow \Sigma_{:,p} - \Sigma_{:,q}$ 
    if  $\|\delta\|_1 = 1$  then
       $A_{p,q} \leftarrow$  index of nonzero component in  $\delta$ 
    end if
  end for
end for
 $A \leftarrow A + A^T$   $\triangleright$  upper-triangular to symmetric
for  $p \in \{1, \dots, P\}$  do
   $C_p \xleftarrow{\text{solve (3.13)}} \mathcal{H}, A$ 
   $\mathcal{R} \leftarrow \mathcal{R} \cup \{C_p\}$ 
end for
return  $\mathcal{R}, A, P$ 

```

---

The feasibility and adjacency matrices for  $\mathcal{H}$  are

$$\Sigma = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 3 & 2 & 1 & 0 \\ 3 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 3 & 0 \end{bmatrix},$$

which gives the subregions as depicted in Fig. 3.3b expressed by

$$\begin{aligned} C_1 &:= \{x \in \mathcal{D} \mid \begin{bmatrix} -1 & 2 & 5 \\ 0.1 & -0.5 & -0.2 \\ -1 & 1 & 0 \end{bmatrix} x \leq 1\}, & C_2 &:= \{x \in \mathcal{D} \mid \begin{bmatrix} -1 & 2 & 5 \\ 1 & -1 & 0 \end{bmatrix} x \leq 1\}, \\ C_3 &:= \{x \in \mathcal{D} \mid \begin{bmatrix} -0.1 & 0.5 & 0.2 \end{bmatrix} x \leq 1\}, & C_4 &:= \{x \in \mathcal{D} \mid \begin{bmatrix} 1 & -2 & -5 \\ -1 & 1 & 0 \end{bmatrix} x \leq 1\}, \\ C_5 &:= \{x \in \mathcal{D} \mid \begin{bmatrix} 1 & -2 & -5 \\ 1 & -1 & 0 \end{bmatrix} x \leq 1\}, \end{aligned}$$

### 3.4 CUT-BASED PWA APPROXIMATION

The PWA approximation of a nonlinear function is obtained by solving (3.3) by optimizing the partitioning strategy via  $\mathcal{R}$  and the local affine approximations  $\mathcal{J}$  and  $\mathcal{K}$ , while satisfying (3.2). Using the cut-based parametric region definition through Algorithms 2 and 3,

constraints (3.2) are automatically satisfied since we cut the domain with hyperplanes which gives  $\mathcal{R}$  as a set of non-overlapping partitions that cover the whole  $\mathcal{D}$ .

If the aim is to give a continuous PWA approximation of the system, the following constraint should be imposed on the dynamic modes corresponding to neighboring subregions, which is derived based on [78]:

$$\exists c \in \mathbb{R}^d \quad \text{s.t.} \quad J_p - J_q = c h_i^T, \quad \forall p, q \in \{1, \dots, P\}, H_i = C_p \cap C_q. \quad (3.14)$$

**Corollary 3.1.** *The difference of  $J_p$  and  $J_q$  in (3.14) is of rank one [78]. As a result, the continuity can be imposed on the PWA approximation by considering rank of the Jacobian matrices in neighboring modes.*

Even with a fixed number of cutting hyperplanes, the number of subregions (and consequently the dimensions of the decision space) differs for different  $\mathcal{H}$  arrangements with the same  $n_c$ . Due to its varying-dimensional nature, we formulate the PWA approximation problem as a bi-level optimization problem. At the lower level, we find the minimum approximation error for the region set  $\mathcal{R}$  as

$$\Gamma^*(\mathcal{R}) = \min_{J, K} \sum_{p=1}^P \int_{C_p} \frac{\|F(x) - J_p x - K_p\|_2^2}{\|F(x)\|_2^2 + 1} dx, \quad (3.15)$$

$$\begin{aligned} \text{s.t.} \quad & \text{rank}(J_p - J_q) = 1, \\ & \forall p, q \in \{1, \dots, P\}, A_{p,q} \neq 0, \end{aligned} \quad (3.16)$$

where (3.16) is the continuity constraint and can be disregarded in case a continuous PWA form is not required. Problem (3.15)-(3.16) is a nonlinear least-squares optimization problem that can be solved using e.g. gradient-based methods with multiple starting points.

At the higher level, we solve the following optimization problem:

$$\min_{\phi} \Gamma^*(\mathcal{R}) + \lambda P \quad (3.17)$$

$$\text{s.t.} \quad \mathcal{H} = \{1_{1 \times d} X_i^{-T} x = 1\}, \quad \forall i \in \{1, \dots, n_c\}, \quad (3.18)$$

$$\Sigma = \text{chambers}(\mathcal{H}, \mathcal{D}), \quad (3.19)$$

$$(\mathcal{R}, A, P) = \text{regions}(\mathcal{H}, \Sigma), \quad (3.20)$$

$$\phi_{i,j} \in [0, \pi], \quad \phi_{i,d-1} \in [0, 2\pi], \quad \forall i \in \{1, \dots, n_c\}, \forall j \in \{1, \dots, d-2\}, \quad (3.21)$$

with  $\phi$  collecting  $\phi_{i,j}$ ,  $\Gamma^*$  being the approximation error obtained in the lower level and the `chambers` and `regions` functions correspond to the Algorithms 2 and 3, respectively. Here we penalize the number of subregions  $P$  as well by a regularizing weight  $\lambda > 0$ . Similar to the lower-level problem, (3.17)-(3.21) is a nonlinear optimization problem. However, being not smooth, we propose solving it using global optimization methods such as genetic algorithm.

Equations (3.15)-(3.21) are solved for a fixed  $n_c$ . In our approach, we start by  $n_c = 1$  and in case the best solution does not reach a user-defined error tolerance, we increase  $n_c$  and solve (3.15)-(3.21) again. Algorithm 4 describes this iterative cut-based PWA optimization problem.

---

The term “chambers” is often used in the combinatorial geometry literature for the “subregion” concept. Here, to distinguish between the functions in Algorithms 2 and 3, we use this term as a label for clarity.

**Algorithm 4** Iterative cut-based PWA approximation

---

**Require:**  $F, D$   
 generate  $S$   
 $\text{cond} \leftarrow \text{true}$   
 $n_c \leftarrow 0$   
 $\text{iter} \leftarrow 0$   
**while**  $\text{cond}$  **do**  
    $n_c \leftarrow n_c + 1$   
    $\text{iter} \leftarrow \text{iter} + 1$   
    $(\Gamma^*, \mathcal{J}^*, \mathcal{K}^*, \mathcal{R}^*) \leftarrow \text{solve (3.15)-(3.21)}$   
   **if**  $\Gamma^* \leq \text{tol}_{\text{err}}$  **then**  
      $\text{cond} \leftarrow \text{false}$   
     **return**  $\mathcal{J}^*, \mathcal{K}^*, \mathcal{R}^*$   
   **else if**  $\text{iter} \geq \text{iter}_{\text{max}}$  **then**  
      $\text{cond} \leftarrow \text{false}$   
     **return**  $\text{print('Exceeded iter}_{\text{max}}')}$   
   **end if**  
**end while**

---

### 3.5 RESULTS AND DISCUSSION

In this section, we analyze the performance of our proposed cut-based PWA approximation method in three steps: first, we use a case study as an illustrative example of a nonlinear function for a user-defined approximation error tolerance. Then, we compare our method with [73] and [65] to show the flexibility of our approach, leading to lower approximation errors for a lower complexity of PWA the form.

#### 3.5.1 CASE STUDY

To illustrate our cut-based PWA approximation approach, consider the system

$$\dot{x} = \sin(x + u^2),$$

with a 2-dimensional domain

$$\mathcal{D} := \{ (x, u) \in \mathbb{R}^2 \mid x \in [-2, 2], u \in [-2, 2] \},$$

to be able to display the function and the subregions. We approximate the nonlinear function Using Algorithm 4 by selecting the stopping criterion

$$\text{tol}_{\text{err}} = 5\%$$

and imposing the continuity constraint in (3.16). The algorithm reaches the stopping criterion with  $n_c = 8$  cuts by partitioning the domain into  $P = 16$  subregions. Figure 3.4 shows the resulting PWA approximation, region set, and the cutting hyperplanes.

#### 3.5.2 COMPARISON WITH APPROACHES IN THE LITERATURE

The comparison analysis is done separately for each approach from the literature, using case studies from its respective publication.

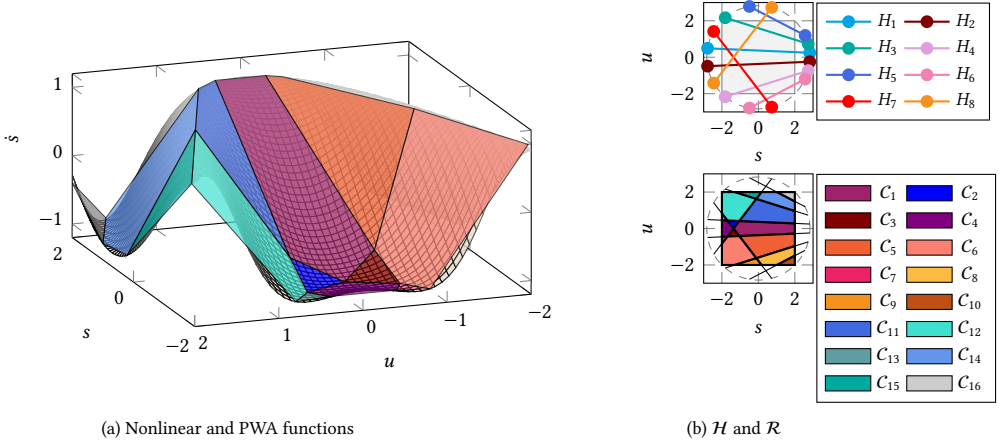


Figure 3.4: Cut-Based PWA approximation of the nonlinear function  $\hat{s} = \sin(s + u^2)$  using 8 cuts. Each local mode is shown in the same color as its corresponding subregion.

### MESH-BASED RECURSIVE ABSTRACTIONS

Here, we use the same benchmark incorporated in [65]: the Dubins vehicle dynamics  $F : [-2, 2] \times [0, 2\pi] \rightarrow \mathbb{R}$  given by

$$F(x) = x_1 \cos(x_2).$$

The abstraction error is defined as the distance between the upper and lower PWA approximations of  $F$ . Therefore, for a fair comparison, we assume the approximation error to be half of the abstraction error.

Table 3.1 compares the number of required subregions to reach three different user-defined maximum approximation errors. It is evident that the new cut-based approach can reach the same accuracy level with significantly fewer subregions. However, it should be noted that the two methods introduced in [65] converge to their minima faster than our approach. Therefore, the benefit of the new cut-based method is best realized for offline computations aimed at reaching a more accurate, yet simpler, PWA approximation forms.

Table 3.1: Number of required subregions using different PWA approximation methods

Approach	Maximum error		
	10%	5%	2.5%
Cut-based approach	12	24	40
Method I [65]	58	108	210
Method II [65]	61	121	257

### PARAMETRIC PLANE-BASED CUTTING STRATEGY

In the previous chapter, a cut-based PWA approximation is done by pairing the input-state vectors to form 2D hyperplanes and the cutting hyperplanes are defined perpendicular to

the selected pairs. As one of the illustrative examples in [73], the longitudinal velocity of a single-track vehicle model with linear tires is denoted by  $F : \mathbb{R}^6 \rightarrow \mathbb{R}$  is given by

$$F(x, u) = \frac{u_1 \cos(u_3) + u_2}{\eta_1} + \eta_2 \left[ \tan^{-1} \left( \frac{x_2 + \eta_3 x_3}{x_1} \right) - u_3 \right],$$

with  $\eta_1 = 1970$ ,  $\eta_2 = 64.36$ , and  $\eta_3 = 1.48$ , incorporating 3 states and 3 inputs. To approximate  $F$ , 3 non-intersecting cuts are required on  $(x_2, u_3)$  to partition the domain into 4 subregions and achieving the – significantly high – approximation error of 30%. Moreover, the resulting PWA approximation is not continuous.

Table 3.2 compares different cut-based PWA approaches in terms of the continuity of their respective number of regions, their maximum error, and the continuity of their PWA form. Using our proposed cutting strategy, we are no longer limited to plane-based cutting of the domain, which significantly improves our flexibility in partitioning the domain, which results in obtaining a PWA approximation with 3 local modes for a maximum error of 3%. Moreover, we can now enforce a continuous PWA approximation of the dynamics. As a result, we are able to reach 3% approximation error using only two cuts: one on  $v_y - r$  and one on  $v_x - v_y$  axis.

Table 3.2: Comparing cut-based PWA approximation approaches

Approach	Continuous form	Number of regions	Maximum error
Current method	Yes	4	3%
Current method	No	3	3%
Previous method [73]	No	4	30%

### 3.6 CONCLUSIONS

In this chapter, we have proposed a novel approach for PWA approximation of nonlinearities using a hinging-hyperplane formulation to partition the domain into subregions. Our proposed method does not require prior knowledge of the dynamics, is applicable to nonlinear systems defined on multi-dimensional domains, and allows for a straightforward formulation of the continuity constraint for the PWA approximation. To avoid unnecessary complexity in the final form, the number of cutting hyperplanes is iteratively increased in case the solutions of the approximation problem are unable to satisfy a user-defined error tolerance. The flexibility of our proposed approach allows for various polytopic subregion definitions and adaptability for different approximation requirements. By comparing the performance of our approach to other state-of-the-art methods from the literature, we have showcased its potential for practical applications in complex, large-scale systems, paving the way for future advancements in nonlinear function approximation and control.





## 4

# H4MPC: A HYBRIDIZATION TOOLBOX FOR MPC

4

*Adults are bewitched by their language because they try to apply discrete words to continuous activities. Before the breakaway of natural language, words were activities. Yet we speak, very discretely. This is just what, and all, we can do.*

— Council of Human Hybrid-Attractors, *Incessance: Incesancia*

*The computational complexity of Nonlinear Model Predictive Control (NMPC) poses a significant challenge in achieving real-time levels of 4 and 5 of automated driving. This chapter presents the open-access Hybridization toolbox for Model Predictive Control (MPC) (H4MPC), targeting computational efficiency of NMPC thanks to several modules to hybridize NMPC optimization problems commonly encountered in automated driving applications. H4MPC is designed as a user-friendly solution with a graphical user interface within the MATLAB environment. The toolbox facilitates intuitive and straightforward customization of the hybridization process for any given function appearing in the equality or inequality constraints within the MPC framework. The initial release, Version 1.0, is freely available from <https://bit.ly/H4MPCV1>. To provide a clear illustration of the toolbox capabilities, we present two case studies: one to hybridize a vehicle model and another one to approximate tire saturation constraints.*

## 4.1 INTRODUCTION

Nonlinearity of the MPC optimization problem in automated driving is a significant obstacle towards real-time vehicle control [12]. Approximating the nonlinearities is often done in many applications [80] to come up with improved computational efficiency in solving the nonlinear control optimization problem. In this line, hybridization techniques [81]

This chapter has been published in the proceedings of IEEE Conference on Advance Motion Control [79].

approximate a nonlinear function using hybrid systems formalism, with both continuous and discrete-time dynamics involved in the approximation [18]. For more information on hybrid systems, the reader is referred to [23, 82].

Hybridization has been extensively employed in automated driving research, e.g., in vehicle control [21] by approximating the nonlinear model using a mixed-logical dynamics [26], or by approximating nonlinear tire forces using PieceWise Affine (PWA) dynamics [36, 83, 84]. Efficiency of MPC after hybridization has been recently studied in [85, 86].

This chapter presents H4MPC [87], a hybridization toolbox in MATLAB that provides a user-friendly interface to formulate and solve optimization problems to approximate the nonlinearities in NMPC using hybrid systems formalism, in particular PWA modeling framework. The toolbox exploits the formulation from Max-Min-Plus-Scaling (MMPS) systems [24] to allow for an intuitive adjustment of the complexity level in the approximated form. Further, H4MPC facilitates approximation of the nonlinear constraints via covering the resulting non-convex feasible region by a union of convex subregions, namely ellipsoids or polytopes, where the latter are obtained using MMPS formalism as well.

This chapter describes the H4MPC modules in detail and demonstrates its capabilities using two case studies: approximating a single-track vehicle model [88], and hybridizing the non-convex feasible region due to tire saturation limits, known as the Kamm circle constraint [89] for a Pacejka tire model [90]. The chapter is structured as follows: Section 4.2 covers the preliminary definitions, Section 4.3 presents the architecture of H4MPC and Section 4.4 illustrates the case studies and analysis of the results. Finally, Section 4.5 summarizes the results of this chapter.

## 4

## 4.2 PRELIMINARIES

### 4.2.1 NONLINEAR PROBLEM DESCRIPTION

Consider a given discrete-time nonlinear system  $s(k+1) = F(s(k), u(k))$  where  $s \in \mathbb{R}^n$  and  $u \in \mathbb{R}^m$  respectively represent the state and input vectors, and the domain of  $F$  is denoted by  $\mathcal{D} \subseteq \mathbb{R}^{n+m}$ . With the state and input vectors defined over the whole prediction horizon  $N_p$  as

$$\tilde{s}(k+1) = [\hat{s}^T(k+1|k) \quad \hat{s}^T(k+2|k) \quad \dots \quad \hat{s}^T(k+N_p|k)]^T,$$

$$\tilde{u}(k) = [u^T(k) \quad u^T(k+1) \quad \dots \quad u^T(k+N_p-1)]^T,$$

and  $\hat{s}^T(k+i|k)$  with  $i \in \{1, \dots, N_p\}$  representing the prediction of the states in step  $k+i$  given the measured states at step  $k$ , the nonlinear MPC problem at step  $k$  is formulated in the general form:

$$\min_{\tilde{s}(k), \tilde{u}(k)} \|\Theta_s \tilde{s}(k)\|_\rho + \|\Theta_u \tilde{u}(k)\|_\rho, \quad (4.1a)$$

$$\text{s.t. } \hat{s}(k+i|k) = F(\hat{s}(k+i-1|k), u(k+i-1)), \quad \forall i \in \{1, \dots, N_p\}, \quad (4.1b)$$

$$G(\hat{s}(k+i-1|k), u(k+i-1)) \leq 1, \quad \forall i \in \{1, \dots, N_p\}, \quad (4.1c)$$

where (4.1b) represents the equality constraints due to the prediction model, and (4.1c) expresses the non-convex feasible region via the normalized nonlinear constraint function  $G$

resulting from physics-based constraints such as tire saturation or vehicle stability. Without loss of generality, we assume  $G$  to be a scalar function. The objective function (4.1a) is a sum of the  $\rho$ -norm of the state and input vectors with  $\rho \in \{1, 2, \infty\}$ , induced by weight matrices  $\Theta_s$  and  $\Theta_u$ .

#### 4.2.2 APPROXIMATION OF THE NONLINEAR PROBLEM

Hybridization of the NMPC problem is done in two steps: (1) approximating the prediction model, i.e.  $F$ , and (2) approximating the nonlinear constraints, i.e.  $G$ . For a more compact representation, we use the augmented state vector  $x = [s^T \ u^T]^T$  to define  $F(x) := F(s, u)$  and  $G(x) := G(s, u)$ .

##### MODEL APPROXIMATION

We approximate each component  $F_w$  of  $F = [F_1 \ \dots \ F_n]^T$  by an MMPS function  $f_w$  in the Kripfganz form [91] as

$$f_w(x) = \max \left\{ \phi_{w,1}^+(x), \phi_{w,2}^+(x), \dots, \phi_{w,P_w^+}^+(x) \right\} - \max \left\{ \phi_{w,1}^-(x), \phi_{w,2}^-(x), \dots, \phi_{w,P_w^-}^-(x) \right\}, \quad (4.2)$$

$\forall w \in \{1, \dots, n\},$

where the vectors  $\phi_s^\eta : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{P^\eta}$  with  $\eta \in \{+, -\}$  are affine functions of  $x$ , also referred to as dynamic modes. Figure 4.1 shows an illustrative example for a 1-dimensional case with  $(P^+, P^-) = (3, 4)$ .

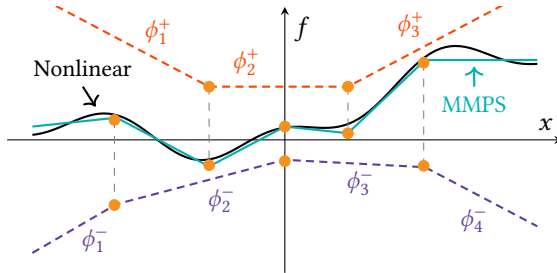


Figure 4.1: MMPS approximation of a nonlinear function using the difference of two max functions

##### CONSTRAINT APPROXIMATION

With the nonlinear, non-convex constraints given as  $G(x) \leq 1$ , we approximate the feasible region  $\mathcal{C} := \{x \in \mathcal{D} \mid G(x) \leq 1\}$  by a union of convex subregions  $\mathcal{R}$ . The shape of the subregions in  $\mathcal{R}$  can either be polytopic, which we obtain by an MMPS approximation of the boundary, or ellipsoidal.

Similar to the prediction model, MMPS approximation of the constraints is expressed by

$$g_{\text{MMPS}}(x) = \max \left\{ \gamma_1^+(x), \gamma_2^+(x), \dots, \gamma_{R^+}^+(x) \right\} - \max \left\{ \gamma_1^-(x), \gamma_2^-(x), \dots, \gamma_{R^-}^-(x) \right\}, \quad (4.3)$$

with the vectors  $\gamma^\eta : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{R^\eta}$  and  $\eta \in \{+, -\}$  being affine functions of  $x$ .

In the ellipsoidal approach,  $G$  is approximated by  $g_{\text{ELLP}}$  as

$$g_{\text{ELLP}}(x) = \min_{e \in \{1, \dots, n_e\}} \left\{ (x - x_{e,0})^T Q_e (x - x_{e,0}) - 1 \right\}, \quad (4.4)$$

with  $Q_e$  being positive definite matrices and  $x_{e,0}$  representing the center coordinates of the (possibly rotated)  $n_e$  ellipsoids. Figure 4.2 represents a schematic view of MMPS and ellipsoidal constraint approximations.

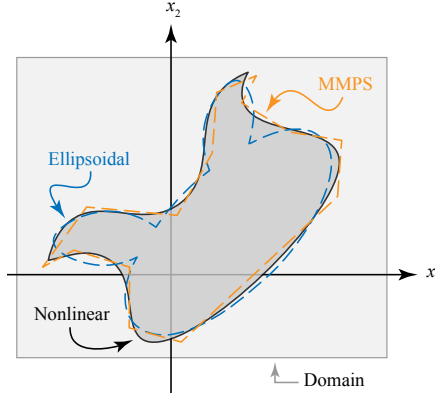


Figure 4.2: MMPS and ellipsoidal constraint approximation.

### 4.2.3 APPROXIMATION PROBLEM FORMULATION

All the nonlinear functions  $H \in \{F_1, \dots, F_n, G\}$  are approximated by their respective hybrid formulations  $h \in \{f_1, \dots, f_n, g\}$  for  $g \in \{g_{\text{MMPS}}, g_{\text{ELLP}}\}$  via solving the nonlinear optimization problem

$$\min_{\mathcal{A}} \int_{\mathcal{D}} \frac{\|H(x) - h(x)\|_2}{\|H(x)\|_2 + \epsilon_0} dx, \quad (4.5)$$

where  $\mathcal{A}$  represents the decision variables used to define  $h$  and the positive value  $\epsilon_0 > 0$  added to the denominator avoids division by very small values for  $\|H(x)\|_2 \approx 0$ . For the nonlinear constraint, (4.5) approximates the boundary of the feasible region, and therefore we call this approach “boundary-based”.

Another method to formulate the constraint approximation problem is the “region-base” approach where we formulate the optimization problem as

$$\min_{\mathcal{A}} \gamma_c \frac{\mathcal{V}\{C \setminus \mathcal{R}\}}{\mathcal{V}\{C\}} + (1 - \gamma_c) \frac{\mathcal{V}\{\mathcal{R} \setminus C\}}{\mathcal{V}\{\mathcal{D} \setminus C\}}, \quad (4.6)$$

where the operator  $\mathcal{V}$  gives the size or “volume” of the region, and  $\gamma_c \in [0, 1]$  is a tuning parameter to adjust the relative penalization weight for the misclassification errors regarding inclusion error  $C \setminus \mathcal{R}$ , i.e., failing to cover the feasible region, and the violation error  $\mathcal{R} \setminus C$  which corresponds to violating the constraints.

## 4.3 TOOLBOX ARCHITECTURE

The graphical user interface of the H4MPC toolbox is shown in Fig. 4.3 and consists of three steps, which correspond to the three modules in H4MPC as shown in Fig. 4.4: grid generation, model approximation, and constraint approximation. In each module, during grid generation, model approximation, and constraint approximation, the toolbox saves the results as separate .mat files in case the user is only interested in the output from one of the modules. The arrows in Fig. 4.4 illustrate the possible flow of using each module within H4MPC.

**Hybridization Toolbox for Model Predictive Control**

**Step 1: Grid Generation**

Main Information	Input/State Bounds	Model Grid	Constraint Grid
Number of states	<input type="text" value="2"/>	Number of inputs	<input type="text" value="1"/>
		Sampling time	
		<input type="text" value="0.01"/>	
Nonlinear model function			
<input type="text" value="@dyn_cv"/>			
<small>Note: This function should have two arguments in the following order: input signal vector, and the state vector. It should return the state derivatives as its output. Example: <math>\dot{x} = \text{model}(u, x)</math>.</small>			
Nonlinear constraint function			
<input type="text" value="@kamm"/>			
<small>Note: This function should have two arguments in the following order: input signal vector, and the state vector. The output should be a non-negative scalar. Example: <math>g = \text{constraint}(u, x)</math> where <math>0 \leq g \leq 1</math> means the combination of <math>x</math> and <math>u</math> is feasible.</small>			
<small>Note: Please fill in the above fields first.</small> <input type="button" value="Generate Training Grids"/> <span style="float: right;">Status</span>			

**Step 2: Model Approximation**

Enter P+ values for all the states	<input type="text" value="2,2"/>	Number of starting points (for multistart)	<input type="text" value="10"/>
Enter P- values for all the states	<input type="text" value="2,2"/>		
<small>Note: Please fill in the above fields first. P+ and P- values should be separated by comma. Example: 2,2.</small>			
<input type="button" value="Approximate Prediction Model"/> <span style="float: right;">Status</span>			

**Step 3: Constraint Approximation**

<div style="border: 1px solid black; padding: 5px;"> <p>Approach</p> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px; text-align: center;">Boundary-based</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">Region-based</div> </div> </div>	<div style="border: 1px solid black; padding: 5px;"> <p>Subregions</p> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px; text-align: center;">Polytopes</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">Ellipsoids</div> </div> </div>	<div style="display: flex; justify-content: space-between;"> <div>Number of ellipsoids</div> <div><input type="text" value="2"/></div> </div> <div style="display: flex; justify-content: space-between;"> <div>R+ and R- values</div> <div><input type="text" value="4,4"/></div> </div> <div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"><math>\gamma_c</math></div> <div style="flex-grow: 1; position: relative;"> <div style="position: absolute; left: 0; top: -5px;">0</div> <div style="position: absolute; right: 0; top: -5px;">0.5</div> <div style="position: absolute; right: 0; top: -5px;">1</div> </div> </div>	
<small>Note: Please fill in the above fields first. R+ and R- values should be separated by comma. Example: 2,2.</small>			
<input type="button" value="Approximate Constraints"/> <span style="float: right;">Status</span>			

Figure 4.3: Graphical user interface

### 4.3.1 GRID GENERATION MODULE

In the first step, the user is asked to provide information on:

- number of states and inputs,
- function handles for the nonlinear model and constraint functions,

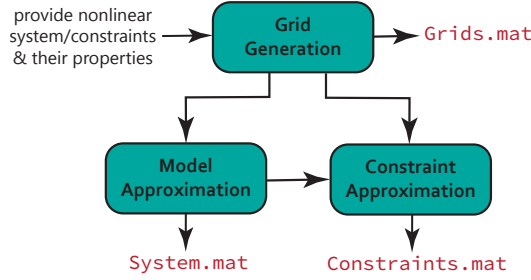


Figure 4.4: Toolbox architecture

## 4

- sampling time for forward Euler discretization if the model function is continuous-time (can be set to 0 if the provided function is discrete-time), and
- input and state bounds.

For the highly-nonlinear model or constraint functions in automated driving, analytical closed-form solutions for (4.5)–(4.6) often do not exist. Therefore, a grid is generated on  $\mathcal{D}$  to solve these optimization problems by minimizing the objective function across the grid points. For model approximation, the user can select among four methods described in [85]:

- **Domain-based:** *[points are directly sampled from  $\mathcal{D}$ ]*
  - **Uniform (U):** the points are generated by picking  $n_{\text{samp}}$  uniformly-spaced points along each axis in  $\mathcal{D}$ .
  - **Random (R):** a total of  $n_{\text{rand}}$  points are randomly selected from  $\mathcal{D}$ .
- **Trajectory-based:** *[ $n_{\text{sim}}$  open-loop simulations with  $n_{\text{step}}$  steps of  $F$  are run using random inputs from  $\mathcal{D}$ ]*
  - **Steady-state (S):** the initial state of each simulation is selected as the steady-state solution w.r.t. the initial input, i.e., it is assumed that each simulation starts from a steady state.
  - **Randomly-initiated (T):** the initial state of each simulation is randomly selected from  $\mathcal{D}$ .

For constraint approximation, the grid points are sampled from the whole domain  $\mathcal{D}$ . Since the region close to the boundary of the feasible region where  $G(x) = 1$  is of more interest, the constraint approximation grid is generated by combining a uniform grid (U) with a random grid (R) on the boundary region with width  $\epsilon_b$ , where  $|G(x) - 1| \leq \epsilon_b$ . The user can select the number of uniform and boundary grid points, as well as  $\epsilon_b$  in the user interface. After clicking on the “Generate Training Grids” button, the parameters are saved in the `params` struct and the model and constraint approximation grids, respectively SM and SC, are generated.

### 4.3.2 MODEL APPROXIMATION MODULE

In this module, the user is asked to provide  $(P^+, P^-)$  values for model approximation, for each state/input separately, i.e. for components of  $F$ . We solve (4.5) for the user-defined values of  $(P^+, P^-)$  by MATLAB's nonlinear least squares optimizer, `lsqnonlin`, using the trust-region-reflective algorithm, in view of the number of optimization variables. We solve the problem with  $n_{\text{multistart}}$  random initial guesses, where  $n_{\text{multistart}}$  is provided by the user, and we select the one with the lowest objective value as the optimal solution. By clicking on the "Approximate Prediction Model" button, the model approximation optimization problem is solved and the hybridized model is saved in the `system` struct.

### 4.3.3 CONSTRAINT APPROXIMATION MODULE

The third module approximates the nonlinear constraints using the desired approach and subregion types that can be selected by the user. Similar to the model approximation module, the boundary-based approach leads to a smooth problem, which H4MPC solves for the user-defined values of  $(R^+, R^-)$  or  $n_c$ . The boundary-based approach offers greater flexibility in fine-tuning the trade-off between encompassing the non-linear region and potentially infringing upon it, ultimately resulting in improved coverage of the non-convex region. However, if the application demands strict adherence to non-linear constraints, it is advisable to opt for the region-based approach in (4.6), which results in a non-differentiable objective function. The user can then select the parameter  $\gamma_c$  to adjust the weight factor in (4.6). H4MPC solves the approximation problem using the particle swarm optimizer in MATLAB, which does not require the problem to be differentiable where -based on extensive numerical experiments- we select the swarm size to be 10 times larger than the number of decision variables. The user can then click on the "Approximate Constraint" button, to get the hybridized constraints as the `const` struct.

## 4.4 CASE STUDY

In this section, two examples are investigated to showcase model and constraint approximations handled by H4MPC. The first example involves the single-track vehicle model from [88] as an illustrative model approximation example, and the second considers the Kamm circle constraint [89] as a function of the lateral and longitudinal slips to represent a non-convex feasible region. Multiple selections in H4MPC are tested to highlight the effect of various tuning parameters for hybridization.

### 4.4.1 NONLINEAR VEHICLE MODEL

The lateral dynamics of the single-track vehicle model in Fig. 4.5 is characterized by

$$\dot{r} = \frac{1}{I_{zz}} (l_f F_{yf} - l_r F_{yr}), \quad (4.7a)$$

$$\dot{\beta} = \arctan \left( \frac{F_{yf} + F_{yr}}{mv_x} - r \right), \quad (4.7b)$$

Table 4.1: System parameters and variables\*

System parameters			
Parameter	Definition	Value	Unit
$m$	Vehicle mass	1725	kg
$I_{zz}$	Inertia moment about z-axis	1300	kg/m <sup>2</sup>
$l_f$	CoG** to front axis distance	1.35	m
$l_r$	CoG to rear axis distance	1.15	m
$\mu$	Friction coefficient	1	–
$v_x$	Longitudinal velocity	20	m/s
$F_{zf}$	Normal load on the front axis	5000	N
$F_{zr}$	Normal load on the rear axis	5000	N
$B_\kappa$	Pacejka tire coefficients	11.4	–
$C_\kappa$		1.4	–
$B_\alpha$		10.0	–
$C_\alpha$		1.6	–
System variables			
Variable	Definition	Bound	Unit
$\beta$	Sideslip angle	[-0.3,0.3]	rad
$r$	Yaw rate	[-0.5, 0.5]	rad/s
$\delta$	Steering angle (road)	[-0.3, 0.3]	rad
$F_{xf}$	Longitudinal force on the front axis	[-5000, 0]	N
$F_{xr}$	Longitudinal force on the rear axis	[-5000, 5000]	N
$F_{yf}$	Lateral force on the front axis	[-5000, 5000]	N
$F_{yr}$	Lateral force on the rear axis	[-5000, 5000]	N
$\alpha_f$	Front slip angle	[-0.4,0.4]	rad
$\alpha_r$	Rear slip angle	[-0.4,0.4]	rad
$\kappa_f$	Front slip ratio	[-1,1]	–
$\kappa_r$	Rear slip ratio	[-1,1]	–
$s$	State vector $:= \begin{bmatrix} r & \beta \end{bmatrix}^T$	–	–
$u$	Input vector $:= \delta$	–	–

\*These kinematic parameters are from [88].

\*\*Center of Gravity

with the tire forces described by the Pacejka tire model [90] as

$$F_{xa} = F_{za} \mu \sin(C_\kappa \arctan(B_\kappa \kappa_a)), \quad (4.8a)$$

$$F_{ya} = F_{za} \mu \sin(C_\alpha \arctan(B_\alpha \alpha_a)), \quad (4.8b)$$

the  $\kappa_a$  representing the slip ratio on axle  $a \in \{f, r\}$ , and the slip angles being

$$\alpha_f = \arctan\left(\beta + \frac{l_f r}{v_x}\right) - \delta, \quad (4.9a)$$

$$\alpha_r = \arctan\left(\beta - \frac{l_r r}{v_x}\right). \quad (4.9b)$$

The system parameters are shown in Table 4.1. The tire forces should satisfy the tire saturation limits, i.e. Kamm circle constraint [89]

$$F_{xf}^2 + F_{yf}^2 \leq (\mu F_{zf})^2, \quad (4.10a)$$

$$F_{xr}^2 + F_{yr}^2 \leq (\mu F_{zr})^2. \quad (4.10b)$$



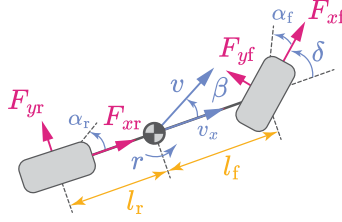


Figure 4.5: Single-track vehicle model.

#### 4.4.2 APPROXIMATION OF THE VEHICLE DYNAMICS

We use the vehicle model (4.7) as a case study to investigate model approximation with two states, represented by the variables  $r$  and  $\beta$ , and the input  $\delta$ . To showcase the impact of the parameters  $(P^+, P^-)$  on the level of complexity of the hybrid formulation, we examine two sets of parameter values:  $(P^+, P^-) \in \{(2, 2), (4, 3)\}$  for  $\dot{r}$  and  $(P^+, P^-) \in \{(2, 2), (5, 5)\}$  for  $\dot{\beta}$ .

As visualizing a three-dimensional input/state space can be not easy to read, we plot two specific cases in Fig. 4.6:  $\dot{r}(\beta, r)$  for  $\delta = 0$  rad and  $\dot{\beta}(\beta, \delta)$  for  $r = -0.3$  rad. For more clarity, we provide two cuts along different paths in each plot with their own distinguished colors. We compare the original nonlinear function with its MMPS approximations by using dashed lines for a simpler MMPS approximation and solid lines for a more complex one. Notably, the more complex MMPS approximation, which includes additional terms, provides a more accurate representation of the nonlinear function. For a more clear dynamic comparison, the phase portrait of the more accurate MMPS approximation is compared with the nonlinear one from [88] in Fig. 4.7 for three values of  $\delta$ , which shows that the MMPS approximation can generate sufficiently-accurate trajectories on the phase portrait as well.

#### 4.4.3 APPROXIMATION OF THE KAMM CIRCLE CONSTRAINT

To demonstrate the use of hybrid approximation for non-convex feasible regions, we provide the following example with a more intuitive interpretation: approximate the feasible region associated with the Kamm circle constraint defined in (4.10) within the  $\kappa$ - $\alpha$  plane as its two input/states. Therefore, we employ region-based approximation techniques while ensuring that the constraint violation error is maintained at zero. Figure 4.8 showcases the hybrid approximations using the ellipsoidal and MMPS methodologies for different values of  $n_e$  and  $(R^+, R^-)$ . Figure 4.8 illustrates the obtained hybrid approximations.

Additionally, to emphasize the significance of defining boundaries properly, we examine simpler cases with  $n_e = 2$  for the ellipsoidal approach and  $(R^+, R^-) = (2, 2)$  for the MMPS method under two scenarios: setting the boundary on the domain in terms of  $\alpha$  to either 0.1 or 0.4 radians. A comparison between the orange and purple approximations for both the ellipsoidal and MMPS approaches reveals that, as the optimizer minimizes the error across the domain, it converges to a more accurate approximation in the vicinity of the origin when  $|\alpha|$  is less than or equal to 0.1. However, when the domain is extended to  $|\alpha| \leq 0.4$ , the optimizer converges to an ellipsoidal or polytopic approximation of the feasible region close

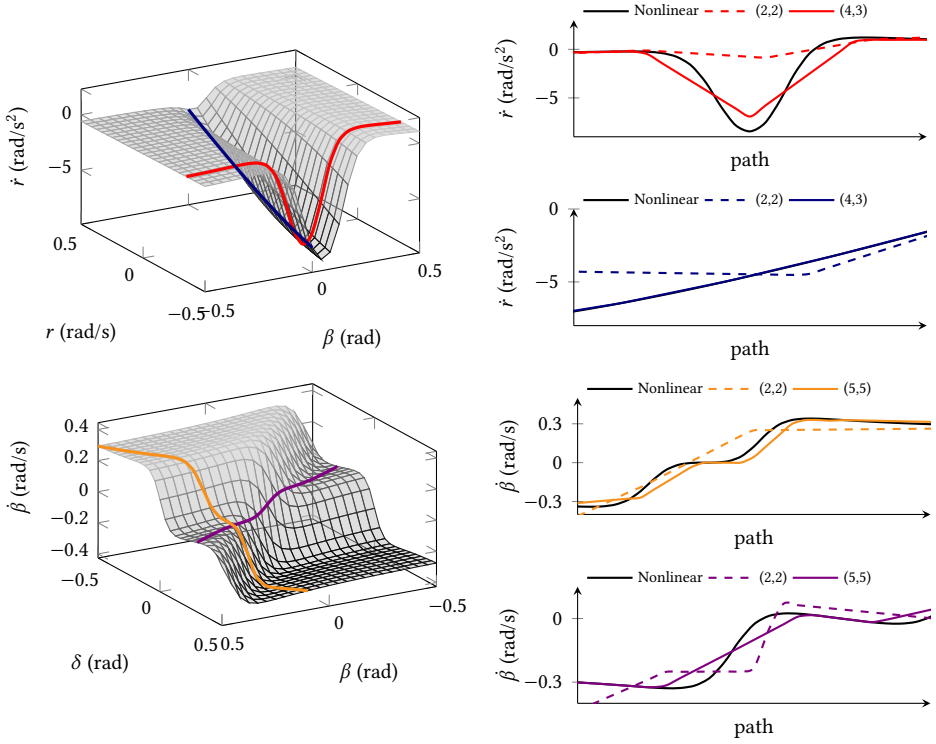


Figure 4.6: Comparison of two MMPS approximations of the vehicle model with different  $(P^+, P^-)$  values. For a more clear representation, the functions are plotted along four paths as 2-dimensional cuts of the 3-dimensional function representation.

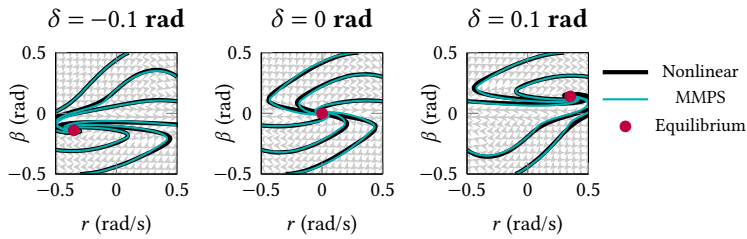


Figure 4.7: Phase portraits of the nonlinear and the MMPS approximation with  $(P^+, P^-) \in \{(4, 3), (5, 5)\}$ .

to the origin, which cannot adequately capture the complexity of the nonlinear constraint further from the origin. By increasing the complexity of the approximation, such as using five ellipsoids or  $(R^+, R^-) = (5, 5)$ , the optimizer finds hybrid approximations that provide better coverage of the feasible region. This improved approximation is represented in pink in both figures.

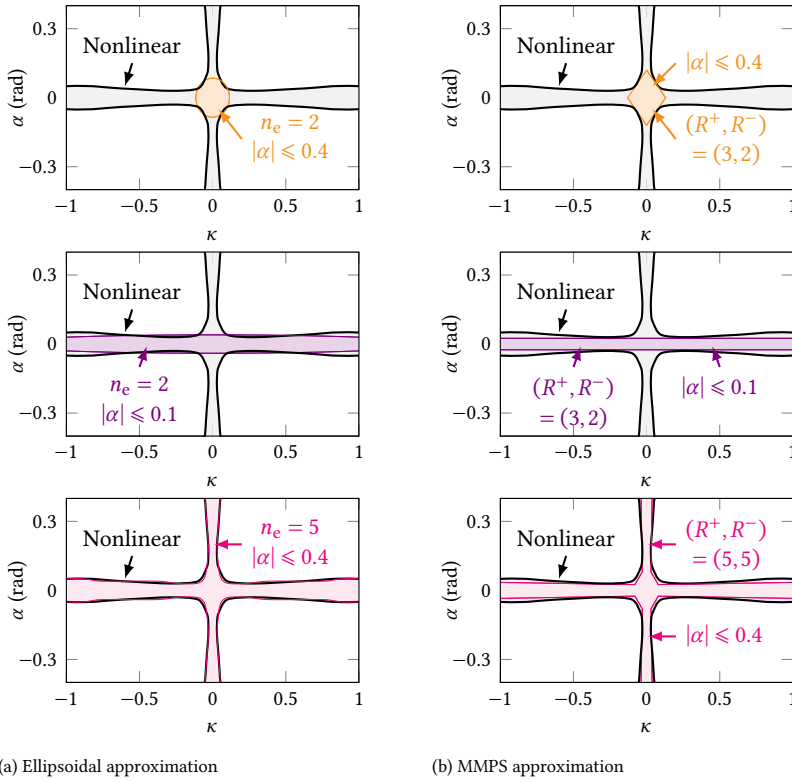


Figure 4.8: Illustration of MMPS and ellipsoidal approximation of the Kamm circles (4.10) as a function of  $\alpha$  and  $\kappa$ .

## 4.5 CONCLUSIONS

H4MPC is an open-source toolbox for hybridization of nonlinear model and constraints in NMPC to allow a hybrid formulation of the nonlinear optimization problem. The toolbox includes three modules and provides a user-friendly interface to allow the user to customize the approximation. In this chapter, a single-track vehicle model and the tire saturation limits were investigated as two examples to showcase multiple approximation approaches handled in H4MPC and to highlight the influence of their corresponding parameters. We expect the toolbox to be useful in a variety of applications such as automated driving or control of robotic systems. The H4MPC toolbox is freely available from <https://bit.ly/H4MPCV1>. The next versions of the toolbox will include controller design using the hybridized form of the nonlinear model and physics-based constraints to investigate the effect of approximation complexity on computation time of the NMPC optimization problem.



## 5

# SENSITIVITY ANALYSIS FOR PWA APPROXIMATIONS OF NLPs

5

*To be a theorist you have to admit the possibility of being wrong – the provisionality of knowledge – and you know you cannot spin your way out of a theoretical problem.*

— Yanis Varoufakis, *And the Weak Suffer What They Must?*

*NonLinear Program (NLP)s are prevalent in optimization-based control of nonlinear systems. Solving general NLPs is computationally expensive, necessitating the development of fast hardware or tractable suboptimal approximations. This chapter investigates the sensitivity of the solutions of NLPs with polytopic constraints when the nonlinear continuous objective function is approximated by a PieceWise Affine (PWA) counterpart. By leveraging perturbation analysis using a convex modulus, we derive guaranteed bounds on the distance between the optimal solution of the original polytopically-constrained NLP and that of its approximated formulation. Our approach aids in determining criteria for achieving desired solution bounds. Two case studies on the Eggholder function and control of an inverted pendulum demonstrate the theoretical results.*

## 5.1 INTRODUCTION

NLPs are commonly encountered in optimization-based control of nonlinear systems, e.g. Nonlinear Model Predictive Control (NMPC) [93]. Solving non-convex NLPs is intractable, posing a great challenge in applying optimization-based controllers in real-time operations, especially for systems having fast dynamics. Various solutions have been proposed to address this issue, such as adaptive problem formulations [94], learning-based methods [95], and sensitivity analysis of NLPs [96].

This chapter has been published in IEEE Control Systems Letters [92].

PWA approximations are widely used due to their tractability [65, 70]. To obtain a continuous PWA approximation, min and max operators can be used to maintain continuity and to resolve numerical issues in the resulting optimization problem [28, 97, 98]. The approximated problem can be used to obtain a suboptimal solution [99], whose optimality highly depends on the accuracy of the approximation. For example, a warm start of a non-convex NLP can be obtained by solving the approximated optimization problem [100, 101]. Optimality guarantees of such approaches can be derived using sensitivity analysis, establishing an upper bound on the distance between the original solutions and the approximated ones. As a result, by finding a subset of the decision space around the approximated solution, one can sample a structured or random warm start to solve the original non-convex NLP more efficiently.

Quantitative bounds on the distance between the original and the approximated solutions have been studied in the sensitivity analysis of quadratic [102] and convex [103] optimization problems. Regarding NLPs, there exist several results on their sensitivity to the parameters in the optimization formulation [104, 105] and the initial solution [106]. In addition, optimality and dissipativity conditions for the perturbed convex NLP problem have also been established [107, 108]. For a more extensive study, the reader can refer to [109]. However, obtaining quantitative bounds on the distance between the solutions of a non-convex NLP and its PWA approximation is still a gap that needs to be filled, and our work addresses this problem.

## 5

In this chapter, we present a method to bound the solutions of a polytopically-constrained non-convex NLP problem using a continuous PWA approximation of the nonlinear objective function. We employ the Max-Min-Plus-Scaling (MMPS) formulation of continuous PWA functions in [28] to construct a piecewise convex approximation formalism. Leveraging some results in [103], we derive guaranteed bounds on the distance between the original and the approximated optimal solution. Moreover, our approach can not only establish such bounds but also aid in determining necessary criteria during the approximation stage to attain a desired solution bound. To demonstrate the theoretical findings, we present a case study on the Eggholder function [110], a renowned benchmark optimization problem with multiple local minima.

The rest of the chapter is organized as follows. In Section 5.2, we present preliminaries regarding the sensitivity analysis of NLPs and the PWA approximation of nonlinear functions. Section 5.3 formulates the problem and Section 5.4 elaborates our proposed approach to theoretically compute the confidence radius for the local minima of the corresponding approximated function. In Section 5.5, we then demonstrate the derived confidence radius through a case study on the Eggholder function and we apply our analysis to an NMPC optimization example. Section 5.6 concludes this chapter.

*Notation:* For a positive integer  $P$ , we use  $\mathcal{I}_P$  to denote the set  $\{1, 2, \dots, P\}$ . For a connected set  $\mathcal{D} \subseteq \mathbb{R}^n$ , the diameter of  $\mathcal{D}$  is defined as  $\text{diam}(\mathcal{D}) := \max_{x_1, x_2 \in \mathcal{D}} \|x_1 - x_2\|$ , where  $\|\cdot\|$  is the Euclidean or the 2-norm.

## 5.2 PRELIMINARIES

### 5.2.1 REPRESENTATION OF CONTINUOUS PWA FUNCTIONS

We start with formally defining a continuous PWA function which will be frequently used throughout this chapter.

**Definition 5.1** (Continuous PWA function [111]). *A scalar-valued function  $f : \mathcal{D} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$  is said to be a continuous PWA function if and only if the following conditions hold:*

1. *the domain space  $\mathcal{D}$  is divided into a finite number of closed polyhedral regions  $\mathcal{R}_1, \dots, \mathcal{R}_R$  with non-overlapping interiors,*
2. *for each  $r \in \mathcal{I}_R$ ,  $f$  can be expressed as*

$$f(x) = \alpha_r^T x + \beta_r \quad \text{if } x \in \mathcal{R}_r,$$

*with  $\alpha_r \in \mathbb{R}^n$  and  $\beta_r \in \mathbb{R}$ , and*

3.  *$f$  is continuous on the boundary between any pair of regions.*

PWA functions can be expressed in different forms, among which the MMPS form is powerful for decomposing PWA functions.

**Theorem 5.1** (MMPS representation [28]). *For a scalar-valued continuous PWA function  $f$  as in Definition 5.1, there exist non-empty index sets  $\mathcal{I}_p$  and  $\mathcal{I}_{Q_p}$  such that*

$$f(x) = \min_{p \in \mathcal{I}_p} \max_{q \in \mathcal{I}_{Q_p}} (a_{p,q}^T x + b_{p,q}), \quad (5.1)$$

*for real numbers  $b_{p,q}$  and vectors  $a_{p,q} \in \mathbb{R}^n$ .*

For convenience, we define the  $p$ -th local convex segment of  $f$  as

$$f_p(x) := \max_{q \in \mathcal{I}_{Q_p}} (a_{p,q}^T x + b_{p,q}), \quad (5.2)$$

where  $f_p$  is convex since it is defined as the maximum of a finite number of affine functions and its domain is also convex. In addition, we define the region  $\mathcal{C}_{p,q}$  in which a certain affine function is activated and the region  $\mathcal{C}_{p,\cdot}$  in which a convex PWA function is activated, that is,

$$\mathcal{C}_{p,q} := \{x \in \mathcal{D} \mid f(x) = a_{p,q}^T x + b_{p,q}\}, \quad (5.3a)$$

$$\mathcal{C}_{p,\cdot} := \{x \in \mathcal{D} \mid f(x) = f_p(x)\}. \quad (5.3b)$$

Further, we have

$$\mathcal{C}_{p,\cdot} = \bigcup_{q=1}^{Q_p} \mathcal{C}_{p,q}.$$

Lastly, we define  $\sigma_p : \mathcal{C}_{p,\cdot} \rightarrow \mathcal{I}_{Q_p}$  as the region index function for  $f_p$  as

$$\sigma_p(x) = q \iff x \in \mathcal{C}_{p,q}. \quad (5.4)$$

### 5.2.2 SENSITIVITY OF THE CONVEX OPTIMIZATION PROBLEM

The convexity modulus [103], being used to quantify convexity, is useful in the sensitivity analysis of convex functions. In the following, we define the convexity modulus specifically for  $f_p$ , the  $p$ -th convex segment of  $f$ , and its domain  $C_{p..}$ .

**Definition 5.2** (Convexity modulus [103]). *For a given convex function  $f_p$ , the convexity modulus  $h_1 : [0, +\infty) \rightarrow [0, +\infty)$  over the domain  $C_{p..}$  is defined as*

$$h_1(\gamma) := \begin{cases} \inf_{\substack{v, w \in C_{p..} \\ \|v - w\| = \gamma}} J(v, w) & \text{if } \gamma < \text{diam}(C_{p..}) \\ +\infty & \text{if } \gamma \geq \text{diam}(C_{p..}) \end{cases}, \quad (5.5)$$

where  $v$  and  $w$  are two points in  $C_{p..}$  satisfying  $\|v - w\| = \gamma$ , and  $J(v, w)$  is given as

$$J(v, w) = \frac{f_p(v) + f_p(w)}{2} - f_p\left(\frac{v + w}{2}\right). \quad (5.6)$$

5

**Theorem 5.2** (Theorem 4.5 in [103]). *Suppose that  $f_p : C_{p..} \rightarrow \mathbb{R}$  is a scalar-valued convex function and  $\delta_p : C_{p..} \rightarrow \mathbb{R}$  is an arbitrary function satisfying*

$$\sup_{x \in C_{p..}} |\delta_p(x)| = \Delta_p < \infty. \quad (5.7)$$

Let  $x_p^*$  be any global infimizer of  $f_p$  and  $\hat{x}_p^*$  be any global infimizer of  $\hat{f}_p = f_p + \delta_p$ . Then

$$\|\hat{x}_p^* - x_p^*\| \leq h_1^{-1}(2\Delta_p), \quad (5.8)$$

where  $h_1$  is the convexity modulus in Definition 5.2.

For a more compact notation, we call the right-hand side of (5.8), the confidence radius, defined as follows:

**Definition 5.3** (Confidence radius). *For a given function  $\hat{f}_p : C_{p..} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$  approximating the function  $f_p : C_{p..} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$  with the maximal approximation error  $\Delta_p$  from (5.7), the confidence radius is the upper-bound on the distance between  $\hat{x}_p^*$ , the global minimizer of  $\hat{f}_p$ , and  $x_p^*$ , the global minimizer of  $f_p$ , and is obtained by*

$$\chi = h_1^{-1}(2\Delta_p),$$

where  $h_1$  is the convexity modulus in Definition 5.2.

**Proposition 5.1** (Proposition 2.2, 2.5 in [103]). *Given a convex function  $f_p$  on a compact domain  $C_{p..}$  with convexity modulus  $h_1$  defined as (5.5), the following hold:*

1.  $h_1$  is left-continuous on  $(0, \text{diam}(C_{p..}))$ , and
2. for  $\gamma_1, \gamma_2 \in [0, \text{diam}(C_{p..})]$ , if  $\gamma_1 < \gamma_2$ , then  $h_1(\gamma_1)/\gamma_1 \leq h_1(\gamma_2)/\gamma_2$ .



### 5.3 CONTINUOUS PWA APPROXIMATION OF NLPs

Consider an NLP with polytopic constraints

$$\min_{x \in \mathcal{X}} F(x), \quad (5.9)$$

where  $F : D \subset \mathbb{R}^n \rightarrow \mathbb{R}$  is the nonlinear objective function and  $\mathcal{X} \subseteq D$  is the polytopic feasible region. From now on, we assume that the domain  $D$  is compact. We approximate  $F$  by a continuous PWA function  $f$  of the MMPS form (5.1) via solving the approximation problem

$$\min_{\mathcal{A}, \mathcal{B}} \int_D \left| F(x) - \min_{p \in I_P} \max_{q \in I_{Q_p}} (a_{p,q}^T x + b_{p,q}) \right| dx, \quad (5.10)$$

to minimize the absolute approximation error, where the ordered sets  $\mathcal{A}$  and  $\mathcal{B}$  respectively collect  $a_{p,q}$  and  $b_{p,q}$ .

**Example 5.1.** Figure 5.1 shows a 1-dimensional example of approximating a nonlinear objective function  $F$  by a continuous PWA function  $f$  using the MMPS form (5.1) as

$$F(x) \approx f(x) = \min \left( \underbrace{\max(f_{1,1}, f_{1,2})}_{f_{1,.}}, \underbrace{\max(f_{2,1}, f_{2,2})}_{f_{2,.}}, f_{3,.} \right),$$

with  $P = 3$ ,  $Q_1 = Q_2 = 2$ , and  $Q_3 = 1$ . The convex segments of  $f$  are shown by  $f_{p,.}$  which give the maximum value among  $Q_p$  affine functions  $f_{p,q}$ ,  $q \in I_{Q_p}$ . The subregions  $C_{p,q}$  are shown in the same color as their corresponding active affine functions,  $f_{p,q}$ . In this 1-dimensional example,  $\text{diam}(C_{p,.})$  is the distance between the upper and lower bounds of  $C_{p,.}$  on the  $x$ -axis.

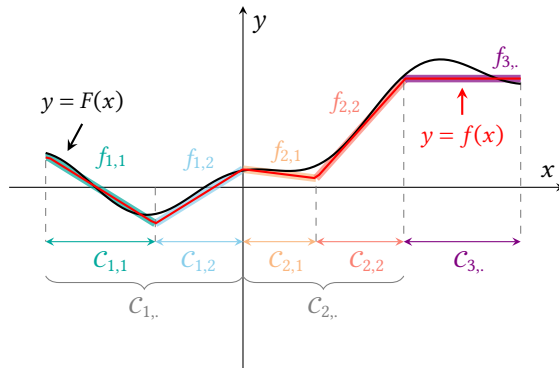


Figure 5.1: A conceptual example of approximating a nonlinear function  $F$  with a continuous PWA approximation  $f$  using the MMPS form in (5.1).

## 5.4 THEORETICAL ANALYSIS

The main result of this section is the sensitivity bound in Theorem 5.3. Before, we present some building blocks about continuity and boundedness of the convexity modulus (cf. Lemma 5.1, 5.2 and 5.3).

**Lemma 5.1.** *For a convex PWA function  $f_p$  expressed by (5.2),  $\partial h_1 / \partial \gamma$  is a piecewise constant function.*

*Proof.* For compactness, let us denote

$$i = \sigma_p(v), \quad j = \sigma_p(w), \quad k = \sigma_p\left(\frac{v+w}{2}\right),$$

with  $i, j, k \in \mathcal{I}_{Q_p}$ . For  $f_p$  expressed by (5.2), the function  $J$  in (5.6) can be written as

$$J(v, w) = \frac{a_{p,i}^T - a_{p,k}^T}{2} v + \frac{a_{p,j}^T - a_{p,k}^T}{2} w + \frac{b_{p,i} + b_{p,j} - 2b_{p,k}}{2}. \quad (5.11)$$

The necessary Lagrange conditions for optimality at  $(v^*, w^*)$  in (5.5) state that there must exist  $\mu \in \mathbb{R}$  that satisfies the following simultaneously:

$$\nabla J(v^*, w^*) + \mu \nabla(\|v - w\| - \gamma) \Big|_{\substack{v=v^* \\ w=w^*}} = 0, \quad (5.12a)$$

$$\|v^* - w^*\| = \gamma. \quad (5.12b)$$

By calculating the gradient of (5.11), we have that (5.12) becomes

$$\frac{a_{p,i}^T - a_{p,k}^T}{2} + \mu \frac{v^* - w^*}{\|v^* - w^*\|} = 0, \quad (5.13a)$$

$$\frac{a_{p,j}^T - a_{p,k}^T}{2} + \mu \frac{w^* - v^*}{\|v^* - w^*\|} = 0, \quad (5.13b)$$

$$\|v^* - w^*\| = \gamma, \quad (5.13c)$$

which implies the existence of  $\mu \in \mathbb{R}$  satisfying

$$\frac{a_{p,i}^T - a_{p,j}^T}{2} + 2\mu \frac{v^* - w^*}{\gamma} = 0. \quad (5.14)$$

Note that  $\partial h_1 / \partial \gamma$  is equal to  $\partial J / \partial \gamma$ , except where the indices  $i, j$ , and  $k$  change. At such points,  $h_1$  is not differentiable with respect to  $\gamma$ , which does not conflict with  $\partial h_1 / \partial \gamma$  being a piecewise constant function. To find the slope of  $h_1$  where it exists, the chain rule can be applied as

$$\frac{\partial h_1}{\partial \gamma} = \frac{\partial J}{\partial v^*} \Big/ \frac{\partial \gamma}{\partial v^*} + \frac{\partial J}{\partial w^*} \Big/ \frac{\partial \gamma}{\partial w^*} = \frac{a_{p,i}^T - a_{p,j}^T}{2} \frac{\|v^* - w^*\|}{v^* - w^*},$$

which, considering (5.14), leads to

$$\frac{\partial h_1}{\partial \gamma} = -2\mu, \quad (5.15)$$

which implies that  $\partial h_1 / \partial \gamma$  is a function of the  $a_{p,i}^T - a_{p,j}^T$  values.  $\square$

**Lemma 5.2.** *For a convex PWA function  $f_p$  expressed by (5.2), the convexity modulus  $h_1$  in (5.5) is continuous on  $[0, \text{diam}(C_{p,\cdot}))$ .*

*Proof.* From Proposition 5.1, we know that  $h_1$  is left-continuous on  $[0, +\infty)$ . Seeking a contradiction, let us assume that  $h_1$  is not right-continuous in  $\gamma_0 \in [0, \text{diam}(C_{p,\cdot}))$ , hence,

$$\lim_{\gamma \rightarrow \gamma_0^+} h_1(\gamma) \neq h_1(\gamma_0).$$

The monotonicity property of  $h_1$  in Proposition 5.1 requires

$$\lim_{\gamma \rightarrow \gamma_0^+} h_1(\gamma) > h_1(\gamma_0).$$

Therefore, without loss of generality, we assume there exists a gap  $\epsilon_0 > 0$  and a point  $\gamma_0 < \gamma_0^+ < \text{diam}(C_{p,\cdot})$  such that

$$h_1(\gamma_0^+) = h_1(\gamma_0) + \frac{\partial h_1}{\partial \gamma} \Big|_{\gamma=\gamma_0} (\gamma_0^+ - \gamma_0) + \epsilon_0. \quad (5.16)$$

Using (5.5), we define the points  $v_0$ ,  $w_0$ , and  $w_0^+$  such that

$$h_1(\gamma_0) = \inf_{\substack{v, w \in C_{p,\cdot} \\ \|v-w\|=\gamma_0}} J(v, w) = J(v_0, w_0),$$

and  $\|v_0 - w_0^+\| = \gamma_0^+$ . Considering the optimality property in (5.5), we have

$$h_1(\gamma_0^+) \leq J(v_0, w_0^+),$$

and knowing that  $J$  is a continuous function by definition, we can deduce

$$h_1(\gamma_0^+) \leq h_1(\gamma_0) + \rho(w_0^+ - w_0),$$

where  $\rho : C_{p,\cdot} \rightarrow \mathbb{R}$  is a function with the following property:

$$\lim_{v \rightarrow 0^+} \rho(v) = 0. \quad (5.17)$$

Substituting (5.16) into (5.17) and taking the limit on both sides when  $\gamma_0^+$  approaches  $\gamma_0$  leads to

$$h_1(\gamma_0) + \epsilon_0 \leq h_1(\gamma_0),$$

which contradicts the fact that  $\epsilon_0 > 0$ . Therefore,  $h_1$  is right-continuous on  $[0, \text{diam}(C_{p,\cdot}))$ .  $\square$

**Proposition 5.2.** *For a convex PWA function  $f_p$  expressed by (5.2), we have*

$$h_1(\gamma) = 0, \quad \forall \gamma \leq \max_{q \in I_{Q_p}} \{ \text{diam}(C_{p,q}) \}.$$

*Proof.* By seeking a contradiction, let us assume that

$$\exists \gamma_0 \leq \max_{q \in I_{Qp}} \{ \text{diam}(C_{p,q}) \}, \quad \text{such that} \quad h_1(\gamma_0) > 0,$$

with the corresponding optimal points  $v_0^*$  and  $w_0^*$  from (5.5) such that

$$h_1(\gamma_0) = \inf_{\substack{v, w \in C_{p..} \\ \|v-w\|=\gamma_0}} J(v, w) = J(v_0^*, w_0^*).$$

Let us select two points,  $v_0$  and  $w_0$ , on the largest subregion in  $C_{p..}$  such that

$$\|v_0 - w_0\| = \gamma_0.$$

which results in  $J(v_0, w_0) = 0$ . Considering the optimality property in (5.5), we have

$$J(v_0^*, w_0^*) \leq J(v_0, w_0),$$

which contradicts the initial assumption that  $h_1(\gamma_0) > 0$  and  $h_1(\gamma_0) = J(v_0^*, w_0^*)$ .  $\square$

**Lemma 5.3.** *For a convex PWA function  $f_p$  expressed by (5.2), the convexity modulus  $h_1$  is bounded by  $\hat{h}_1 \leq h_1$ , with*

$$\hat{h}_1(\gamma) := \begin{cases} 0 & \text{if } \gamma < \text{diam}(C_{p..}) \\ c_1 \gamma + c_0 & \text{if } \gamma \geq \text{diam}(C_{p..}) \end{cases}, \quad (5.18)$$

where

$$c_1 = \min_{j \in I_{Qp}} \left\{ \frac{a_{p,i}^T - a_{p,j}^T}{2} \right\}, \quad (5.19a)$$

$$\text{s.t.} \quad i = \arg \max_{q \in I_{Qp}} \text{diam}(C_{p,q}), \quad (5.19b)$$

$$C_{p,i} \cap C_{p,j} \neq \emptyset, \quad (5.19c)$$

and  $c_0 = c_1 \text{diam}(C_{p..})$ .

*Proof.* This can be directly deduced from Proposition 5.2, considering the continuity of  $h_1$  from Lemma 5.2, the piecewise-constant property of  $\partial h_1 / \partial \gamma$  from Lemma 5.1, and the increasing property of  $h_1$  from Proposition 5.1.  $\square$

We are now in the position to state our main result:

**Theorem 5.3.** *Let  $F : \mathcal{D} \rightarrow \mathbb{R}$  be a scalar-valued objective function and let  $f$  be a continuous PWA function as in Definition 5.1 that approximates  $F$  with bounded approximation error  $\delta = f - F$ . Let  $f_p$  in (5.2) be the local convex segment of  $f$  in its MMPS form (5.1) on the set  $C_{p..}$ , and let  $\delta_p : C_{p..} \rightarrow \mathbb{R}$  be the corresponding approximation error bounded by*

$$\sup_{x \in C_{p..}} |\delta_p(x)| = \Delta_p < \infty.$$

Let  $x_p^*$  be any global minimizer of  $f_p$  and  $\hat{x}_p^*$  be any global minimizer of  $F$  on  $C_{p,\cdot}$ . Then, the following condition holds:

$$\|\hat{x}_p^* - x_p^*\| \leq \frac{2\Delta_p}{c_1} + \max_{q \in I_{Q_p}} \{ \text{diam}(C_{p,q}) \}, \quad (5.20)$$

where  $c_1$  is defined in (5.19).

*Proof.* This can be directly concluded by extending Theorem 5.2 via considering Proposition 5.2 and Lemma 5.3.  $\square$

## 5.5 CASE STUDY

To showcase the application of Theorem 5.3, we select the 1-dimensional cut of the well-known Eggholder test function [110] at  $x_2 = 0$  given by

$$F(x) = -47 \sin \left( \sqrt{\left| \frac{x}{2} + 47 \right|} \right) - x \sin \left( \sqrt{|x - 47|} \right).$$

We approximate  $F$  by a continuous PWA function  $f$  that we arbitrarily take as

$$f(x) = \min_{p \in I_5} (f_{p,\cdot}).$$

with local convex segments

$$f_{1,\cdot} = \max_{q \in I_3} (f_{1,q}), \quad C_{1,\cdot} = [-512, -385], \quad (5.21a)$$

$$f_{2,\cdot} = f_{2,1}, \quad C_{2,\cdot} = [-385, -330], \quad (5.21b)$$

$$f_{3,\cdot} = \max_{q \in I_3} (f_{3,q}), \quad C_{3,\cdot} = [-330, -180], \quad (5.21c)$$

$$f_{4,\cdot} = \max_{q \in I_3} (f_{4,q}), \quad C_{4,\cdot} = [-180, 180], \quad (5.21d)$$

$$f_{5,\cdot} = f_{5,1}, \quad C_{5,\cdot} = [180, 512]. \quad (5.21e)$$

Figure 5.2 shows the plots for the nonlinear objective function  $F$  and its PWA approximation  $f$ . The subregions  $C_{p,\cdot}$  with  $p \in I_5$  are illustrated by different colors.

Theorem 5.3 can be used in two ways:

1. guaranteeing bounds on the distance between the regional minima of  $F$  and  $f$  on each subregion, given a bound on the approximation error, and
2. finding the required criteria for the approximation to allow guaranteeing a desired bound on the distance between these minima, which we refer to as the confidence radius.

We discuss each case separately by focusing on the approximation on  $C_{3,\cdot}$ .

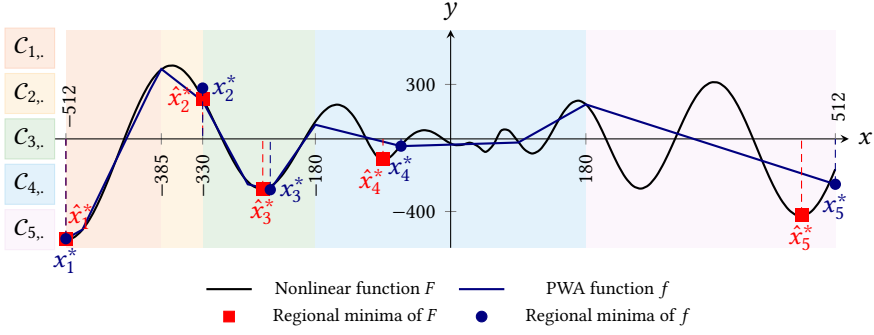


Figure 5.2: Plots of the nonlinear objective function  $F$  and its PWA approximation  $f$ .

### 5.5.1 CASE 1: FINDING THE CONFIDENCE RADII

Figure 5.3a shows  $F$  and two PWA approximations on  $C_{3,}$  with two approximation errors. The first is  $f_3^{(1)}$ , which divides  $C_{3,}$  into 3 subregions with maximum approximation error  $\Delta_3^{(1)} = 19.9$ . The second approximation is  $f_3^{(2)}$  with 8 affine pieces, the maximum error  $\Delta_3^{(2)} = 2.6$ . The approximations are given by

$$f_3^{(1)}(x) = \max \left\{ \begin{array}{l} -7.8x - 2365.7 \\ -0.9x - 501.2 \\ 6.1x + 1176.1 \end{array} \right\}, \quad f_3^{(2)}(x) = \max \left\{ \begin{array}{l} -8.6x - 2613.1 \\ -6.8x - 2095.6 \\ -4.6x - 1477.9 \\ -2.2x - 829.8 \\ 0.3x - 191.6 \\ 2.8x + 412.5 \\ 5.1x + 944 \\ 6.9x + 1348.1 \end{array} \right\}.$$

The inverses of the corresponding convexity moduli are shown in Fig. 5.3b in the same color as their corresponding  $f$  in Fig. 5.3a, where  $\chi$  is the confidence radius in Definition 5.3.

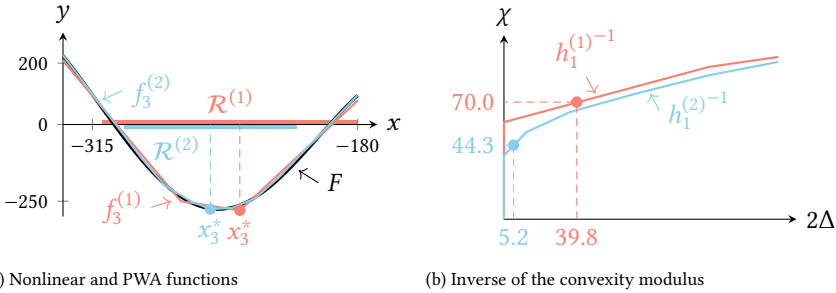


Figure 5.3: Comparison of two different PWA approximations of the nonlinear function on  $C_{3,}$ .

Using Theorem 5.3, the confidence radius for  $f_3^{(1)}$  by is obtained by  $\chi^{(1)} = 70$ , which

is the same value obtained by finding  $h_1$  and its inverse function using (5.5), which is presented in Fig. 5.3b. The same process can be performed for the second approximation,  $f_3^{(2)}$ , which gives  $\chi^{(2)} = 44.3$ . Note that Theorem 5.3 is more conservative for larger values of  $\Delta$ , compared to directly using the definition of  $h_1$ . For instance, if  $\Delta_3^{(2)} = 12.5$ , employing Theorem 5.3 leads to  $\chi^{(2)} = 65.7$ , while the computed confidence radius using  $h_1$  is 58.4. The areas within the confidence radii for the PWA approximations are highlighted on the  $x$ -axis in Fig. 5.3a as well.

### 5.5.2 CASE 2: FINDING THE APPROXIMATION CRITERIA

In this case, we approach the problem from another direction: we select a desired confidence radius  $\chi^{(3)}$  and find the required criteria for the corresponding approximated function,  $f_3^{(3)}$ . Let the desired  $\chi^{(3)} = 15$ ; then,

$$\frac{2\Delta_3^{(3)}}{c_1} + \max_{q \in I_{Qp}} \{ \text{diam}(C_{3,q}) \} \leq 10,$$

which means the diameter of the largest subregion  $C_{3,\cdot}$  must be smaller than 10. Firstly, given that  $\text{diam}(C_{3,\cdot}) = 150$ , it can be concluded that the PWA approximation requires at least 10 partitions. We can then start the approximation by partitioning  $C_{p,\cdot}$  into 15 subregions with the same diameter and find the lowest possible error bound  $\Delta_3^{(3)}$  for the approximation, which is obtained as 2.83 with  $c_1 = 0.0072$ . For this approximation,  $\chi^{(3)}$  already exceeds  $\text{diam}(C_{3,\cdot})$ .

To improve upon this example, we add another partition to reduce the largest partition diameter further and this time we do not aim at partitions of  $C_{3,\cdot}$  with the same diameter, but require

$$\max_{q \in I_{16}} \{ \text{diam}(C_{3,q}) \} \leq 10.$$

We find  $\Delta_3^{(3)} = 2.47$  with  $c_1 = 1.03$  and

$$\max_{q \in I_{16}} \{ \text{diam}(C_{3,q}) \} = 9.4.$$

For this values, we obtain  $\chi^{(3)} = 14.24$ . In case this value is acceptable, we can use the corresponding PWA approximation while ensuring that the minimizer of  $F$  on  $C_{3,\cdot}$  lies in a ball of radius 14.24 around the minimizer of  $f^{(3)}$ . In case a tighter confidence radius is desired, the same procedure can be followed by adding more subregions.

### 5.5.3 APPLICATION FOR NMPC OPTIMIZATION

To showcase the application of our proposed method in PWA approximation to control optimization problems, we use an inverted-pendulum NMPC problem as in [93]. Considering a prediction horizon of  $N_p = 2$  and zero initial rotation velocity  $\theta$  at time step  $k$ , the objective function  $J_{\text{NMPC}}$  can be formulated as a function of the measured pendulum angle  $\theta_k$ , and the control inputs over the horizon  $u_k$  and  $u_{k+1}$ . For instance, for  $\theta_k = 0$  we have

$$J_{\text{NMPC}}(0, u_k, u_{k+1}) = \sqrt{(0.02u_k + \pi)^2 + 2\pi^2} + 0.02 \sqrt{u_k^2 + (u_k + u_{k+1})^2} + 0.01 \sqrt{u_k^2 + u_{k+1}^2}.$$

Moreover, the feasible region is defined as the box constraint  $|u_{k+i-1}| \leq 20N$ ,  $i \in \mathcal{I}_2$  with  $\text{diam}(C_{1,\cdot}) = 56.4$ .

We approximate  $J_{\text{NMPC}}$  by two convex MMPS forms  $f^{(1)}$  and  $f^{(2)}$  – with  $P^{(1)} = P^{(2)} = 1$  in (5.1) – with different complexities in terms of the number of affine functions as

$$\begin{aligned} Q^{(1)} &= 4, \quad \Delta^{(1)} = 0.19, \quad \max_{q \in \mathcal{I}_4} \left\{ \text{diam} \left( C_{1,q}^{(2)} \right) \right\} = 28.2, \\ Q^{(2)} &= 24, \quad \Delta^{(2)} = 0.01, \quad \max_{q \in \mathcal{I}_{24}} \left\{ \text{diam} \left( C_{1,q}^{(2)} \right) \right\} = 14.6. \end{aligned}$$

The inverse of the convexity modulus and the corresponding confidence regions for both approximations are shown in Fig. 5.4. While  $f^{(1)}$  has a low approximation error, its complexity level does not allow to guarantee a confidence radius lower than the diameter of the feasible region. However, the more accurate approximation  $f^{(2)}$  guarantees a smaller confidence radius. Moreover, a general approximation criterion can be obtained, similar to the Eggholder NLP example, for an NMPC problem. In this case, it can be observed from (5.20) and Fig. 5.4a that  $\chi$  is lower-bounded by the maximum subregion diameter. Therefore, if a particular confidence radius is desired, the approximation problem (5.10) can be solved while imposing constraints on the diameter of subregions, e.g. an upper bound on the maximum subregion diameter.

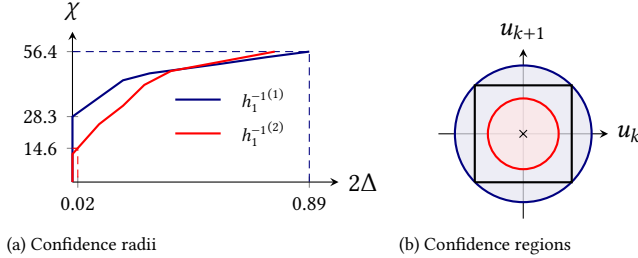


Figure 5.4: Comparison of two PWA approximations of NMPC.

## 5.6 CONCLUSIONS

This chapter has introduced a novel approach for bounding the minimizers of polytopically-constrained NLPs with nonlinear continuous objective functions using continuous PWA function approximations. We have leveraged the continuity of the PWA approximations resulting from employing an MMPS formalism to construct a locally-convex representation of the PWA approximation, thus facilitating the derivation of guaranteed bounds on the distance between the original and the approximated optimal solutions of the NLP by considering the maximal approximation error. Our approach offers a practical tool for determining criteria to achieve desired solution bounds. The effectiveness of the method has been demonstrated through a case study on the Eggholder function, highlighting the practical application of the proposed method and its potential impact in optimization and control.



For future work, our primary objective is to conduct an in-depth analysis of the conservatism of our approach and reducing it by refining our sensitivity analysis addressing global minima on the whole domain, as well as extending our method to NLPs with non-convex constraints. Moreover, we aim at conducting a comprehensive analysis to gain insight into the impacts of the improved computational efficiency through PWA approximation in light of the corresponding solution bounds. Finally, investigating the effects of probabilistic error bounds would also be an interesting direction to help integrating our approach into learning-based and data-driven applications.



# II

## COLLISION AVOIDANCE IN AUTOMATED DRIVING



## 6

# DESIGN AND NUMERICAL ANALYSIS OF HYBRIDIZATION BENCHMARK FOR NMPC

*Science does not aim at establishing immutable truths and eternal dogmas; its aim is to approach the truth by successive approximations, without claiming that at any stage final and complete accuracy has been achieved.*

— Bertrand Russell, *Sceptical Essays*

6

*Despite the extensive application of nonlinear Model Predictive Control (MPC) in automated driving, balancing its computational efficiency with its control performance and constraint satisfaction remains a challenge in emergency scenarios: in such situations, sub-optimal but computationally rapid responses are more valuable than optimal responses obtained after long computations. This chapter introduces a hybridization approach for efficient approximation of the nonlinear vehicle dynamics and of its non-convex constraints, e.g., arising during emergency evasive maneuvers. Hybridization, i.e., the use of hybrid systems modeling, allows to reformulate the nonlinear MPC problem as a hybrid MPC problem. Max-Min-Plus-Scaling (MMPS) hybrid modeling is used to approximate the nonlinear vehicle dynamics. Meanwhile, different formulations for constraint approximation are presented, and various grid-generation methods are compared to solve these approximation problems. Among these, two novel grid types are introduced to structurally include the influence of the nonlinear vehicle dynamics on the grid point distributions in the state domain. Overall, the work presents and compares three hybrid models and four hybrid constraints for efficient MPC synthesis and offers guidelines for implementation of the presented hybridization framework in other applications.*

## 6.1 INTRODUCTION

MPC has become increasingly popular in automated driving research over the past few decades [112]. This is mainly due to its capability to handle constraints and its ability to adapt to the system by performing controller synthesis in a rolling-horizon optimization-based manner. However, high computation loads remain a major obstacle towards real-time implementation of MPC for high levels of automation. In particular, Level 4 and Level 5 of automation defined by the Society of Automated Engineers [5] must be able to handle hazardous scenarios without any intervention from the human driver. Clearly, in such critical situations, sub-optimal but computationally rapid responses are more valuable than optimal responses obtained after long computations. Thus, improving the computational efficiency of MPC in critical scenarios remains a crucial research challenge.

Several lines of research have been investigated to deal with this challenge: suggested approaches to increase computational efficiency include decoupling the lateral and longitudinal vehicle dynamics [113] or using ad-hoc kinematics and dynamics [8]. Partly-related research lines have looked at how model fidelity affects the control performance during critical maneuvers in limits of friction [114] or around drift equilibria [115].

Another line of research has been studying computationally more efficient solutions to the nonlinear optimization problem e.g., via new numerical algorithms [84] or offline explicit solutions [116]. Nevertheless, Tavernini et al. [117] demonstrated that the offline explicit MPC approach does not yield significant computational improvements. Adaptive weights, adaptive prediction horizon [118] or adaptive sampling times [119] have also been examined, which can sometimes improve computational efficiency although Wurts et al. [10] argue that varying sampling times can increase the computational burden due to the resulting change in integration points in the prediction horizon.

Switching-based control designs are another line of research for computational efficiency of MPC, for instance, by switching among different prediction models [120]. Nevertheless, there is often no systematic way to define a good switching strategy, as the switching can be defined in different ways such as switching to a higher-fidelity model in case of uncontrollable error divergence [121], or switching among different drifting/driving modes [122]. In this sense, a more systematic framework that covers switching-based design as a special case is hybridization [81]. Hybridization refer to approximating the control optimization problem using a hybrid systems formulation incorporating both continuous and discrete dynamics [18]. Hybridization is equivalent to breaking down a nonlinear possibly complex form into multiple modes with lower complexity, each mode being valid in a local activation region. By this approach, nonlinearity is traded with the introduction of discrete dynamics, representing the switching among the different modes of the system [23].

Hybridization has been used to improve the computational speed in various applications [33, 35, 41]. In the automated driving literature, different approaches to hybridize the vehicle dynamics include representing the nonlinear tire forces by a piecewise-affine function [36, 83, 84], using a grid-based linear-parameter-varying approximation [123], or using a hybrid equivalent state machine [124]. Nevertheless, to the best of our knowledge, hybridization has not yet been incorporated into emergency evasive maneuvers and/or highly-nonlinear vehicle dynamics. For example, the hybridization in [21] via a Mixed-Logical-Dynamical (MLD) formalism [26] is only valid at low-speeds where vehicle

nonlinearity can be neglected and the coupling between lateral and longitudinal vehicle dynamics is weaker.

Indeed, in addition to the nonlinear vehicle model that enters the MPC problem as equality constraint, another crucial source of nonlinearity in the control optimization problem is caused by the physics-based inequality constraints such as handling and tire force limits that are generally non-convex. The hybridization problem in MPC must necessarily involve both model and constraint approximations, an aspect that is often neglected in the literature. Despite some similarities, there are clear distinctions in the two resulting hybridization problems that must be taken into account.

Among different hybrid modeling frameworks, MMPS systems [24] do not require to explicitly represent the activation regions, which simplifies the approximation by significant reduction of the number of decision variables. For this reason, the MMPS approach is the one adopted in this work. As its name suggests, MMPS formulation represents a function using only (and possibly nested) max, min, adding and scaling operators. Kripfganz [91] showed that any MMPS function can also be equivalently represented by the difference of two convex MMPS functions, which can increase computational tractability.

Physics-based non-convex constraints have been dealt with in different ways. For instance, [125] considers the convex hull of the non-convex polyhedral constraints and disregards non-optimal solutions using the binary search tree of [126]. In reachability analysis, [127] computes an inner-approximation of the feasible region using an outer approximation of the reachable sets.

Lossless or successive convexification is a common approach to deal with non-convex constraints, as often considered in real-time trajectory planning [128–130]. However, the real-time capability of the convexification method is a crucial and non-trivial aspect, since the non-convex constraints imposed by the environment are changing at each control time step.

Convexification problem can be solved offline only when the constraints are known to be fixed. In some applications such as path planning in cluttered environments, it is important to find a feasible region for the next control time step, which translates into finding the largest convex subset of a given cluttered feasible region [131]. Nevertheless, a generic offline convexification problem can be obtained by approximating a non-convex region by a union of convex subregions. As defining these subregions manually is unpractical [132], approaches from computational geometry have been proposed. For instance, it has been shown that convexification is analogous to the NP-hard problem of Approximate Convex Decomposition [133] with applications to shape analysis [134] or decision region in pattern recognition [135]. Indeed, the recent advances in this field have been tailored more and more toward the specific needs of pattern recognition. For example, more emphasis is given on shape analysis by concavity matrices [136]: however, in critical automated driving scenarios, it is rather important to analyze the approximations inaccuracy with respect to the distance to the non-convex boundary. Existing methods in this sense are mainly tailored for non-convex polyhedral regions [137], but several physics-based constraints arising during critical maneuvers are not polyhedral.

In practice, hybridization has rarely been considered for highly complex vehicle models; e.g., to the best of our knowledge, there are no studies that include hybridization of the coupled longitudinal and lateral vehicle dynamics. Moreover, controlling evasive maneuvers

in critical scenarios requires a systematic analysis of the vehicle model complexity and the resulting computation trade-off, which has not been conducted as far as we are aware.

In this chapter, we provide a comparison benchmark to analyze and improve the computational performance of MPC optimization problem for vehicle control in critical high-velocity scenarios using hybrid formulation of the control optimization problem. This benchmark is divided in two parts: the first part is dedicated to the hybridization of the MPC via approximating the constraints, i.e., prediction model and physics-based constraints, whereas the second part investigates the improvements of the resulting hybrid MPC controller in comparison with the original nonlinear MPC controller.

The current chapter contributes to the state of the art by:

- presenting of a novel hybrid approximation of the system using an MMPS formulation,
- developing a new generalized formalism for constraint approximation problem including an approach based on a polytopic definition of the regions by an MMPS function, and comparing the resulting approximations with two methods from the literature,
- introducing two trajectory-based grid generation method for model approximation,
- investigating grid-based numerical solutions of the model and constraint approximation with respect to the grid behavior, and
- presenting a novel benchmark for evaluating and comparing the computational efficiency of various nonlinear MPC controllers.

The chapter is organized as follows: Section 6.2 covers the preliminary definitions of the model and constraint approximation problems. Section 6.3 describes the grid generation methods, including the novel trajectory-based approach in non-uniform sampling of the input/state pairs. Section 6.4 defines the approximation problems. Section 6.5 presents the hybridization framework for model and constraint approximation using the generated grids and the validation results of the said approximation problems. Section 6.6 summarizes the hybridization framework, findings, and outlook for implementation and future work. The application and analysis of the presented hybridization framework is discussed in detail in the next chapter.

## 6.2 BACKGROUND

Consider a given nonlinear system, either in continuous-time  $\dot{x} = F(x, u)$  or in discrete-time  $x^+ = F(x, u)$  where  $x \in \mathbb{R}^n$  and  $u \in \mathbb{R}^m$  respectively represent the state and input vectors, and the domain of  $F$  is denoted by  $(x, u) \in \mathcal{D} \subseteq \mathbb{R}^{m+n}$ . In many physics-based applications, the model  $F$  is valid over a region  $\mathcal{C} \subseteq \mathcal{D}$  defined by

$$\mathcal{C} := \{(x, u) \in \mathcal{D} \mid 0 \leq G(x, u) \leq 1\},$$

which collects a set of physics-based constraints. For instance, most typical vehicle models in the literature are no longer valid if e.g., the vehicle is rolling over. Here we aim at



approximating both the nonlinear model  $F$  and the nonlinear, non-convex set  $C$ . Therefore, we need to hybridize both  $F$  and  $C$ . Both approximation problems can essentially be expressed as the minimization of the approximation error over their respective domains. The approximation error, as well as the domain, are different for each problem, as discussed hereafter.

**Remark 6.1.** *We use the normalized constraint formulation  $0 \leq G \leq 1$  instead of the generic form  $G \leq 0$  to avoid numerical issues in solving the approximation/control optimization problems.*

### 6.2.1 MODEL APPROXIMATION

The system  $F$  is approximated by a hybrid formulation  $f$  via solving the nonlinear optimization problem

$$\min_{\mathcal{A}} \int_C \frac{\|F(x, u) - f(x, u)\|_2}{\|F(x, u)\|_2 + \epsilon_0} d(x, u), \quad (6.1)$$

where  $\mathcal{A}$  represents the decision variables used to define  $f$ . The positive value  $\epsilon_0 > 0$  added to the denominator is to avoid division by very small values for  $\|F(x, u)\|_2 \approx 0$ . Note that the domain in the model approximation problem is  $C$ .

### 6.2.2 CONSTRAINT APPROXIMATION

With the nonlinear, non-convex constraints given as  $0 \leq G(x, u) \leq 1$ , we approximate the feasible region  $C$  by a union of convex subregions  $\mathcal{R}$ .

This approximation problem can be formulated in two ways: region-based and boundary-based. In the region-based approach, we minimize the misclassification error via solving the following optimization problem

$$\min_v \gamma_c \frac{\mathcal{V}\{C \setminus \mathcal{R}\}}{\mathcal{V}\{C\}} + (1 - \gamma_c) \frac{\mathcal{V}\{\mathcal{R} \setminus C\}}{\mathcal{V}\{D \setminus C\}}, \quad (6.2)$$

where  $v$  represents the decision variables used to define  $\mathcal{R}$ , the operator  $\mathcal{V}$  gives the size or “volume” of the region, and  $\gamma_c \in [0, 1]$  is a tuning parameter to adjust the relative penalization weight for the misclassification errors regarding inclusion error  $C \setminus \mathcal{R}$ , i.e., failing to cover the feasible region, and the violation error  $\mathcal{R} \setminus C$  which corresponds to violating the constraints.

In the boundary-based approach, we approximate the boundary  $G$  by a hybrid function  $g$  and minimize the boundary-approximation error similar to (6.1) via solving the optimization problem

$$\min_v \int_D \frac{|G(x, u) - g(x, u)|}{|G(x, u)| + \epsilon_0} d(x, u). \quad (6.3)$$

with  $\epsilon_0 > 0$ . Note that as  $G$  is a scalar function, the 2-norm is replaced by the absolute value here.

**Remark 6.2.** *The proposed ideas also apply in case of more inequalities e.g.,*

$$0 \leq G_i(x, u) \leq 1, \quad \text{for } i \in \{1, 2, \dots, N\},$$

*by simply formulating  $G(x, u)$  as*

$$G(x, u) = \max_{i \in \{1, 2, \dots, N\}} \{G_i(x, u)\}.$$

*Another possibility is to approximate each  $G_i$  independently; however, this may lead to redundant approximations of boundaries or parts of  $G_i$  that do not belong to the overall boundary feasible region.*

### 6.2.3 RELATION TO THE STATE-OF-THE-ART

The nonlinear non-convex constraints arise from the physics-based limitations of the system. Therefore,

- the physics-based nature of the constraints results in a connected feasible region,
- the highly-nonlinear (boundary of the) constraints limits the analytical investigation of “attainability” or optimality,
- the approximation approach is intended to be used within a hybridization benchmark, which means the method should be applicable for systems of higher degree and/or with high-dimensional feasible regions,
- the constraint violation is evaluated by ensuring that the solution lies within any of the subregions, which means overlapping subregions are acceptable,
- in light of improving the computational efficiency, it is desired to have a minimal approximation of the constraints, i.e., approximating the non-convex feasible region with a union of fewer number of subregions is desired as well as an accurate coverage of the whole region, which leads to the need for
- a systematic approach to cover the non-convex feasible region by a union of convex subregions that allows balancing the violation vs. coverage of the approximation close to the constraint boundaries.

Considering the aforementioned features, the applicability of state-of-the-art methods based on convex-hull generation [138] is limited for the current case as input-state spaces for complex vehicle models exceed four dimensions and a systematic division of the feasible region is not computationally efficient in terms of memory usage and speed for our desired accuracy. To compare our constraint approximation approach, we consider two state-of-the-art methods that share the most common elements with the aforementioned considerations in their respective problems.

The first method is from [133], where a non-convex region is covered by a number of ellipsoids. There, an optimization problem is solved to minimize the misclassification error due to the region approximation where the center and radii of the ellipsoids are the

Attainability of a point means that there exists an input such that the point is obtained by the system dynamics.

decision variables. We refer to this approach as non-parametric elliptical learning, which is equivalent to region-based approximation of the constraints by a union of ellipsoids. Our constraint approximation framework can be seen an extension and generalization of this approach by investigating boundary-based vs. region-based approximations and polytopic vs. ellipsoidal definition of the subregions.

The second method is from [139], where the gripping limits of the vehicle are approximated by a convex intersection of second-order cone constraints. There, the constraints are formulated using the system dynamics and the parameters of the combined formulation are fitted using experimental data. We refer to this approach as the convex envelope method, which is equivalent to a boundary-based approximation of the constraints by the intersection of multiple convex subregion. Since this method approximates the non-convex feasible region by a convex one, in Section 6.5 we will show its limitation in converging to an accurate approximation of the constraints in comparison with our proposed framework.

Since analytical closed-form solutions for (6.1)–(6.3) do not exist, we propose solving them numerically by generating a grid of samples from their regarding domains  $\mathcal{C}$  and  $\mathcal{D}$ , respectively denoted by  $\mathcal{C}^*$  and  $\mathcal{D}^*$ . As the grid generation method influences the quality of the final fit, we provide various grid-generation methods for both approximation problems in the next section and examine the resulting fits in our results in Section 6.5.

## 6.3 GRID GENERATION

We use two main approaches to generate  $\mathcal{D}^*$ : domain-based and trajectory-based. In the domain-based approach, both the input and state elements of the grid points are selected from the input/state domain  $\mathcal{D}$ , regardless of the system's behavior. While a domain-based grid can have a good coverage of  $\mathcal{D}$ , it does not take into account the “likelihood” of the points being visited in a simulation with respect to the system dynamics. The trajectory-based way of generating  $\mathcal{D}^*$  tackles this issue by selecting the input elements of the grid points  $u^*$  from  $\mathcal{D}$ , while assigning the state elements to the points from an  $n_{\text{step}}$ -step-ahead simulation of  $F$  given  $u^*$  as the input. As a result, the obtained  $\mathcal{D}^*$  will have a higher density in regions of  $\mathcal{D}$  where the input/state pairs have a higher likelihood of being attainable.

Each of these two approaches can be implemented in two ways, giving rise to a total of four methods to generate  $\mathcal{D}^*$ :

- **Domain-based:** *[points are directly sampled from  $\mathcal{D}$ ]*
  - **Uniform ( $\mathcal{D}_{\text{U}}^*$ , also referred to as U grid type):** the points are generated by picking  $n_{\text{samp}}$  uniformly-spaced points along each axis in  $\mathcal{D}$ .
  - **Random ( $\mathcal{D}_{\text{R}}^*$ , also referred to as R grid type):** a total of  $n_{\text{rand}}$  points are randomly selected from  $\mathcal{D}$ .
- **Trajectory-based:** *[ $n_{\text{sim}}$  open-loop simulations with  $n_{\text{step}}$  steps of  $F$  are run using random inputs from  $\mathcal{D}$ ]*

---

For instance, another approach to solving the aforementioned approximation problem is the Monte Carlo integration method.

- **Steady-state** ( $D_S^*$ , also referred to as **S grid type**): the initial state of each simulation is selected as the steady-state solution w.r.t. the initial input, i.e., it is assumed that each simulation starts from a steady state.
- **Randomly-initiated** ( $D_T^*$ , also referred to as **T grid type**): the initial state of each simulation is randomly selected from  $D$ .

Algorithms 5 and 6 respectively explain the domain-based and trajectory-based grid generation methods. The total number of grid points for each type denoted by  $\mathcal{N}$  is

$$\begin{aligned}\mathcal{N}(D_U^*) &= (n_{\text{samp}})^{m+n}, \\ \mathcal{N}(D_R^*) &= n_{\text{rand}}, \\ \mathcal{N}(D_S^*) &= \mathcal{N}(D_T^*) = n_{\text{sim}} \cdot n_{\text{step}}.\end{aligned}$$

## 6

---

### Algorithm 5 Domain-based grid generation

---

**Require:**  $F$ ,  $D$ ,  $n_{\text{samp}}$ ,  $n_{\text{rand}}$ ,  $\text{type} \in \{\text{'U'}, 'R'}\}$

$D_{\text{type}}^* \leftarrow \{\}$

**if**  $\text{type} = \text{'U'}$  **then**

**for**  $k \in \{1, 2, \dots, m+n\}$  **do**

$I_k \leftarrow \{\}$

▷  $I_k := \text{sample set}$

**for**  $i \in \{0, \frac{1}{n_{\text{samp}} - 1}, \dots, 1\}$  **do**

$I_k \leftarrow I_k \cup \{D_{(k)_{\min}} + (D_{(k)_{\max}} - D_{(k)_{\min}}) \cdot i\}$

**end for**

**end for**

$D_U^* \leftarrow I_1 \times I_2 \times \dots \times I_{m+n}$

▷ Cartesian product

**else if**  $\text{type} = \text{'R'}$  **then**

**for**  $k \in \{1, 2, \dots, n_{\text{rand}}\}$  **do**

$(x_k, u_k) \xleftarrow{\text{random}} D$

$D_R^* \leftarrow D_R^* \cup \{(x_k, u_k)\}$

**end for**

**end if**

**return**  $D_{\text{type}}^*$

---

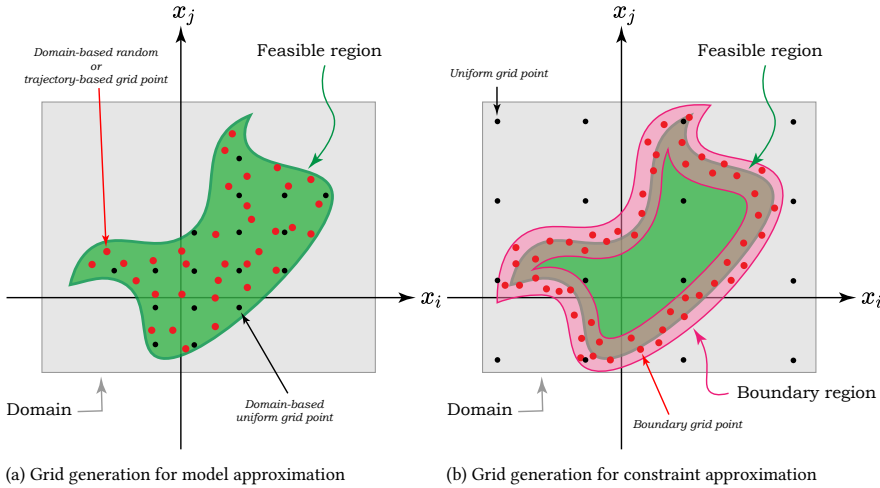
**Algorithm 6** Trajectory-based grid generation**Require:**  $F$ ,  $D_x$ ,  $D_u$ ,  $n_{\text{sim}}$ ,  $n_{\text{step}}$ ,  $\text{type} \in \{\text{'S'}, \text{'T'}\}$  $D_{\text{type}}^* \leftarrow \{\}$ **for**  $s \in \{1, 2, \dots, n_{\text{sim}}\}$  **do** $u \xleftarrow{\text{random}} D_u$  $\triangleright D_u := \text{input domain}$  $x \xleftarrow{\text{random}} D_x$  $\triangleright D_x := \text{state domain}$ **if**  $\text{type} = \text{'S'}$  **then** $x_1 \xleftarrow{\text{solve for } x} F(x, u_1) = 0$  $\triangleright \text{steady-state solution}$ **end if****for**  $k \in \{2, 3, \dots, n_{\text{step}}\}$  **do** $x_k \leftarrow x_{k-1} + F(x_{k-1}, u_{k-1})$ **if**  $(x_k, u_k) \notin D_x \times D_u$  **then****break** $\triangleright \text{stop current simulation}$ **end if** $D_{\text{type}}^* \leftarrow D_{\text{type}}^* \cup \{(x_k, u_k)\}$ **end for****end for****return**  $D_{\text{type}}^*$ 

Figure 6.1: A schematic view of different implementations of the proposed grid-generation approaches for model and constraint approximation.

The grid  $D^*$  plays the role of domain in the approximation problem. Therefore, it should be tailored to the objective of the problem itself. In this sense, Figure 6.1 shows a schematic view of the implementation of the proposed grid-generation approaches for both model and constraint approximation problems.

For model approximation, the grid should be generated only from  $\mathcal{C}$ , as the points outside  $\mathcal{C}$  are infeasible, which translates to zero likelihood of attainability. Therefore, while Algorithms 5 and 6 are implemented on  $\mathcal{D}$ , only the samples from the feasible region should be kept. Then, the four resulting grids,  $\mathcal{C}_U^*$ ,  $\mathcal{C}_R^*$ ,  $\mathcal{C}_S^*$ , and  $\mathcal{C}_T^*$  can be used to examine their efficacy.

Contrary to the model approximation problem, the points for constraint approximation should be distributed in the whole domain  $\mathcal{D}$  to allow examining the approximation error. In addition, for constraint approximation, the areas close to the boundary of  $\mathcal{C}$  are of more interest than the areas with higher likelihood of attainability. Therefore, while trajectory-based methods are useful for model approximation, to find the constraints, we are interested in using a domain-based grid with a higher density in the neighborhood of  $G(x, u) = 0$ . This grid can be obtained by combining a uniform grid  $\mathcal{D}_U^*$  with a random grid  $\mathcal{B}_R^*$  on the boundary region  $\mathcal{B}$  where

$$\mathcal{B} := \{(x, u) \in \mathcal{D} \mid |G(x, u)| \leq \epsilon_b\}.$$

The resulting generated grid is  $\mathcal{D}_U^* \cup \mathcal{B}_R^*$ .

**Remark 6.3.** To ensure that trajectory-based grids are generated by “realistic” inputs, we impose a bound constraint on the random inputs as

$$|u^*(k+1) - u^*(k)| < \Delta_u^*.$$

This can also account for the physical limitations of the actuators and be considered to be part of the physics-based constraints  $\mathcal{C}$  and it is best selected based on data from real operation of the system.

**Remark 6.4.** Depending on the problem characteristics such as the system dynamics, domain, and the nature of the input/state signals, some points in the generated grids (except for the  $U$  grid type) can be very close to each other. To avoid these points from having larger importance than other points during approximation, Algorithms 5 and 6 can further be refined by keeping only one point from each set of points that are closer to each other than a user-defined distance threshold.

## 6.4 APPROXIMATION PROBLEM FORMULATION

### 6.4.1 MODEL APPROXIMATION

We approximate the nonlinear system  $F$  by the MMPS function  $f$  with the Kripfganz form [91] as

$$f(x, u) = \max_{p \in \{1, 2, \dots, P^+\}} \{\phi_p^+(x, u)\} - \max_{q \in \{1, 2, \dots, P^-\}} \{\phi_q^-(x, u)\}, \quad (6.4)$$

where  $P^+$  and  $P^-$  are user-selected integers, and  $\phi_p^+$ , and  $\phi_q^-$  are affine functions of  $x$  and  $u$ , sometimes referred to as dynamic modes, and expressed as

$$\phi_p^+(x, u) = A_p^+ x + B_p^+ u + H_p^+,$$

$$\phi_q^-(x, u) = A_q^- x + B_q^- u + H_q^-.$$

We implement the MMPS approximation in the following fashion: each dimension of the nonlinear function, i.e., each component of  $F$ , is approximated independently. Thus,  $P^+$  and  $P^-$ , as well as the affine functions  $\phi^+$  and  $\phi^-$  are separately found for each component of  $F$ . Therefore, for brevity and without loss of generality, one can assume  $F$  to be scalar in the remaining of this section.

For a fixed pair  $(P^+, P^-)$  that corresponds to the number of affine terms in the first and second max operators in (6.4), we solve the nonlinear optimization problem (6.1) subject to (6.4) to find the optimal  $\phi^+$  and  $\phi^-$  functions where

$$\mathcal{A} = \left\{ A_p^+, A_q^-, B_p^+, B_q^-, H_p^+, H_q^- \right\}_{p \in \{1, 2, \dots, P^+\}, q \in \{1, 2, \dots, P^-\}}. \quad (6.5)$$

**Remark 6.5.** To solve the nonlinear optimization problem in (6.1), we generate a grid  $C^*$  of feasible samples from  $\mathcal{D}$  as expressed in Section 6.3, and minimize the objective function across  $C^*$ .

**Remark 6.6.** The Kripfganz form essentially expresses the function using  $P^+ \cdot P^-$  hyperplanes as there are  $P^+$  and  $P^-$  affine functions in each max operator. Therefore, the hinging hyperplanes representing the local dynamics are obtained by subtraction of the affine functions  $\phi^-$  from  $\phi^+$  which means that the optimal  $\mathcal{A}$  in (6.1) would not be unique.

Considering Remarks 6.5 and 6.6 and to avoid numerical problems, it is convenient to add a regularization term to (6.1) by penalizing the 1-norm of the decision vector as

$$\min_{\mathcal{A}} \int_{C^*} \frac{|F(x, u) - f(x, u)|}{|F(x, u)| + \epsilon_0} d(x, u) + \gamma_m \|\mathcal{A}\|_1, \quad \text{s.t. (6.4),} \quad (6.6)$$

where  $\gamma_m \in \mathbb{R}^+$  serves as a weighting coefficient to balance the penalization of the 1-norm of  $\mathcal{A}$  with respect to the approximation error.

## 6.4.2 CONSTRAINT APPROXIMATION

We approximate the feasible region  $C$  by either a union of convex polytopes using the MMPS formalism, or by a union of ellipsoids. Figure 6.2 depicts both approaches to constraint approximation.

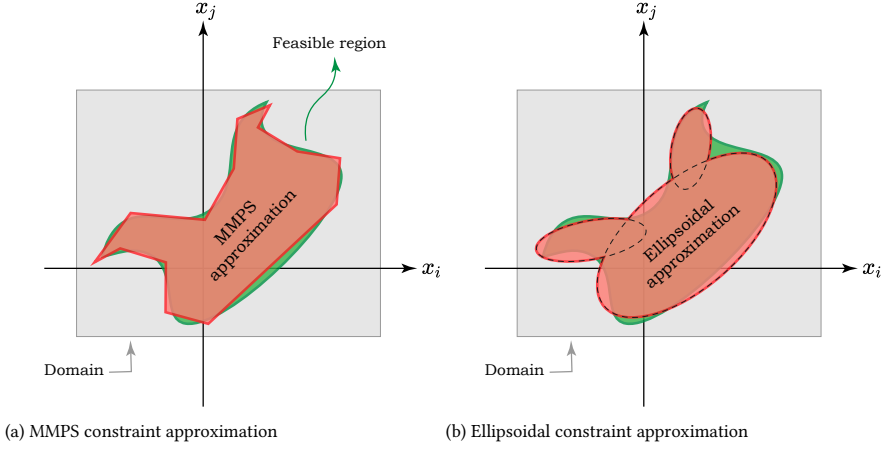


Figure 6.2: Illustration of MMPS and ellipsoidal approximation of the nonlinear constraints.

In the MMPS approach, a similar formulation to the MMPS model approximation problem is used: we approximate  $G$  by an MMPS function  $g_{\text{MMPS}}$  of the Kripfganz form in (6.4) with

$$\begin{aligned}\phi_p^+(x, u) &= C_p^+ x + D_p^+ u + I_p^+, \\ \phi_q^-(x, u) &= C_q^- x + D_q^- u + I_q^-.\end{aligned}$$

The resulting feasible region  $\mathcal{R}_{\text{MMPS}}$  is then expressed as

$$\mathcal{R}_{\text{MMPS}} := \{(x, u) \in \mathcal{D} \mid g_{\text{MMPS}}(x, u) \leq 0\}, \quad (6.7)$$

The MMPS approximation of the feasible region is then obtained via solving either the region-based (6.2) or the boundary-based (6.3) optimization problems subject to

$$\mathcal{R} = \mathcal{R}_{\text{MMPS}},$$

and

$$v = \{C_p^+, C_q^-, D_p^+, D_q^-, I_p^+, I_q^-\}_{p \in \{1, 2, \dots, P^+\}, q \in \{1, 2, \dots, P^-\}}, \quad (6.8)$$

where the matrices  $C$ ,  $D$ , and  $I$  represent the constraint-approximation counterparts of matrices  $A$ ,  $B$ , and  $H$  in (6.5) and  $(P^+, P^-)$  stand for the respective number of affine terms.

The second way is to approximate the feasible region by a union of  $n_e$  ellipsoids

$$\mathcal{R}_e := \left\{ (x, u) \in \mathcal{D} \mid \begin{pmatrix} x - x_{0_e} \\ u - u_{0_e} \end{pmatrix}^T Q_e \begin{pmatrix} x - x_{0_e} \\ u - u_{0_e} \end{pmatrix} \leq 1 \right\}, \quad (6.9)$$

with  $Q_e$  being a positive definite matrix and  $(x_0, u_0)$  representing the center coordinates of the ellipsoid. Note that this notation includes rotated ellipsoids as well. The approximated region  $\mathcal{R}_{\text{ELLIP}}$  is

$$\mathcal{R}_{\text{ELLIP}} = \bigcup_{e=1}^{n_e} \mathcal{R}_e := \{(x, u) \in \mathcal{D} \mid g_{\text{ELLIP}}(x, u) \leq 0\}, \quad (6.10)$$



whose boundary can be expressed by

$$g_{\text{ELLP}}(x, u) = \min_{e \in \{1, 2, \dots, n_e\}} \left\{ \begin{pmatrix} x - x_{0_e} \\ u - u_{0_e} \end{pmatrix}^T Q_e \begin{pmatrix} x - x_{0_e} \\ u - u_{0_e} \end{pmatrix} - 1 \right\}. \quad (6.11)$$

The ellipsoidal approximation is found by solving either the region-based (6.2) or the boundary-based (6.3) optimization problems subject to

$$\mathcal{R} = \mathcal{R}_{\text{ELLP}},$$

and

$$v = \{(x_{0_e}, u_{0_e}), Q_e\}_{e \in \{1, 2, \dots, n_e\}}. \quad (6.12)$$

## 6.5 MODEL AND CONSTRAINT HYBRIDIZATION FOR VEHICLE CONTROL

In this section, the hybridization framework consisting of the model and constraint approximation approaches is implemented on a nonlinear single-track vehicle model with Dugoff tire forces and varying friction. First, the nonlinear system and physics-based constraints are described, then the training and validation grids are defined, which are next used for model and constraint approximation problems within the hybridization framework. The results are then discussed to evaluate the performance of the different approaches and analyzed for application in other nonlinear problems.

6

### 6.5.1 NONLINEAR SYSTEM DESCRIPTIONS

A single-track representation of the vehicle is shown in Fig. 6.3. With the system variables and parameters respectively defined in Tables 6.1 and 6.2, the nonlinear vehicle model is described by the following equations [8]:

$$\dot{v}_x = \frac{1}{m} [F_{xf} \cos \delta - F_{yf} \sin \delta + F_{xr}] + v_y r, \quad (6.13)$$

$$\dot{v}_y = \frac{1}{m} [F_{xf} \sin \delta + F_{yf} \cos \delta + F_{yr}] - v_x r, \quad (6.14)$$

$$\dot{r} = \frac{1}{I_{zz}} [F_{xf} \sin \delta l_f + F_{yf} \cos \delta l_f - F_{yr} l_r], \quad (6.15)$$

and the lateral forces are given by the Dugoff model

$$F_{ya} = \frac{C_{\alpha_a}}{1 - \kappa_a} f_{\lambda}(\lambda_a^w) \alpha_a,$$

with  $a \in \{f, r\}$  where  $\mu_a$  is the varying friction coefficient, and  $\lambda_a^w$  and  $f_{\lambda}$  are the weighting coefficient and function, defined as

$$\mu_a = \mu_0 \left( 1 - e_r v_x \sqrt{\kappa_a^2 + \tan^2 \alpha_a} \right),$$

$$\lambda_a^w = \frac{\mu_a F_{za} (1 - \kappa_a)}{2 \sqrt{(C_{\kappa_a} \kappa_a)^2 + (C_{\alpha_a} \tan \alpha_a)^2}},$$

$$f_\lambda(\lambda_a^w) = \begin{cases} \lambda_a^w(2 - \lambda_a^w) & \lambda_a^w < 1 \\ 1 & \lambda_a^w \geq 1 \end{cases}.$$

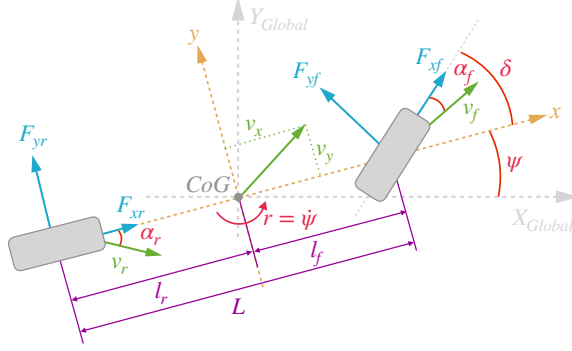


Figure 6.3: Configuration of the single-track vehicle model.

Table 6.1: System variables

Variable	Definition	Unit	Bounds
$v_x$	Longitudinal velocity	m/s	[5, 50]
$v_y$	Lateral velocity	m/s	[-10, 10]
$\psi$	Yaw angle	rad	–
$r$	Yaw rate	rad/s	[-0.6, 0.6]
$\delta$	Steering angle (road)	rad	[-0.5, 0.5]
$F_{xf}$	Longitudinal force on the front axis	N	[-5000, 0]
$F_{xr}$	Longitudinal force on the rear axis	N	[-5000, 5000]
$F_{yf}$	Lateral force on the front axis	N	–
$F_{yr}$	Lateral force on the rear axis	N	–
$F_{zf}$	Normal load on the front axis	N	–
$F_{zr}$	Normal load on the rear axis	N	–
$\alpha_f$	Front slip angle	rad	–
$\alpha_r$	Rear slip angle	rad	–
$\kappa_f$	Front slip ratio	–	–
$\kappa_r$	Rear slip ratio	–	–
$\mu_f$	Friction coefficient on the front tire	–	–
$\mu_r$	Friction coefficient on the rear tire	–	–
$x$	State vector $:= [v_x \ v_y \ r]^T$	–	–
$u$	Input vector $:= [F_{xf} \ F_{xr} \ \delta]^T$	–	–

Table 6.1 also shows the bounds we impose on state and input vectors for grid generation. The feasible region is defined by two other physics-based constraints:

1. the working limits of the vehicle (known as the g-g diagram constraint [8]) should be satisfied to allow derivation of the dynamics equation in (6.13) to (6.15); this entails

$$(\dot{v}_x - v_y r)^2 + (\dot{v}_y + v_x r)^2 \leq (\min_{a \in \{f, r\}} \{\mu_a g\})^2, \quad (6.16)$$

Table 6.2: System parameters\*

Parameter	Definition	Value	Unit
$m$	Vehicle mass	1970	kg
$I_{zz}$	Inertia moment about z-axis	3498	kg/m <sup>2</sup>
$l_f$	CoG** to front axis distance	1.4778	m
$l_r$	CoG to rear axis distance	1.4102	m
$C_{\alpha_f}$	Front cornering stiffness	126784	N
$C_{\alpha_r}$	Rear cornering stiffness	213983	N
$C_{\kappa_f}$	Front longitudinal stiffness	315000	N
$C_{\kappa_r}$	Rear longitudinal stiffness	286700	N
$\mu_0$	Zero-velocity friction	1.076	–
$e_r$	Friction slope	0.01	–

\*These values correspond to the IPG CarMaker BMW vehicle model

\*\*Center of Gravity

- the tires can provide forces up to their saturation limit, known as the Kamm circle constraint [8], which means

$$F_{xa}^2 + F_{ya}^2 \leq (\mu_a F_{za})^2, \quad a \in \{f, r\}. \quad (6.17)$$

Therefore, the feasible region  $C$  can be expressed as

$$C := \{(x, u) \in D \mid (6.16), (6.17)\}.$$

### 6.5.2 GRID DEFINITION AND COVERAGE

Table 6.3 shows the grid properties for the model and constraint approximation problems. For the model, all four U, R, S, and T grid types are used for training and later validated on a finer U, R, S, T grid type, respectively, plus C grid type, i.e. a grid that combines all of them. For the constraint approximation, only one combined grid consisting of the union U and R grids is used for training and the approximations are validated on a finer and more extended combined grid.

For a visual comparison of the grid-point distribution for different types, we have plotted the coverage of the model approximation training and validation grids in the velocity domain  $(v_x, v_y)$  in Fig. 6.4. While the grids have a similar total number of points, the density of the points among different grid types varies significantly as follows:

- The domain-based grids cover  $C$  with a uniform density compared to the trajectory-based grids.
- Compared to its random counterpart, the U grid represents a sparser distribution in the velocity domain, which stems from the fact that representation of all the possible combinations of input/state pairs on lower-dimensional sub-spaces of  $C$  projects many points on the exact same location in the viewed plane.
- Between the trajectory-based grids, the T grid gives a better coverage of  $C$ . Contrarily, the S grid favors the regions of  $C$  where the states are attainable from a steady-state

solution within a bounded number of steps, which explains the high density of points in low-speed region and the loose coverage of high-speed regions with zero lateral velocity.

Table 6.3: Properties of the grid used in the approximation problems (training and validation grids)

<i>Training Grids for Model Approximation</i>				
Type	Domain	Properties	No. Points	Feasible
U	$C$	$n_{\text{samp}} = 6$	$\approx 7,000$	100%
R	$C$	$n_{\text{rand}} = 7000$	$\approx 7,000$	100%
S	$C$	$n_{\text{sim}} = 500, n_{\text{step}} = 1000$	$\approx 7,000$	100%
T	$C$	$n_{\text{sim}} = 300, n_{\text{step}} = 1000$	$\approx 7,000$	100%
<i>Validation Grids for Model Approximation</i>				
Type	Domain	Properties	No. Points	Feasible
U	$C$	$n_{\text{samp}} = 7$	$\approx 21,000$	100%
R	$C$	$n_{\text{rand}} = 21,000$	$\approx 21,000$	100%
S	$C$	$n_{\text{sim}} = 3000, n_{\text{step}} = 1000$	$\approx 21,000$	100%
T	$C$	$n_{\text{sim}} = 1200, n_{\text{step}} = 1000$	$\approx 21,000$	100%
C	$C$	combining all the above	$\approx 84,000$	100%
<i>Training Grids for Constraint Approximation</i>				
Type	Domain	Properties	No. Points	Feasible
U	$D$	$n_{\text{samp}} = 5$	$\approx 15,000$	68%
R	$B$	$n_{\text{rand}} = 15,000, \epsilon_b = 0.1$	$\approx 15,000$	41%
C	$D$	combining all the above	$\approx 30,000$	55%
<i>Validation Grids for Constraint Approximation</i>				
Type	Domain	Properties	No. Points	Feasible
U	$D$	$n_{\text{samp}} = 6$	$\approx 47,000$	68%
R	$B$	$n_{\text{rand}} = 45,000, \epsilon_b = 0.2$	$\approx 45,000$	56%
C	$D$	combining all the above	$\approx 92,000$	62%

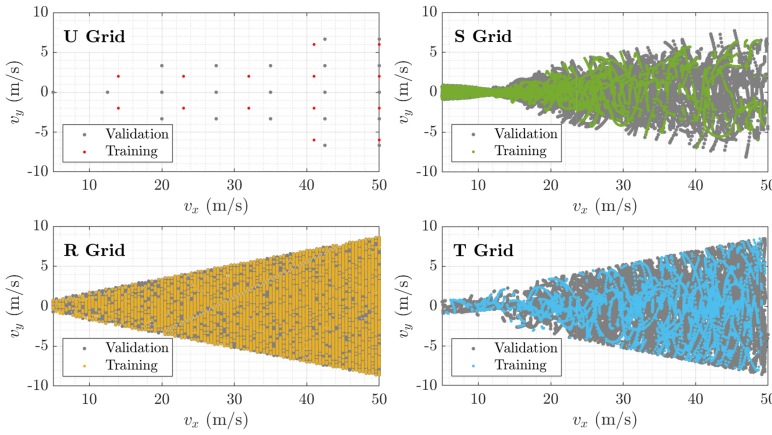


Figure 6.4: Location of training and validation grid points in the  $v_x - v_y$  domain for different grid-generation approaches in model approximation

The constraint approximation grids in the velocity domain are shown in Fig. 6.5. Besides generating more grid points in the validation grids, the width  $\epsilon_b$  of its boundary region is selected twice as large as for the training one, which increases the relative density of the grid points in the high-speed region as visible in Fig. 6.5. Moreover, both grids have 50-60% of their points in the feasible region, which is a reasonable ratio for a fair comparison.

6

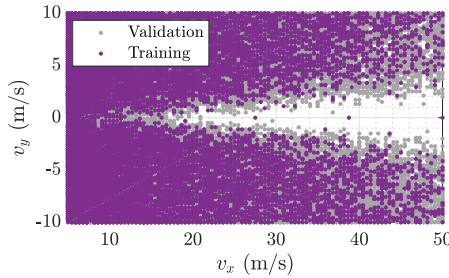


Figure 6.5: Location of the training and validation combined grid points in the  $v_x - v_y$  domain for constraint approximation

### 6.5.3 MODEL APPROXIMATION RESULTS

Using the four model training grids in Table 6.3, we approximate the dynamics of the three states independently by Kripfganz MMPS functions with  $(P^+, P^-)$  with  $P^+, P^- \in \{1, 2, \dots, 8\}$ . Since the approximated model will eventually be discretized before being incorporated in the MPC formulation, we already use a discretized form of the dynamics  $\dot{x}$  in (6.13) to (6.15) for approximation as

$$x(k+1) = \Delta x(k) + x(k).$$

Here,  $\Delta x(k)$  is approximated instead of  $x(k+1)$  for two reasons: first, the assumptions and the approximation procedure remains valid by switching from  $\dot{x}$  to  $\Delta x$ , and second,

in cases such as  $v_x$  where the state values are of a significantly larger order of magnitude compared to their rates of change, approximating  $\Delta x$  leads to a more numerically-stable representation of the error.

We solved the optimization problem (6.6) for every fixed pair of  $(P^+, P^-)$  by MATLAB's nonlinear least squares optimizer, `lsqnonlin`, using the trust-region-reflective algorithm. This optimizer further exploits the structure of the nonlinear problem by approximating the Gauss-Newton direction through minimizing the 2-norm of the function deviation in the next step. The problem is then solved for 1000 initial random guesses to provide sufficient accuracy without excessive computational effort, among which we select the lowest objective value as the optimal solution. The codes for grid generation and hybrid approximations are available from our published hybridization toolbox [87].

Fig. 6.6 shows the training validation errors of the optimal solutions for  $\Delta v_x$ ,  $\Delta v_y$ , and  $\Delta r$  on model approximation validation grids in Table 6.3. The lateral dynamics of the nonlinear model has a higher degree of nonlinearity, which explains the different error scales in the MMPS approximation. The plots are grouped based on the system and the type of the training grid to gain a better insight into the behavior of each grid and its effect on the accuracy of the approximation.

Firstly, it is observed that U and R grids overfit for lower numbers of hyperplanes compared to their trajectory-based counterparts, which is represented by high oscillations after a certain degree of complexity in the approximation form. The S grid shows the lowest oscillatory behavior in validation results, which can indicate the inability of this grid in converging to an accurate fit due to its grid-point distribution with higher density in regions that are attainable from a steady-state solution of the system dynamics.

For  $\Delta v_x$ , U and R grids show overfitting behavior for  $P^+ + P^- \geq 4$  modes and T grid overfits for  $P^+ + P^- \geq 5$ . However, the S grid does not show overfitting until 13 modes with a lower validation error ( $\approx 0.4\%$ ) compared to the other grid types ( $\approx 0.8\%$ ). It is worth noting that the trajectory-based validation grids start overfitting for a much larger number of modes compared to the domain-based types.

For  $\Delta v_y$ , U and R grids again overfit at 4 modes, with 3% and 2% validation errors, respectively. The S grid overfits at 12 to 14 modes with reaching a validation error that is slightly above 1%, and the T grid overfits at 11 modes with an error of 2%.

For  $\Delta r$ , U grid overfits at 4 modes and its validation error remains above 42%. On the other hand, the R, S, and T grids reach their best fits at 12 to 15 modes, all with an error of about 9%. The S grid, while having the lowest training error in most cases, has the highest offset between the validation and the training error. This could be due to the S grid needing more points to provide a more realistic training error. However, it should be noted that the steady-state-initiated method's ability to generate new "distinct" points is limited; as Table 6.3 shows, to generate a validation grid three-times as large as the training one, the number of simulations needed to be multiplied by 6, which is not the case for its randomly-initiated counterpart, T. As the set of points attainable by a random input signal from a steady-state solution is limited, this difference is understandable. Nevertheless, this limitation is not restricting the S grid's ability to fit the model significantly (compared to e.g., the U grid).

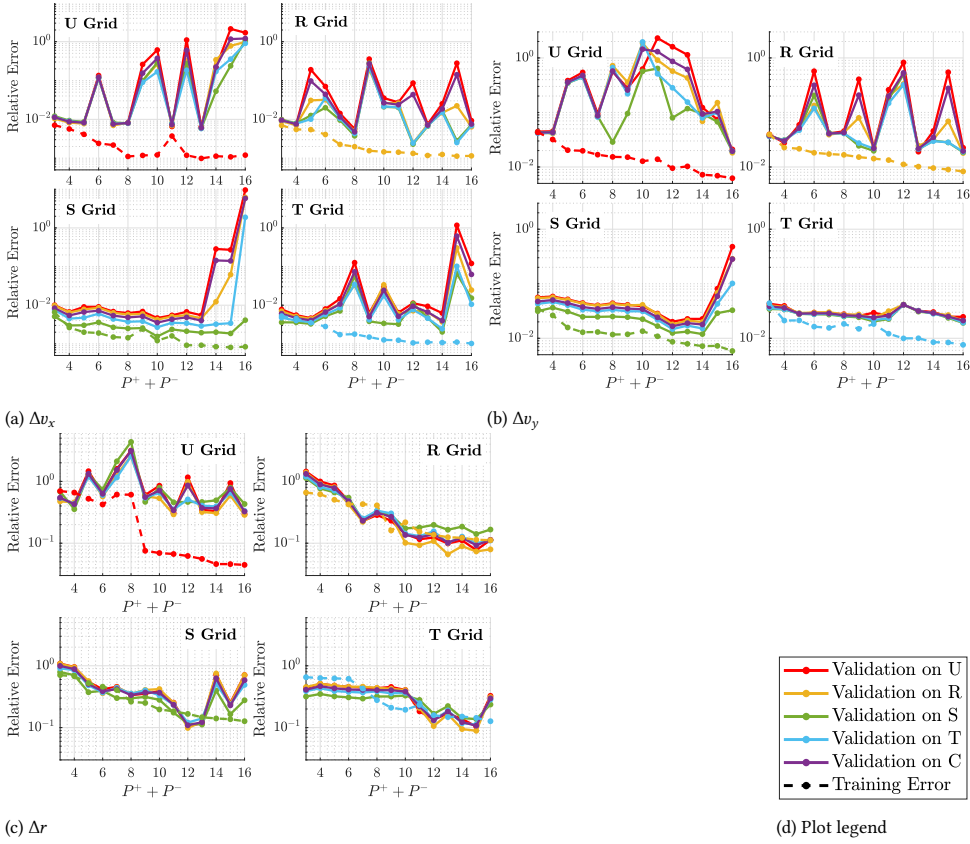


Figure 6.6: Cross-validation of the MMPS approximations for different dynamics using four grid types. Since all the plots share the same legend, it is placed separately.

### 6.5.4 CONSTRAINT APPROXIMATION AND VALIDATION

For constraint approximation, both training and validation steps are done on the two constraint approximation C grids defined in Table 6.3. The nonlinear constraints are approximated by either an intersection of second-order cones, which corresponds to the implementation of the convex envelope method from [139], or a union of convex subregions, which gives a non-convex approximation of the feasible region. Based on the formulation of the approximation problem, i.e., (6.2) or (6.3), the approach is region- or boundary-based. The shape of the subregions is also either ellipsoidal or polytopic, where the latter is developed by an MMPS formulation of the nonlinear constraint. This leads to four methods of constraint approximation as shown in Table 6.5 where the best fits and their corresponding parameters as well as their approximation errors are presented. It should be noted that the region-based ellipsoidal approximation is a modified implementation of the non-parametric ellipsoidal learning method [134].

Similar to model approximation, we solved the boundary-based optimization problems

Table 6.4: Best validation fits for different grid types

Grid type	$\Delta v_x$		$\Delta v_y$		$\Delta r$	
	$(P^+, P^-)$	Error*	$(P^+, P^-)$	Error*	$(P^+, P^-)$	Error*
<b>U</b>	(2,3)	0.8%	(2,2)	4.3%	(2,2)	42.8%
<b>R</b>	(2,2)	0.7%	(2,2)	3.0%	(7,8)	9.2%
<b>S</b>	(3,7)	0.3%	(6,8)	1.8%	(6,6)	9.8%
<b>T</b>	(2,3)	0.5%	(6,3)	2.6%	(7,8)	8.8%

\* Relative validation error on the C grid

(6.3) for every fixed pair of  $(P^+, P^-)$  or  $n_c$  by MATLAB's nonlinear least squares optimizer, `lsqnonlin` for 1000 initial guesses (selected in a similar way as for the model approximation). However, the region-based approach results in a non-smooth optimization problem (6.2) which we solved using the particle swarm optimizer in MATLAB, which does not require the problem to be differentiable. The swarm size was selected to be 10 times larger than the number of decision variables as a sufficiently large number for our experiments, and the problem was solved 1000 times for each case of  $(P^+, P^-)$  or  $n_c$  and the best solution was kept as the optimal one. In addition, the convex envelope approach from [139] where the boundary of the nonlinear constraints is approximated by an intersection of  $n_c$  second-order cone constraints is also implemented in the same fashion for different values of  $n_c$ . Figure 6.7 shows the training and validation errors for different constraint approximation methods.

The convex envelope approach approximates the feasible region by a convex area that is the intersection of  $n_c$  second-order cone constraints. Therefore, for systems where the concavity measure, i.e., the difference between the feasible region and its convex hull, is significant compared to its size, this method converges to either high violation or inclusion misclassification errors, which is visible in the behavior of the training and validation plots in Fig. 6.7a. Starting from one second-order cone constraint to approximate the feasible region with, this approach converges to an area covering about 25% of the feasible and 25% of the infeasible regions. Increasing the number of cone constraints to more than 3 leads to a significant improvement in the obtained fit. Nevertheless, the best convex envelope fit is obtained at  $n_c = 6$  with the inclusion and violation errors of 45% and 5% respectively, both of which are not acceptable as a proper fit. This shows that the method is converging to more accurate approximations of the largest convex subset of the feasible region, which is covering about 50% of it.

The difference between the region- and boundary-based approaches is due the fact that in the region-based approximation (6.2), the inclusion and violation misclassification errors are penalized, while in the boundary-based approximation (6.3), the error in approximation of the distance to the boundary is minimized. This difference is more clear in the MMPS approximation plots where with one binary variable, the boundary is approximated by an affine function, i.e., a hyperplane. Problem (6.3) then converges to a hyperplane with the lowest sum of distances from the nonlinear boundary. However, since the violation error is penalized more than the inclusion error with  $\gamma_c < 0.5$ , problem (6.2) converges



to an empty set where the violation error is zero and the inclusion error is 1, giving the optimal misclassification error of  $1 - \gamma_c$ . In all the cases, it is observed that the region-based approximation converges to lower violation and higher inclusion errors due to the same reason.

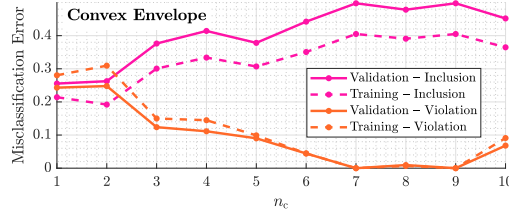
MMPS approximation of the constraints via the region-based approach shows overfitting behavior after considering 6 binary variables. After 3 binary variables, the fits start oscillating between a more “inclusive” approximation and a more “violating” one. However, the best fit is obtained with 7 binary variables. Even by increasing this number, problem (6.2) keeps converging to the same misclassification error.

Boundary-based MMPS approximation reaches the best fit with 8 binary variables where again, adding more binary variables and increasing the complexity level of the fit does not change the inclusion and violation errors significantly and only minor oscillations between converging to a slightly more inclusive approximation or to a slightly more violating one are observed.

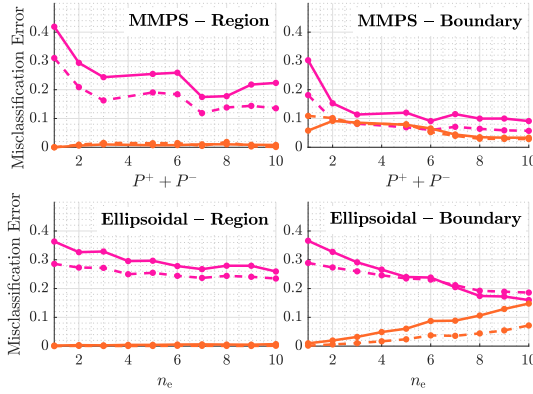
Ellipsoidal approximation of the feasible region generally converges to fits with lower accuracy compared to the MMPS approximation. In the region-based approximation, the training and validation errors stay at the same level with slight oscillations after  $n_e = 7$  with inclusion and violation misclassification errors of respectively 26.7% and 0.6%. In this sense, for the same number of integer variables, the ellipsoidal region-based approximation converges to a similar violation error but a 50% higher inclusion error. The boundary-based ellipsoidal approximation on the other hand shows a different overfitting behavior where increasing the number of ellipsoidal subregions results in convergence to a better coverage at the expense of a significant increase in violation error. Therefore, the best fit should be selected before the point where the violation error exceeds a user-defined accepted threshold. Here we select  $n_e = 5$  since it is the last complexity before the violation error exceeds 6%. Another observed pattern is the divergence of violation errors in training and validation, which mirrors the nature of the approximation approach: increasing the number of ellipsoids translates into generating more ellipsoidal subregions close to the boundary to minimize the distance-to-boundary sum. However, in the validation phase this leads to significantly higher violation errors as a result of the approximation overfitting to the training grid.

Table 6.5: Best constraint approximation fits

Subregions	Approach	Fit Parameters	Error	
			Inclusion	Violation
Intersection of convex subregions [139]				
Cone	Boundary	$n_c = 6$	45.0%	5.0%
Union of convex subregions				
MMPS	Region	$(P^+, P^-) = (5, 2)$	17.5%	0.5%
MMPS	Boundary	$(P^+, P^-) = (4, 4)$	9.9%	3.5%
Ellipsoidal	Region [134]	$n_e = 7$	26.7%	0.6%
Ellipsoidal	Boundary	$n_e = 5$	24.0%	6.0%



(a) Intersection of convex subregions



(b) Union of convex subregions

Figure 6.7: Training and validation plots for different constraint approximation problems. As the axes share the same legend, it is only presented in the first one.

## 6.6 CONCLUSIONS AND OUTLOOK

This chapter has presented a hybridization framework for approximation nonlinear model and constraints. This framework serves as benchmark for formulating nonlinear MPC optimization problems using a hybrid systems formalism to improve computational efficiency and to ensure real-time implementation. The conclusions of the research in this chapter with respect to its contributions, and the hybridization framework are summarized in the following subsections. The hybrid control comparison benchmark is discussed in detail in the next chapter.

### 6.6.1 CONCLUSIONS FOR VEHICLE CONTROL

Introduction of the hybridization framework in this chapter is a result of the following steps where the model and constraint approximation problems were defined by means of several novel descriptions of the approximation problem. First, for the model approximation, the Kripfganz MMPS form was used to approximate the nonlinear system to a user-defined error bound. Second, the nonlinear feasible region resulting from the physics-based constraints was approximated by a union of ellipsoids and polytopes via region- and boundary-based formulation of the approximation problem. Third, the model and constraint approximation

problems were solved numerically across various grids types sampled from the input/state domain and their corresponding fit qualities in terms of accuracy and overfitting behavior were compared. Fourth, among the different grid types, two novel trajectory-based grid generation methods were introduced to structurally increase the density of the grid points in regions of the state domain with higher likelihood of the attainability by the system dynamics. This approach resulted in 15-60% reduction of the approximation error compared to its domain-based counterpart. Finally, the different grid generation and formulations of the approximation problems were analyzed to present a hybridization benchmark for improving the computational performance of the MPC problem for other applications of nonlinear MPC, as well as tracking control in emergency evasive maneuvers; this comparative assessment is explained in the next chapter.

### 6.6.2 GENERALIZED HYBRIDIZATION FRAMEWORK

Our proposed hybridization framework can be implemented in other applications of nonlinear MPC to improve computational efficiency by considering the following guidelines:

1. The model approximation problem should be solved by either an R, S, or T grid. The density of the R-type grid points can vary by sampling using various random distributions. Additionally, if there is a significant variance in the likelihood of attainability for different input/state pairs, it is recommended to use the trajectory-based S or T grids. Depending on the nature of the system dynamics, the S grid is a proper choice if the attainable subset of the state-domain from steady-state solutions is rich or large enough to ensure coverage of the whole domain by selecting a sufficiently large number of sampling points over each trajectory. On the other hand, this will not be an issue for the T grid, at the expense of including input/state pairs that are only attainable from an unattainable initial state. In general, if such properties of the system dynamics are not fully known, it is suggested to consider all three grid types and compare the overfitting behavior as done in this chapter.
2. The Kripfganz MMPS form is a compact and well-formulated way to impose continuity in the hybrid approximation of the nonlinear problem; it provides straightforward and intuitive control over the accuracy of the approximation with respect to the number of introduced binary variables that are assigned to each affine local dynamics appearing in the max operators. The number of affine terms can be increased up until the point where either the maximum number of binary variables or the maximum tolerated approximation error are reached. Both of these stopping criteria can be chosen by the user and based on the application.
3. The nonlinear non-convex feasible region can be approximated by a union of ellipsoids or polytopes using region-, as well as boundary-based formulations of the approximation problem. If the application requires to strictly avoid violating the nonlinear constraints by the approximated ones, it is recommended to use the region-based formulation of the approximation problem. However, the boundary-based formulation leaves more room to balance the trade-off between covering the nonlinear region and violating it, and converges to better coverage of the non-convex region. This trade-off can also be managed within the region-based formulation by

adjusting the tuning parameter  $\gamma_c$ , but its capability in modifying the priority of the costs of inclusion vs. violation error with respect to the distance from the boundary is limited.

Using the above guidelines, the hybridization approach can be implemented in different applications such as motion planning, navigation, or real-time control of systems with fast dynamics where it is required to balance the computational speed and accuracy of the MPC problem.

### 6.6.3 NEXT STEPS AND FUTURE WORK

In the next part of this chapter, we present the hybrid control comparison benchmark using this hybridization framework for balancing the computational efficiency of the MPC optimization problem in vehicle control during emergency evasive maneuvers.

The next steps of the current research can proceed along (but not limited to) the following lines: investigation of the proposed hybridization framework in applications with higher dimensions e.g., large-scale control problems, extension of the model approximation step by incorporating other hybrid modeling frameworks such as piecewise-quadratic or mixed-logical-dynamical systems as compact models for a good trade-off between constraint satisfaction, computational complexity, and control performance.

## 7

# COMPARATIVE ASSESSMENT OF HYBRIDIZATION IN VEHICLE CONTROL

*We may call it the paradox of the decider: as the circulation of information becomes faster and more complex, the time available for the elaboration of relevant information becomes shorter. The more space taken by the available information, the less time there is for understanding and conscious choice.*

— Franco Berardi, *After the Future*

7

*Optimization-based approaches such as nonlinear Model Predictive Control (MPC) are promising approaches in safety-critical applications with nonlinear dynamics and uncertain environments such as automated driving systems. However, the computational complexity of the nonlinear MPC optimization problem coupled with the need for rapid response in emergency scenarios is the main bottleneck in realization of automation levels four and five for driving systems. In this chapter, we construct hybrid formulations of the nonlinear MPC problem for vehicle control during emergency evasive maneuvers and assess their computational efficiency in terms of accuracy and solution time. To hybridize the MPC problem, we combine three hybrid approximations of the prediction model and four approximations of the nonlinear stability and tire saturation constraints and simulate the closed-loop behavior of the resulting controllers during five emergency maneuvers for different prediction horizons. Further, we compare the robustness of the controllers and their accuracy-time trade-off when the friction of the road is either unknown or has an offset error with respect to the prediction model. This robustness is investigated for different levels of friction uncertainty and with respect to the proximity to the vehicle handling limits. Our tests show that the hybridization of the MPC*

*problem results in an efficient implementation of MPC for emergency evasive maneuvers, paving the way for implementation at high levels of automation.*

## 7.1 INTRODUCTION

Real-time implementation of nonlinear MPC for high-speed safety-critical evasive maneuvers is an open research problem [12]. Two specific reasons contribute to this: high computation times for solving a NonLinear Program (NLP) compared to a linear or a Quadratic Program (QP), and possible convergence to local optima, which is highly sensitive to the initial guess provided to the NLP solver.

Proactive vehicle control in emergency scenarios requires using the full control potential of the system, which means that some sub-optimal solution techniques for the NLP [11, 140, 141] are not suitable to incorporate [142]. To mitigate the slow convergence of NLP solvers, an upper bound is often imposed for the computation time as stopping criterion; this bound can be selected e.g., as a function of the complexity of the problem using prediction horizon, decision variable, etc. If the solver does not converge to a optimum before hitting this bound, the solution to the previous step is shifted and used [8]. Nevertheless, if this occurs repeatedly and the controller does not converge to a solution for consecutive steps, this may result in a large degree of suboptimality or even infeasibility.

A popular approach for selecting the initial guess is using a warm-start strategy based on shifting the previous solution to tailor it for the current MPC optimization problem [8, 84, 143], which is suitable provided that the previous step converged to a good solution. This however is a restrictive condition, for which [144] proposed using a tangential solution predictor instead of shifting, which is essentially based on using the concept of parametric sensitivity of the NLP for constructing new initial guesses. Nevertheless, warm start is a suitable strategy only if the solver converged to a “good” solution in the previous step [93]. Other strategies to improve the initial guess include using the reference trajectory [93], using the inverse static model of the system [100], or selecting the solution to a simpler approximation of the NLP e.g., a QP [101]. Nevertheless, the mentioned approaches are not sufficient for real-time control during emergency evasive maneuvers where a more extensive search in the decision space is required.

During emergency maneuvers, relying on one solution is restrictive: even with the improvements on the search direction and transformation, the search for the optimum would be limited within a neighborhood of the solution for the previous time-step. However, abrupt changes to the reference trajectory e.g., due to sudden appearance of an obstacle on the road, require a more extended exploration of the search space to increase the likelihood of finding an acceptable optimum. In this sense, [145] uses a divide-and-conquer strategy in searching for starting regions based on the current state and then picks the first solution that satisfies an acceptable bound on the objective. While this method improves convergence to better optima, it still does not expand the search region in case of abrupt changes in the reference. In [10], multiple filtered random initial guesses are used to solve the NLP problem and in [146], the NLP is solved offline and a dataset of “good” initial guesses to be used in real time is learned, which could be an improvement upon relying on one solution without wasting additional computational effort on initial guesses with lower improvement value. However, this approach is only applicable in case there is sufficient and reliable data to learn such guesses, which is usually not currently available for vehicle

control in emergency scenarios.

While multi-start solution of the NLP improves the chances of converging to a suitable optimum to use the full control potential of the vehicle, it significantly increases the computation time, which is the main obstacle toward real-time implementation of MPC during emergency scenarios. In this sense, hybridization of the nonlinear control optimization problem was proposed [81, 147] to balance the computational efficiency via the trade-off between accuracy and the convergence speed by using a hybrid systems formalism [18] to express the prediction model and the nonlinear constraints.

Hybrid MPC for vehicle tracking control has attracted attention as a potential solution to tackle the problem of computational efficiency [121, 124, 148–150]. Nevertheless, to the best of our knowledge, the capability of the hybridization approach in improving the computational efficiency of MPC has neither been assessed for highly-nonlinear prediction models, nor investigated during hazardous scenarios and aggressive evasive maneuvers. In this sense, such scenarios are particularly important since they require using the full control potential of the vehicle in its handling limits and the need for fast computation is critical in collision avoidance.

This work is the sequel of Chapter 6 where we proposed an approach to approximate the prediction model and nonlinear physics-based constraints using a hybrid system formalism. The contributions of the current chapter are:

1. improving the computational efficiency of MPC, by exploiting the Max-Min-Plus-Scaling (MMPS) formulation to obtain a hybrid representation of the MPC optimization problem,
2. definition of a comparison benchmark, involving highly-nonlinear vehicle models and non-convex constraints
3. evaluation of several hybridization methods in the context of MPC by assessing three hybrid approximations of the prediction model and four approximations of the nonlinear stability and tire saturation constraints, and
4. showcasing the impact and efficacy of hybridization in terms of control performance and computation speed.

In this chapter, we use the approximated prediction model and constraints to formulate and to solve the MPC problem as either a Mixed-Integer Linear Program (MILP) or a Mixed-Integer Quadratically-Constrained Program (MIQCP). We then investigate the trade-off between the accuracy and the computation speed of the resulting hybrid MPC controllers against their nonlinear counterparts. The computational performance of the hybrid and nonlinear controllers are assessed during five aggressive evasive maneuvers, representing abrupt changes in the reference trajectory due to a hazardous situation such as a sudden appearance of an obstacle on the road. Further, we investigate the tracking errors in the presence of uncertainty in the friction coefficient as an offset as well as a disturbance such as a significant decrease of friction due to the presence of water on a section of the road.

This chapter is organized as follows: the theoretical background such as the formulation of the nonlinear and hybrid MPC problems is explained in Section 7.2. To make this chapter self-contained, we recall the hybridization approach in Chapter 6 and its corresponding

notation, to which the reader is referred. Section 7.3 explains different aspects of the comparison benchmark and assessment criteria e.g., the choice of driving scenarios and the prediction horizons. The results of the simulations and the comparative assessment are discussed in Section 7.4 followed by high-fidelity simulations in IPG CarMaker in Section 7.5. Finally, Section 7.7 presents the main results and draws an outlook for future research.

## 7.2 BACKGROUND

### 7.2.1 MODEL AND CONSTRAINT HYBRIDIZATION

Consider a nonlinear discrete-time system

$$x(k+1) = F(x(k), u(k)),$$

where  $x \in \mathbb{R}^n$  and  $u \in \mathbb{R}^m$  represent the state and input vectors, respectively. We approximate each component  $F_s$  of  $F = [F_1 \ \dots \ F_n]^T$  separately by an MMPS function  $f_s$  with the Kripfganz form [91] as

$$f_s(x, u) = \max(\phi_s^+(x, u)) - \max(\phi_s^-(x, u)), \quad \forall s \in \{1, \dots, n\}, \quad (7.1)$$

where the vectors  $\phi_s^\eta : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{p^\eta}$  with  $\eta \in \{+, -\}$  are affine functions of  $x$  and  $u$ , also referred to as dynamic modes, and expressed via matrices

$$A_s^\eta \in \mathbb{R}^{p^\eta \times m}, \quad B_s^\eta \in \mathbb{R}^{p^\eta \times n}, \quad H_s^\eta \in \mathbb{R}^{p^\eta}, \quad \forall \eta \in \{+, -\}, \forall s \in \{1, \dots, n\},$$

as

$$\phi_s^\eta(x, u) = A_s^\eta x + B_s^\eta u + H_s^\eta.$$

The general form of  $f$  is then given as

$$f(x, u) = \Psi^+(x, u) - \Psi^-(x, u),$$

where  $\Psi^+$  and  $\Psi^-$  are vector-valued functions with

$$\Psi_s^\eta(x, u) = \max(\phi_s^\eta(x, u)), \quad \forall \eta \in \{+, -\}, \forall s \in \{1, \dots, n\}.$$

Note that in this notation, the max operator returns the largest component in the vector  $\phi_s^\eta$ .

**Remark 7.1.** In Chapter 6, we used a scalar representation of the MMPS formulation to approximate each component of  $F$  separately. In this chapter, we use vector-valued representation to make the formulation of the MPC optimization problem more compact. However, without loss of generality, we consider  $G$  to be a scalar function.

For bounded  $x$  and  $u$ , the physics-based constraints are in general nonlinear and non-convex and expressed via the normalized boundary function  $G$  as

$$C := \{(x, u) \in \mathbb{R}^{m+n} \mid 0 \leq G(x, u) \leq 1\},$$

where  $C$  is referred to as the feasible region. It should be noted that we normalize the constraint function to the interval  $[0, 1]$  to avoid numerical issues in the subsequent control



optimization problems. The region  $\mathcal{C}$  is approximated either by a union of ellipsoids or by using the MMPS formalism, which corresponds to approximating the  $\mathcal{C}$  by a union of convex polytopes. In the MMPS approach, a similar formulation to the MMPS model approximation problem is used: we approximate  $G$  by an MMPS function  $g_{\text{MMPS}}$  of the Kripfganz form (7.1). The resulting feasible region  $\mathcal{R}_{\text{MMPS}}$  is then expressed as

$$\mathcal{R}_{\text{MMPS}} := \{(x, u) \in \mathbb{R}^{m+n} \mid 0 \leq g_{\text{MMPS}}(x, u) \leq 1\}. \quad (7.2)$$

via the boundary function

$$g_{\text{MMPS}}(x, u) = \max(\gamma^+(x, u)) - \max(\gamma^-(x, u)), \quad (7.3)$$

where  $\gamma^\eta : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{R^\eta}$  are affine functions of  $x$  and  $u$  as

$$\gamma^\eta(x, u) = C^\eta x + D^\eta u + I^\eta,$$

and

$$C^\eta \in \mathbb{R}^{R^\eta \times m}, \quad D^\eta \in \mathbb{R}^{R^\eta \times n}, \quad I^\eta \in \mathbb{R}^{R^\eta}, \quad \forall \eta \in \{+, -\}.$$

The second way is to approximate the feasible region by a union of  $n_e$  ellipsoids as

$$\mathcal{R}_{\text{ELLIP}} := \{(x, u) \in \mathbb{R}^{m+n} \mid 0 \leq g_{\text{ELLIP}}(x, u) \leq 1\}, \quad (7.4)$$

whose boundary can be expressed by

$$g_{\text{ELLIP}}(x, u) = \min(\omega(x, u)), \quad (7.5)$$

where the min operator gives the smallest component in the vector  $\omega$ , and where

$$\omega_e(x, u) = \begin{pmatrix} x - x_{0,e} \\ u - u_{0,e} \end{pmatrix}^T Q_e \begin{pmatrix} x - x_{0,e} \\ u - u_{0,e} \end{pmatrix} - 1, \quad \forall e \in \{1, \dots, n_e\}, \quad (7.6)$$

with  $Q_e$  being a positive definite matrix and  $(x_{0,e}, u_{0,e})$  representing the center coordinates of the ellipsoid. Note that this representation includes rotated ellipsoids as well.

## 7.2.2 MPC OPTIMIZATION PROBLEMS

The state and input vectors over the whole prediction horizon  $N_p$  are defined as

$$\tilde{x}(k+1) = [\hat{x}^T(k+1|k) \quad \hat{x}^T(k+2|k) \quad \dots \quad \hat{x}^T(k+N_p|k)]^T,$$

$$\tilde{u}(k) = [u^T(k) \quad u^T(k+1) \quad \dots \quad u^T(k+N_p-1)]^T,$$

where  $\hat{x}(k+i|k)$  represents the predicted state of the  $(k+i)$ -th time step based on the state measurement at the  $k$ -th time step. For the sake of simplicity, the control horizon is assumed to be equal to the prediction horizon  $N_p$ . In addition and for brevity of expressions, we introduce the generalized form of the systems dynamics  $F$  and inequality constraints  $G$  over the prediction horizon as

$$\begin{aligned} [x(k+1) = F(x(k), u(k))] &\iff [\tilde{x}(k+1) = \tilde{F}(\tilde{x}(k), \tilde{u}(k))] \\ [0 \leq G(x(k), u(k)) \leq 1] &\iff [0 \leq \tilde{G}(\tilde{x}(k), \tilde{u}(k)) \leq 1]. \end{aligned}$$

Note that  $\tilde{F}$  is the generalized counterpart of  $F$  by extending the notation over the prediction horizon and not by recursive substitution. For the sake of brevity,  $x(k)$  is not an explicit argument of  $\tilde{F}$  but note that the dependence of  $\tilde{F}$  on  $x(k)$  is implied within the  $\tilde{x}(k)$  argument.

Using the  $\ell_1$ -norm in defining the objective function in tracking  $\tilde{x}_{\text{ref}}$ , MPC requires solving the optimization problem

$$\min_{\tilde{u}(k)} \quad \|\Theta_x(\tilde{x}(k) - \tilde{x}_{\text{ref}}(k))\|_1 + \|\Theta_u \tilde{u}(k)\|_1, \quad (7.7a)$$

$$\text{s.t.} \quad \tilde{x}(k+1) = \tilde{F}(\tilde{x}(k), \tilde{u}(k)), \quad (7.7b)$$

$$0 \leq \tilde{G}(\tilde{x}(k), \tilde{u}(k)) \leq 1, \quad (7.7c)$$

with  $\Theta_x \geq 0$  and  $\Theta_u \geq 0$  being normalizing diagonal matrices with non-negative entries for the state tracking error and input signals, respectively. Note that the  $\ell_1$ -norm is selected to allow a mixed-integer linear description of the objective function.

The hybrid MPC problem can then be formulated as:

$$\min_{\tilde{u}(k)} \quad \Theta_x \tilde{e}_x(k) + \Theta_u \tilde{e}_u(k), \quad (7.8a)$$

$$\text{s.t.} \quad -\tilde{e}_x(k) \leq \tilde{x}(k) - \tilde{x}_{\text{ref}}(k) \leq \tilde{e}_x(k), \quad (7.8b)$$

$$-\tilde{e}_u(k) \leq \tilde{u}(k) \leq \tilde{e}_u(k), \quad (7.8c)$$

$$\tilde{x}(k+1) = \text{vec}(\Psi^+(k)) - \text{vec}(\Psi^-(k)), \quad (7.8d)$$

$$\Psi_{ij}^+(k) = \max(\phi_i^+(k+j-1)), \quad \forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, N_p\}, \quad (7.8e)$$

$$\Psi_{ij}^-(k) = \max(\phi_i^-(k+j-1)), \quad \forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, N_p\}, \quad (7.8f)$$

where (7.8b)–(7.8c) are introduced to obtain a linear representation of the objective function by defining

$$\tilde{e}_x(k) = \|\tilde{x}(k) - \tilde{x}_{\text{ref}}(k)\|_1, \quad \tilde{e}_u(k) = \|\tilde{u}(k)\|_1,$$

and (7.8d)–(7.8e) are the hybridized model approximation to replace (7.7b). The  $\text{vec}(\cdot)$  operator in (7.8d) converts its matrix argument into a vector by stacking its components into one column vector. Then, constraint approximation can be hybridized by replacing (7.7c) by the MMPS constraints (7.9a) for an MILP or the ellipsoidal constraints (7.9b) for an MIQCP formulation:

$$\Lambda_j^\eta = \max(\gamma^\eta(k+j-1)), \quad \forall \eta \in \{+, -\}, \forall j \in \{1, \dots, N_p\}, \quad (7.9a)$$

$$\Omega_j(k) = \min(\omega(k+j-1)), \quad \forall j \in \{1, \dots, N_p\}. \quad (7.9b)$$

**Remark 7.2.** The binary variables of the optimization problem are introduced via activating the local modes for the hybrid model and constraints (for more details, see Chapter 6). Therefore, the corresponding MILP problem will have

$$N_p \left( R^+ + R^- + \sum_{s=1}^n (P_s^+ + P_s^-) \right)$$

binary variables with  $R^+$  and  $R^-$  being the constraint-approximation counterparts of  $P^+$  and  $P^-$ , and the MIQCP problem will have

$$N_p \left( n_e + \sum_{s=1}^n (P_s^+ + P_s^-) \right)$$

binary variables.

## 7.3 COMPARISON BENCHMARK

### 7.3.1 PREDICTION MODEL AND PHYSICS-BASED CONSTRAINTS

The nonlinear prediction model is a single-track vehicle with a Dugoff tire [151] model with varying friction as described in Section 6.4, with system variables and parameters given in Tables 6.1 and 6.2.

In Chapter 6, we hybridized the nonlinear model using different grid-generation methods via the MMPS formalism (7.1) and obtained three hybrid approximations for the nonlinear model labeled by their corresponding grid types as R, S, and T models. The nonlinear physics-based constraints due to the tire force saturation and vehicle stability were hybridized as well via approximating the feasible region by a union of ellipsoids and by a union of polytopes (using the MMPS formalism) via boundary-based and region-based approximations. There, we obtained four approximations labeled by their approach (R and B) and by the shape of the subregions (MP for MMPS or EL for ellipsoidal) as RMP, BMP, REL, and BEL. Table 7.1 summarizes the abbreviations used in this chapter for different hybrid models, constraints, and their corresponding controllers. For more details of the boundary-based and region-based approximations and their errors, or the number of introduced binary variables by each approach, the reader is referred to Chapter 6.

Table 7.1: Abbreviations for hybrid models and controllers

Hybrid Models		
Approximation Grid Type		Abbreviation
Domain-based random		<b>R</b>
Trajectory-based steady-state initiated		<b>S</b>
Trajectory-based randomly initiated		<b>T</b>
Hybrid Constraints		
	Formulation	
	Ellipsoidal (EL)	MMPS (MP)
Region-based (R)	<b>REL</b>	<b>RMP</b>
Boundary-based (B)	<b>BEL</b>	<b>BMP</b>
Hybrid MPC Controllers		
[Model abbreviation] – [Constraint abbreviation]		
Example: <b>R</b> model + <b>BMP</b> constraint → <b>R–BMP</b> controller		

### 7.3.2 CONTROL PARAMETERS

Given the application, the *control sampling time*  $t_{sc}$  is often restricted by the capabilities of the control hardware such as the maximum operation frequency. In addition, the *model sampling time*  $t_{sm}$  is either known a priori for a discrete-time system, or obtained for a continuous-time system with respect to its natural frequency and dynamic behavior. Therefore, we assume these two parameters to be fixed during all the simulations as known system parameters  $t_{sm} = 0.01\text{s}$  and  $t_{sc} = 0.05\text{s}$ .

In the path tracking MPC literature [12], the time span of the prediction often covers below 1.5s ahead, for control sampling times shorter than 0.1s. Based on our selected control sampling time  $t_{sc} = 0.05\text{s}$ , we therefore test different time spans of the prediction in the range 0.25s to 1.50s, corresponding to  $N_p \in \{5, 10, \dots, 30\}$ . This is further explained in Section 7.3.5.

### 7.3.3 MPC CONTROLLERS

In this benchmark, we consider two nonlinear MPC controllers with the nonlinear prediction model and the physics-based constraints. The first one solves the NLP using the warm start only (i.e. with the shifted solution of the previous time step) labeled as NL-1, and the second one referred to as NL-5 solves the problem for five different initial guesses and selects the best solution it has found. The initial guesses for NL-5 are as follows: one warm start as in NL-1, one random point within the domain, one point at the lower bound, one at the upper bound, and one in the geometric center. The solution for each control time step is fed to the system via a receding-horizon strategy: while the control optimization problem finds the optimal input signal during the next  $N_p$  steps, we only use the solution for the first step.

The computational performance of the two NLP controllers is compared against MPC controllers based on combinations of three hybrid models (R, S, and T) with four constraint approximations: two polytopic (RMP and BMP) and two ellipsoidal (REL and BEL). In total we have six MILP and six MIQCP controllers to compare against their NLP counterparts. The hybrid MPC controllers are labeled by combining the abbreviations for their model and constraints, separated by a dash (–) symbol, as described in Table 7.1. Further, MILP and MIQCP controllers can be constructed using our published hybridization toolbox [87].

### 7.3.4 REFERENCE TRAJECTORY

We compare the computational performance of the nonlinear and hybrid MPC controllers during five maneuvers of two seconds as reference trajectories. These maneuvers are selected to represent aggressive evasive maneuvers with different longitudinal velocities as explained in Table 7.2. Note that for a more intuitive representation of distance to the handling limits and/or tire saturation, we indicate the normalized distance of the reference trajectory in the  $g$ - $g$  diagram (6.16) and Kamm circles (6.17) in their respective column where 1 represents the boundary of the circles, i.e. the constraints. The column  $\beta$ - $r$  represents the normalized location of the stability envelope often used in the literature as  $|r| \leq \mu g v_x$  under the assumption of steady-state cornering conditions as a typical yaw rate constraint [12] and  $|\beta| \leq 5\text{deg}$  [8]. Note that the steady-state cornering conditions do not hold in extreme maneuvers. Nevertheless, we provide the trajectory plots on the  $\beta$ - $r$  envelope to gain more insight in the cornering behavior of the vehicle during the reference

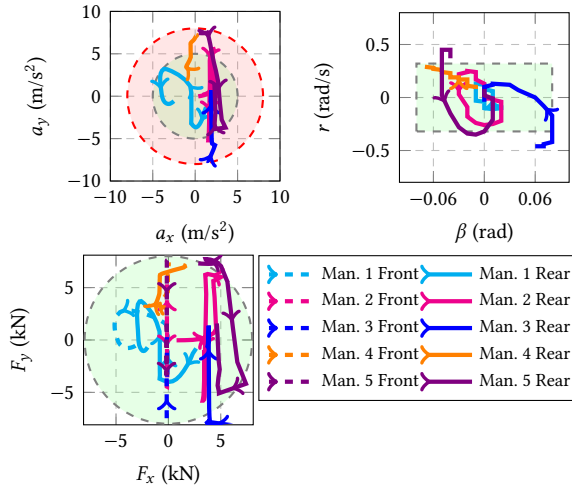


Figure 7.1: Selected maneuvers for the benchmark, represented in terms of the constraints. The green zone in the g-g diagram represents the safe region and the red one corresponds to the aggressive yet acceptable acceleration range.

maneuvers. Moreover, the normalized values represent the distance to the boundary with values between 0 and 1, where 1 indicates the position of the boundary itself. Figure 7.1 shows the schematic view of the five reference maneuvers in terms of these constraints. The two-seconds simulation time is selected to represent the recovery window for the controller in hazardous scenarios in case of an abrupt change in the reference trajectory.

Table 7.2: Selected maneuvers as reference trajectories for the benchmark. The  $v_x$  column represents the average longitudinal velocity in km/h.

No.	Maneuver	$v_x$	$\mathbf{g-g}$	$\beta\text{-r}$	Kamm
1	Safe lane change	130	0–0.5	0–0.4	0–0.7
2	Aggressive lane change	128	0–0.6	0–0.9	0–0.9
3	Drift cornering	73	0.2–0.8	0–0.9	0–1.0
4	High-speed cornering	154	0.3–0.7	0.3–1.1	0.4–0.9
5	Low-speed cornering	75	0.2–0.8	0.2–0.8	0–1.0

### 7.3.5 DRIVING SCENARIOS

We compare the computational performance of the nonlinear and hybrid MPC controllers in the following four driving scenarios:

- **Ideal Case:** The nonlinear prediction model is selected as the real system. The computational performance of the hybrid controllers is evaluated over  $N_p \in \{5, 10, \dots, 30\}$ .

- **Friction Offset:** We use a different tire-road friction in the real system as  $\mu \in \{0.70, 0.75, \dots, 1.00\}$  and compare the computational performance of the controllers for a selected  $N_p$  value over this range of friction offsets.
- **Friction Disturbance:** We assume the road friction for the second quarter of the maneuver to be very low, representing a disruption such as a slippery road surface and we compare the computational efficiencies in the same fashion as the friction offset case.
- **Handling Limits:** We investigate the computational performance of the controller for a fixed friction offset during the lane change maneuver (no. 1 in Table 7.2). We vary the input steering angle during the maneuver to simulate different levels of aggressive steering and assess the closed-loop performance in terms of the closeness to handling limits, i.e., the boundary of the  $g$ - $g$  diagram with the acceleration magnitude between  $0.5g$  up to the  $\mu g$  limit.

Note that  $\delta$  is the steering angle on the road and not the steering wheel angle, and since the scenarios are high speed where compared to an urban scenario, a smaller steering angle can lead to a higher lateral acceleration. For instance, the steering angle bound is approximately 10 degrees in [8] and 20 degrees in [152] or the extreme maneuvers in [114]. For drifting, to the best of our knowledge, the maximum bound is 38 degrees in [153] where the velocity is 54 km/h, while we investigate 75 up to 154 km/h maneuvers. Therefore, we believe 30 degrees is a suitable upper bound for the steering angle for extreme maneuvers in highway scenarios.

### 7.3.6 SOLVER SELECTION

For a fair comparison in terms of computation time, we select the most efficient known solvers within the academic community for the MILP/MIQCP and NLP problems.

The MILP/MIQCP and NLP problems are solved by GUROBI [154] and TOMLAB/KNITRO [155] optimizers, respectively, using MATLAB as interface and overall computation environment. To further improve the solution time for the NLP problems, we provided the objective and constraint functions via MEX files (instead of m-files in MATLAB), which in our experiments reduced the computation time for the NLP problems by around 50% for all the cases.

The simulations were all run on a PC with a 4-core(s) Intel Xeon 3.60 GHz CPU and 8 GB RAM on Windows 10 64-bit and in a MATLAB R2020b environment.

## 7.4 SIMULATION RESULTS

Using the benchmark described in Section 7.3, we compare the computational performance of the hybrid and nonlinear MPC controllers as follows: we first compare these controllers in the ideal case for different  $N_p$  values and then we select the most promising hybrid controllers and compare their robustness to friction uncertainty and their performance close to the handling limits in the next subsections.

### 7.4.1 IDEAL CASE

The computational performance of the nonlinear and hybrid MPC controllers is shown in Figures 7.2 and 7.3 in terms of their average and maximum tracking errors and computation time per control sampling time.

#### AGGRESSIVE LANE CHANGE MANEUVER

First, we start by comparing the hybrid and nonlinear MPC controllers during the aggressive lane change maneuver in Fig. 7.1.

In this maneuver, all MILP controllers, as well as NL-5, perform equally well with different  $N_p$  values. However, NL-1 shows oscillatory tracking accuracy across the  $N_p$  axis. This happens because extending the prediction horizon, even with the same prediction model as the ground truth, makes the search for an optimal solution more difficult, which reduces the chance of reaching an “acceptable” solution when only using the warm-start strategy, especially when performing the aggressive lane change maneuver as shown in Fig. 7.1.

Extending the prediction horizon not only enlarges the optimization problem’s search space but also causes the prediction model error to accumulate, which also contributes to higher computation times as the reference trajectory may become difficult for the prediction model to track. Therefore, the suitable prediction horizon for tracking, in terms of acceptable accuracy for lower computation times, is 10 or 15.

In terms of computation speed, the MILP controllers with the T model show a steady increase rate similar to the nonlinear ideal models while other hybrid controllers show an increase in the rate of computation time after a certain  $N_p$  value, which stems from the prediction model accuracy. Longer prediction horizons do not only increase the dimensions of the search space in the optimization problem, but also lead to accumulation of the prediction model error. This accumulation increases the error as well as computation time as the reference trajectory can become infeasible to track for the prediction model, leading to slower convergence. Therefore, the T model yields the best accuracy compared to the R and S models.

Controllers with the T model in general show the lowest increase of computation time when compared to the other hybrid controllers. In the MIQCP controllers T-REL and T-BEL the rate of increase is higher for  $N_p$  between 5 and 10, but it converges to the same rate as the NLP controllers and their MILP counterparts T-RMP and T-BMP. This stems from the limitations of using the ellipsoidal constraints compared to the polytopic ones: the ellipsoidal approximations of the feasible region have a lower coverage of the feasible region (for more details, see Table V in Chapter 6), which limits using the full control potential for an aggressive maneuver in shorter prediction horizons.

As Fig. 7.2 shows, the MILP controllers exceed the performance of the MIQCP ones in terms of accuracy, as well as computation speed. Therefore, for the next simulations we only consider the MILP controllers as prospective suitable hybrid candidates.

#### CORNERING MANEUVERS

We compare the MILP and NLP controllers during the three cornering maneuvers as shown in Fig. 7.3. In maneuvers 3 and 5 where the input forces vary drastically over the maneuver (see Fig. 7.1), the NL-1 controller shows a poor computational performance and oscillatory

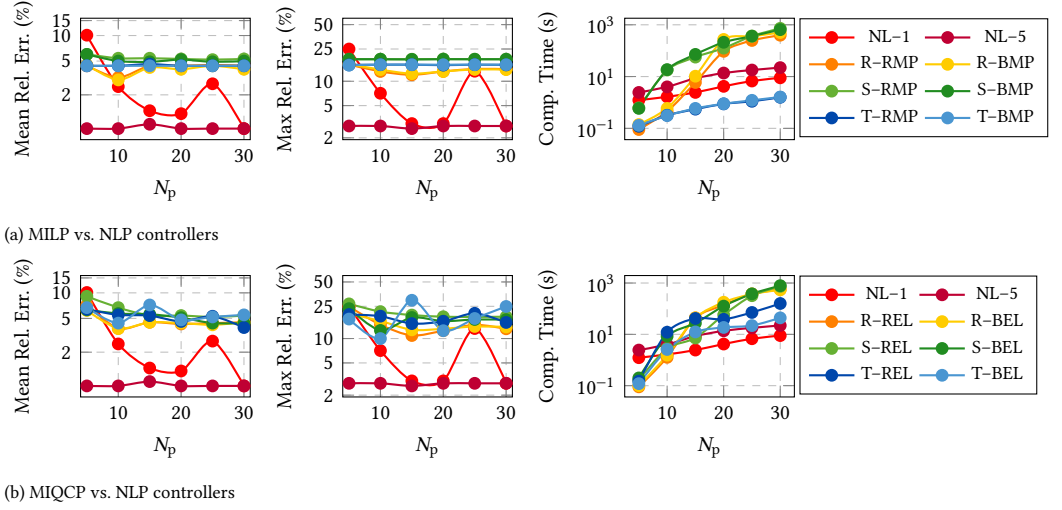


Figure 7.2: Computational performance of the nonlinear and hybrid MPC controllers during the aggressive lane change maneuver (maneuver 2 in Table 7.2) in the ideal scenario.

behavior in the error plots across the  $N_p$  axis, which was also observed in the aggressive lane change maneuver and discussed there.

In all three cornering maneuvers, the controller with the T model yields the best computational performance with its mean tracking error below 4.5% and maximum error below 10% in all cases.

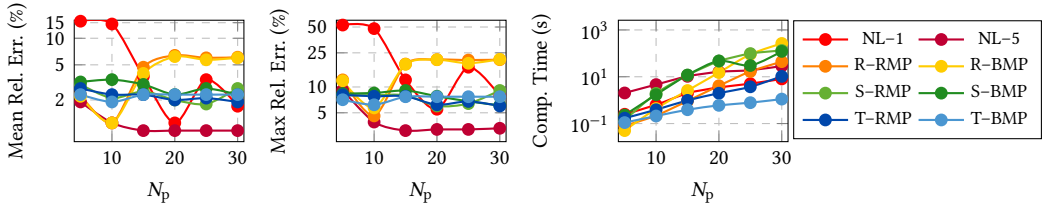
Just as for the lane change maneuvers, increasing  $N_p$  leads to a higher computation time for all the controllers; however, the rate of increase is the lowest for the nonlinear MPC and the T-BMP controller. For the T-RMP controller the same behavior is observed for  $N_p < 20$ . Comparing the performance of the hybrid and nonlinear MPC controllers in all the five maneuvers, the suitable prediction horizon for tracking, in terms of acceptable accuracy for lower computation times, is 10 or 15. Next, we select  $N_p = 10$  for the comparison of the robustness of the controllers to friction uncertainty. However, we also have simulated other  $N_p$  values, reaching similar results. Therefore, for a compact presentation, we present the trends and analyze them for  $N_p = 10$ . In addition, since the MILP controllers with the S prediction model show larger tracking errors and larger computation times, especially for shorter prediction horizons, we disregard them at this stage and compare the four MILP controllers with R and T prediction models against their NLP counterparts.

**Remark 7.3.** We have simulated the scenarios for other  $N_p$  values and reached similar results. Therefore, for a compact presentation, we present the trends and analyze them for  $N_p = 10$ .

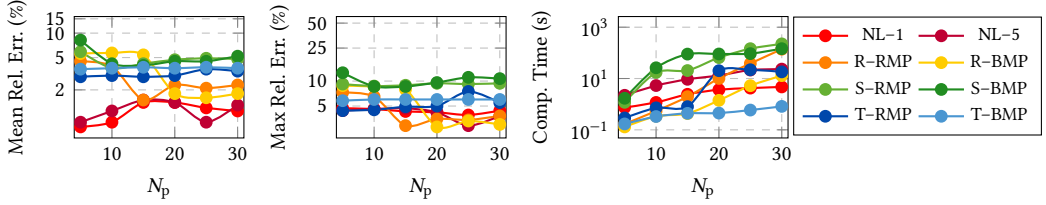
#### 7.4.2 FRICTION OFFSET

The maneuvers 2, 3, and 5 in Table 7.2 are the three most critical ones: here, the vehicle operates close to the tire saturation and stability limits almost the whole time as shown in Fig. 7.1. Thus, we used these maneuvers to study the effect of friction offset. The

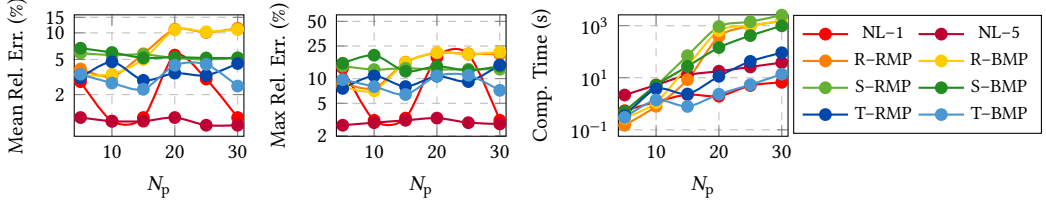




(a) Drift cornering maneuver (maneuver 3 in Table 7.2)



(b) High-speed cornering maneuver (maneuver 4 in Table 7.2)



(c) Low-speed cornering maneuver (maneuver 5 in Table 7.2)

Figure 7.3: Computational performance of the nonlinear and MILP MPC controllers during three cornering maneuvers (maneuvers 3, 4, and 5 in Table 7.2) in the ideal scenario.

prediction horizon is selected as  $N_p = 10$  and the simulations are run for different road friction coefficients in the range  $\mu \in \{0.70, 0.75, \dots, 1.00\}$  to account for uncertain friction in the prediction model. Figure 7.4 shows the computational performance of the nonlinear and hybrid MPC controllers during the three reference maneuvers.

While the computation time for the hybrid controllers does not vary by increasing the friction uncertainty, the nonlinear controllers show an increase in the computation time in maneuvers 3 and 5 where a significant fraction of the maneuver is performed close to the tire-saturation and vehicle-handling limits, which are functions of the friction coefficient.

The maximum and mean tracking errors increase for lower friction coefficients for all the controllers. However, the rate of error increase for the nonlinear controllers is higher. The difference between the tracking errors of NL-1 and NL-5 once again indicates the shortcoming of a warm-start strategy during aggressive maneuvers in searching for the optimal solution in the search space. This however comes at the price of an increase in computation times, best shown in Fig. 7.4c where solving the NLP for five initial guesses increases the computation time tenfold.

**Remark 7.4.** *The reason behind the computational increase in NLP is as follows: compared*

to the shifted solution to the previous step, the other initial guesses are generally further away from a local optimum. As a result, the increase in computation time is more than linear.

In the presence of friction offset, the tracking error of NL-5 converges to that of the MILP controllers in maneuvers 2, 3, and 5 where a more extensive search over the search space is required to perform the maneuver from an initial state with an error from the previous solution. To understand this phenomenon and its two contributing factors, we look at the NL-5 and T-BMP controllers in more detail.

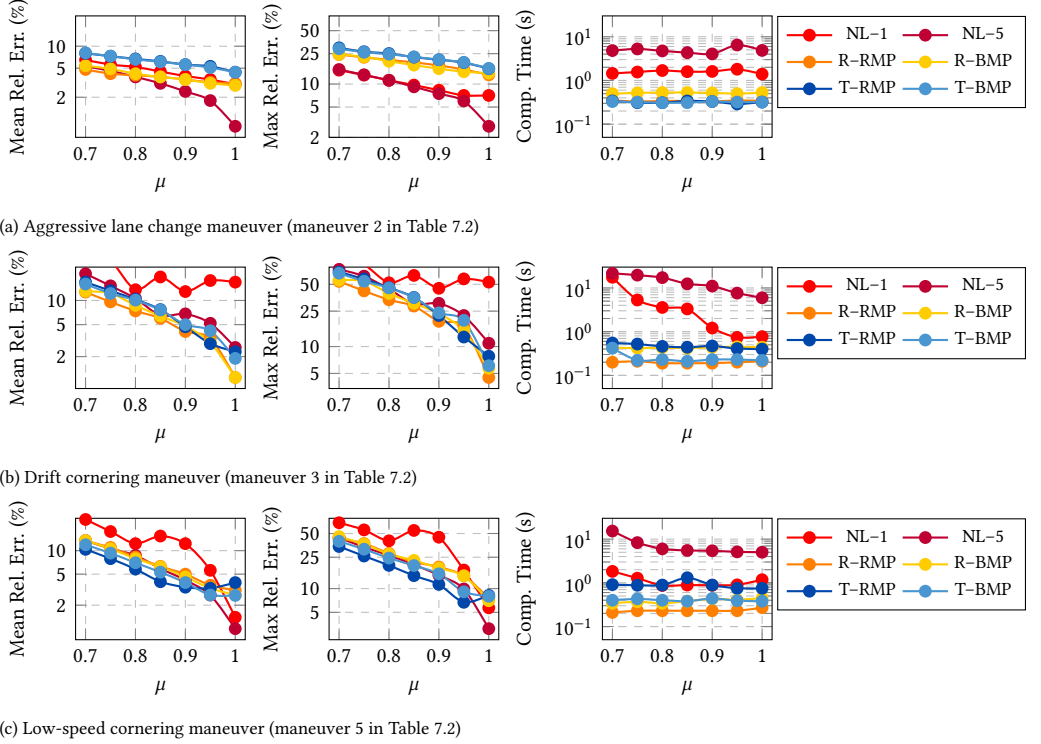


Figure 7.4: Computational performance of the nonlinear and hybrid MPC controllers during three reference maneuvers (maneuvers 3, 4, and 5 in Table 7.2) in the friction offset scenario.

Notice that NL-5 and T-BMP have mean tracking errors below 5% in all the maneuvers of Fig. 7.4 with  $\mu = 1$ , which generates the same friction as their prediction models. When  $\mu$  on the road is decreased to 0.7, the controllers still seek to find the solution (including tire forces) close to the boundary of the feasible region of their model, which assumes  $\mu \approx 1$ . However, these forces cannot be generated by the tire in the real system due to the lower friction on the road. Therefore, the first contributing factor to the error is the fact that error accumulation grows exponentially with the number of control time steps and as a result the controller converges to an infeasible solution for the real system (note that the real feasible region is shrinking with the friction reduction). Secondly, with larger errors,

finding a feasible solution to track the reference trajectory from an initial state with an already large tracking error might not be possible after a certain error bound. This not only increases the convergence time for the NL-5 controller, but also results in converging to even worse solutions both in terms of constraint violation and optimality to the point where we observe the tracking error of NL-5 exceeds the error for T-BMP in Figures 7.4b and 7.4c as with a similar order of model error, the branch-and-bound approach of the MILP solver, as opposed to the NLP solver, guarantees convergence to the global optimum if given enough time, while keeping the computation time as low as for the ideal case.

### 7.4.3 FRICTION DISTURBANCE

In this case, we assume a correct knowledge of the road friction during the maneuver, while exposing the system to a sudden friction reduction to  $\mu = 0.4$  during the second quarter of the maneuver and we evaluate the tracking error, computation times, and the recovering ability of the closed-loop system without any estimation or corrections during the disturbance.

The tracking errors of the controllers are compared in case of sudden reduction of the friction to  $\mu = 0.4$  in the second quarter of the maneuver to represent a case similar to pouring water on the road surface. The tracking errors at each time step are shown in Fig. 7.5 to compare the errors, as well as the ability of the different controllers in recovering from the friction disturbance. The average computation times per control step for each controller are presented in a separate plot in Fig. 7.5f.

During the safe lane change maneuver, all controllers recover to a tracking error below 5% after five time steps, while the tracking error is larger during the other maneuvers that are more aggressive. After the friction disturbance in maneuvers 2 and 5, the tracking error keeps on increasing as the controllers fail to recover to an acceptable error bound. This could be understood by taking into account that the second quarter of maneuver 2 is when tracking the reference trajectory requires tire forces that will no longer be feasible due to the fact that the radius of the Kamm circle for the rear tire has decreased to 40% of its original value, which means that following the planned trajectory will no longer be feasible for the prediction model. The same issue arises during maneuver 5 that can be tracked provided that the rear tire is generating forces close to its saturation limit during the whole maneuver (Fig. 7.1).

The hybrid controllers show smaller tracking errors after recovery during the low-speed cornering maneuver while starting with larger tracking errors before the friction disturbance. This reduction is understandable in light of the fact that the second half of this maneuver requires the vehicle to operate further from the stability boundaries in  $\beta$ - $r$  envelope and the g-g diagram in Fig. 7.1. Meanwhile, the tracking error of NL-1 stays above 5%, which shows the limitations of depending on a warm-start strategy in convergence to better optima. This limitation is even more clear in Fig. 7.5c for the drift cornering maneuver: while NL-5 recovers better after the disturbance, it still fails to reach smaller tracking errors as fast as the hybrid MPC controllers, which suggests that increasing the number of initial guesses by considering more than five points could improve the convergence capability of NL-5 to acceptable optima. However, it should be noted that even for five initial guesses, NL-5 requires 10 times more than the slowest hybrid MPC to converge to its best solution.

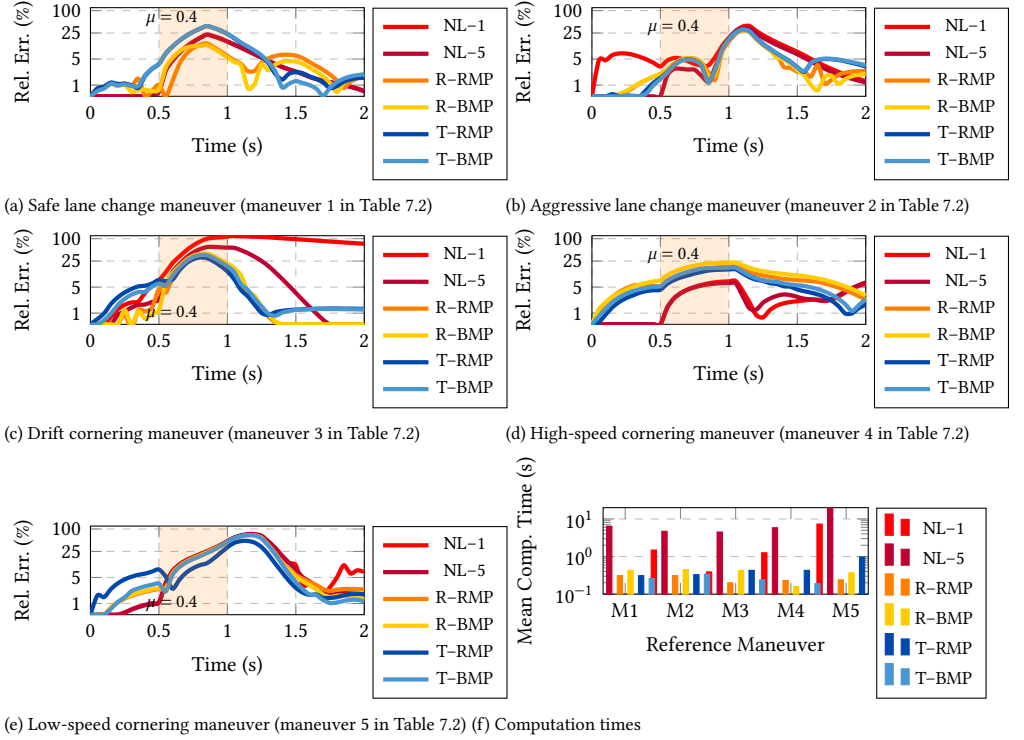


Figure 7.5: Tracking error of the nonlinear and four selected MILP MPC controllers during five reference maneuvers in (a) to (e) in case of friction disturbance. The average computation time for each control time step is shown in (f).

In the safe lane change maneuver, T-RMP and T-BMP show larger tracking errors compared to the other controllers, and this is the scenario where the hybrid controllers in general show the highest error increase of 12% to 30% for 95% computation time reduction. However, the effectiveness of the hybrid MPC controllers in terms of tracking error is more observable in more aggressive maneuvers where T-RMP and T-BMP show a better performance, in some cases even better than that of NL-1 and NL-5, due to the fact that the shortcomings of convergence to local optima is more clear in hazardous scenarios where there are sudden changes in the environment that require a more thorough search across the decision space. Comparing the control performance vs. computation time trade-off during the four aggressive reference maneuvers shows that choosing the hybrid MPC controllers R-BMP and R-RMP decreases the computation time to 2% to 5%, while it increases the maximum error from 5% to 15% in maneuver 5 while decreasing it during maneuvers 2, 3, and 4 compared to NL-5.

#### 7.4.4 ANALYSIS OF COMPUTATIONAL PERFORMANCE

Figure 7.6 plots the range of tracking error and computation times for the NLP and four MILP MPC controllers during all the five reference maneuvers in both friction offset and

friction disturbance scenarios as shaded boxes. The average points are shown by square markings in the shaded area. Comparing the computational performance in Fig. 7.6 shows the power of the hybrid MPC controllers compared to the nonlinear ones in the presence of uncertainty. While the NLP controllers have a lower optimum, their maximum tracking errors reach much higher values while taking more time to converge. In terms of the average points, not only the best MILP controller in the ideal case (T-BMP) has a lower maximum error in both friction uncertainty cases compared to the best NLP one (NL-5), but it also has a higher computational efficiency: in the friction offset case, it trades off an error increase from 8.7% to 9.7% for reducing the computational time from 10.2s to 0.3s, and in the disturbance case it reaches a smaller tracking error (from 9.5% to 8.7%) as well as a lower computation time (from 8.3s to 0.4s).

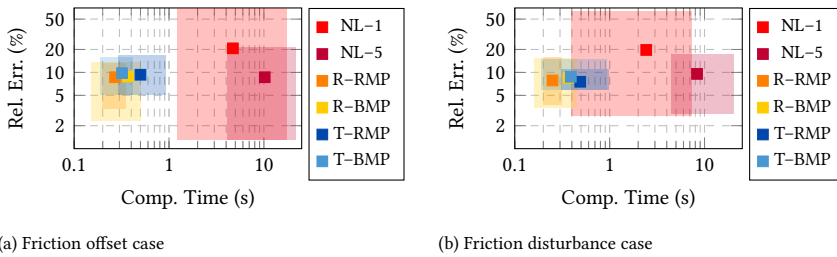


Figure 7.6: Relative errors and computation times for the nonlinear and four MILP MPC controllers during the five reference maneuvers in cases with friction uncertainty.

### 7.4.5 HANDLING LIMITS

For a more clear understanding of the computational performance during hazardous scenarios, we perform a second test in the friction offset case. Here, we fix the road friction to  $\mu = 0.85$  and compare the computational performance of the controllers during the safe lane change maneuver over a range of steering actions to get closer to the boundary of the g-g diagram in Fig. 7.1. Figure 7.7 shows the tracking errors and computation time for MILP and NLP controllers for different levels of aggressiveness in terms of the acceleration magnitude  $\sqrt{a_x^2 + a_y^2}$ , which is bounded by  $\mu g$ .

The computational efficiency of hybrid MPC can be easily seen in Fig. 7.7. The closer the maneuver gets to the boundary of the stability constraint in the g-g diagram, the higher the tracking errors get for all the controllers. However, the error increase for the NLP controllers is much higher as they fail to converge to an acceptable optimum with 1 or even 5 starting points. Therefore, when the acceleration magnitude exceeds  $6.2 \text{ m/s}^2$ , even the NL-5 controller reaches higher mean and maximum tracking errors compared to the T-BMP, R-BMP, and R-RMP controllers. This is happening while NL-5 requires about 20 times more time to converge to its final solution.

Comparing among the MILP controllers, T-BMP can be the best hybrid MPC controller as it shows the most steady computation time and the lowest increase of tracking error as the steering action gets more extreme in our simulations.

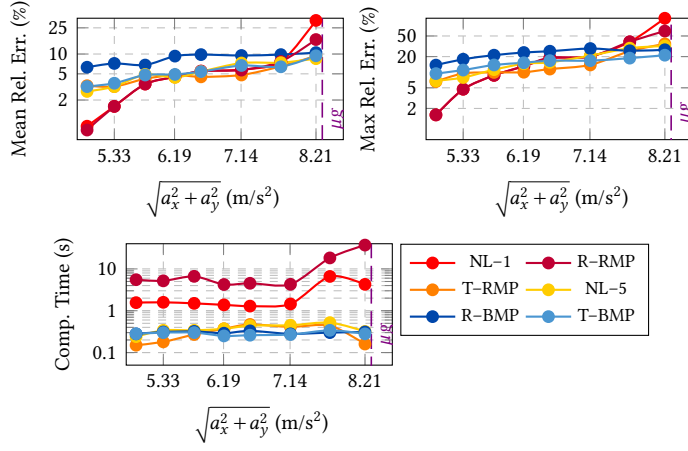


Figure 7.7: Computational performance of the nonlinear and hybrid MPC controllers during lane change maneuver in the friction offset scenario for different levels of aggressive steering in the g-g diagram.

## 7.5 IPG CARMAKER SIMULATION

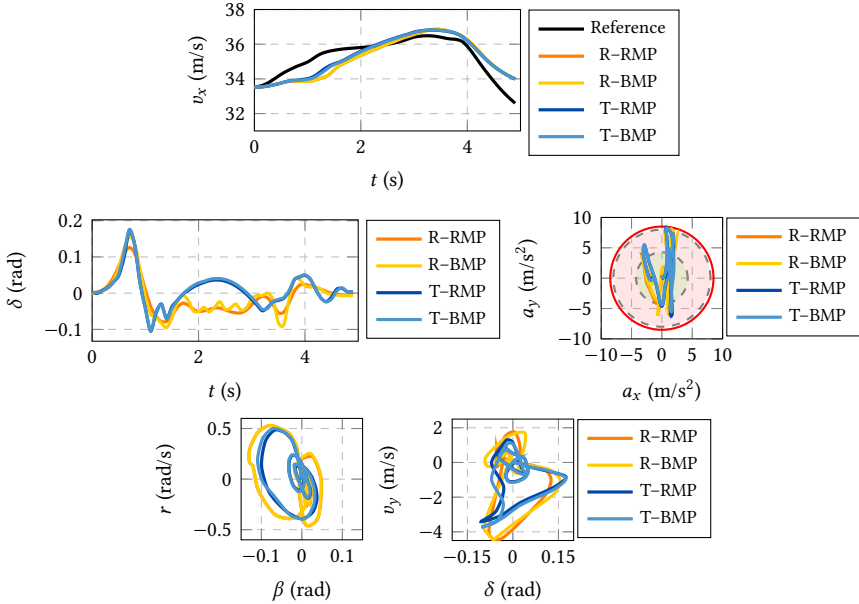


Figure 7.8: Computational performance of the nonlinear and hybrid MPC controllers during lane change maneuver in the friction offset scenario for different levels of aggressive steering in the g-g diagram.

To validate the performance of the MILP controllers, we have simulated the most aggressive double-lane change maneuver in handling-limit scenario using a high-fidelity BMW model in IPG CarMaker with the RealTime tire model. Figure 7.8 shows the state and input plots, as well as the tire force and constraint bounds from the simulations. Despite the fact that the MILP controllers use a much simpler prediction model and friction offset, comparing the states shows that they are able to provide control inputs that satisfy the constraint boundaries in the handling limits as shown on the  $g$ - $g$  diagram.

## 7.6 DISCUSSIONS AND OUTLOOK

### 7.6.1 MILP vs. MIQCP

In general, as MILPs are solved faster than MIQCPs, MILP-based approaches are more suitable choices in terms of computation speed. The control performance highly depends on the accuracy of the hybrid approximation. Consequently, the tracking performance of MIQCP controllers was not as good as MILP controllers due to their less accurate constraint approximation compared to their MILP counterparts. Nevertheless, for systems or applications where a lower approximation error with mixed-integer quadratic constraints is obtained for the nonlinear constraints, MIQCP can be an efficient formulation to hybridize the nonlinear MPC problem. This can for instance be the case for systems with quadratic or bi-variate nonlinearities where considering the quadratic terms helps avoid using large number of local linear or affine modes to obtain the same level of accuracy.

### 7.6.2 ROBUSTNESS TO FRICTION UNCERTAINTY

When friction uncertainty is present, MILP controllers do not require longer computation times, whereas NLP controllers take significantly longer to converge. In addition, the rate of increase in tracking error for the NLP controllers is higher than that of the MILP controllers in the presence of a friction offset.

Furthermore, MILP controllers are better able to recover from friction disturbances due to their more comprehensive search for an optimal solution in the decision space and the fact that they always reach the global optimum if given enough time. This means that if the error is already very high in the initial state for the current time step, the NLP controllers may not find a feasible solution at all in the vicinity of their initial guesses, while the MILP controller will converge to one via branch-and-bound strategy, provided that they have given sufficient time. As a result, even in cases where both nonlinear and hybrid controllers recover from high tracking errors during the friction disturbance, the MILP controller reaches smaller tracking error in fewer control time steps.

The robustness of the MILP controllers compared to the NLP controllers is summarized in Fig. 7.6. While NLP controllers can reach lower relative errors, their behavior in terms of accuracy, as well as computation speed, is not as consistent as MILP controllers in the presence of friction offset or disturbance. However, MILP controllers are considerably (at least 10 times) quicker to converge to their optimal solution and show significantly less variations in the relative error when exposed to friction uncertainty.

### 7.6.3 PERFORMANCE CLOSE TO HANDLING LIMITS

Getting closer to the handling limits leads to larger tracking errors for all the controllers. However, NLP controllers may deviate from the reference significantly as they may fail to converge to an acceptable optimum. Meanwhile, the MILP controllers converge to lower errors in a shorter time, e.g., in the boundary of the acceleration magnitude, the best MILP controller converges to 30% of the tracking error of the best NLP controller with 95% reduction of its corresponding computation time. This shows that MILP controllers are more suitable choices for real-time control in emergency evasive maneuvers.

### 7.6.4 OVERALL COMPUTATIONAL PERFORMANCE

The shortcoming of a warm-start strategy in solving an NLP is more clear in emergency maneuvers, which stems from its limitation in searching for the optimal solution in the search space during aggressive maneuvers. In addition, even with a multi-start strategy, the NLP controller may converge to worse solutions if the uncertainty influences the feasible region and the NLP solutions become infeasible for the real system. However, the branch-and-bound approach of the MILP solver has a better exploration of possible solutions than a potentially real-time implementable NLP solver such as gradient-based solver, while keeping the computation time as low as for the ideal case.

Even in the ideal case where the NLP controllers benefit from employing the same prediction model as the real system, they show high variations in tracking control and computation time, which means they are not suitable options for robust control in hazardous scenarios. In the presence of uncertainty, the NLP controllers suffer from larger tracking errors as well as an exponential growth in their computation time. However, the MILP controllers converge to smaller tracking errors, within a much smaller variation bound, and with significantly less computation times.

7

## 7.7 CONCLUSIONS

This chapter has presented a comparative assessment of nonlinear MPC controllers vs. their various hybridized counterparts in terms of computational efficiency for vehicle control during emergency evasive maneuvers. The hybridization of the nonlinear problem was presented and discussed in Chapter 6, where several guidelines for hybridization are given in a generalized framework.

The benchmark of this chapter uses three hybridized models and four hybridized constraint formulations for a nonlinear single track vehicle model considering nonlinear physics-based constraints for stability and tire-force saturation. Five reference maneuvers were selected to represent emergency situations where the computational efficiency is crucial for real-time proactive vehicle control. The hybrid and nonlinear controllers then were compared in multiple scenarios to compare their control performance and computation time, and their robustness in the presence of friction uncertainty in the form of an offset or a disturbance. Further, we studied the tracking behavior of the controllers with respect to how close the vehicle is operating in handling limits. The conclusions of our comparative assessment are summarized next with respect to different criteria.

Based on our comparative assessment, we propose combining hybrid MPC and hybrid predictive estimation techniques (e.g., moving-horizon estimation) as a potential next re-



search step for improving robustness in hazardous driving scenarios. Moreover, as quadratic forms of nonlinearity are extensively encountered in modeling of physical systems, we propose investigating piecewise-quadratic-based hybridization of the prediction model and physics-based constraints for MIQCP formulation of MPC optimization problem. This can particularly be beneficial for systems with nonlinearities that can better be approximated using quadratic approximations and can lead to significant improvements in terms of accuracy and computational efficiency of the hybrid controller. In addition, we suggest extending the prediction model to include the effects of wheel-speed/torque dynamics for improved control performance in hazardous scenarios. Finally, we suggest hardware implementations of the MILP tracking controllers as a topic for future research.



## 8

# EFFICIENT RESPONSE TO SUDDEN APPEARANCE OF STATIC OBSTACLES

*When they invented the car they invented the collision and the darkness of what time leads the willing body into.*

— David Wojnarowicz, *Close to the Knives: A Memoir of Disintegration*

*The sudden appearance of a static obstacle on the road, i.e. the moose test, is a well known emergency scenario in collision avoidance for automated driving. Model Predictive Control (MPC) has long been employed for planning and control of automated vehicles. However, real-time implementation of automated collision avoidance in emergency scenarios such as the moose test is still an open issue due to the high computational demand of MPC to perform an evasive maneuver in such hazardous scenarios. This chapter brings new insights into real-time collision avoidance via experimental implementation of MPC for motion planning after sudden and unexpected appearance of a static obstacle. As state-of-the-art nonlinear MPC shows limited capability to provide an acceptable solution in real time, we propose a human-like feedforward planner to help in cases where the MPC optimization problem is either infeasible or unable to yield a suitable solution due to the poor quality of the initial guess. We introduce the concept of maximum steering maneuver to design the feedforward planner and to mimic a human-like reaction after the unforeseen detection of the static obstacle on the road. Real-life experiments are conducted in a variety of scenarios with different speeds and level of emergency using an FPEV2-Kanon electric vehicle. Moreover, we demonstrate the effectiveness of our planning strategy via comparison with a state-of-the-art MPC motion planner.*

## 8.1 INTRODUCTION

Motion planning in automated driving has been extensively researched during the past years. Avoiding a collision in hazardous scenarios is particularly challenging on the operational and stability levels [156], i.e. planning a safe trajectory for the ego vehicle and tracking it during an emergency scenario.

Among the various testing scenarios [157], one example of an emergency situation is the sudden appearance of a static obstacle on the road, which can also be considered as the extreme case of leading vehicle deceleration in [158]. Figure 8.1 shows a schematic view of influential elements for the planning problem in such scenarios. Given the considerable relative velocity involved, the time-to-collision criterion [159] in these circumstances proves insufficient for maintaining a safe distance through braking alone. Hence, executing a safe evasive maneuver may require performing an extreme steering action near the boundaries of the vehicle stability constraint, also referred to as driving at handling limits.

In the case of a static obstacle, two critical parameters are the distance to the object – often reflected in time-to-collision or distance-to-collision threat measures in the literature [159] – and the width of the obstacle. For instance, shorter distance to the obstacle with a larger width can both contribute to the criticality of the situation as the width determines the necessary lateral displacement for collision avoidance. Understanding the importance of a wider width is closely linked to the current states of the vehicle as the required braking or steering actions for achieving a specific lateral displacement depend on factors like the current vehicle speed or sideslip angle.

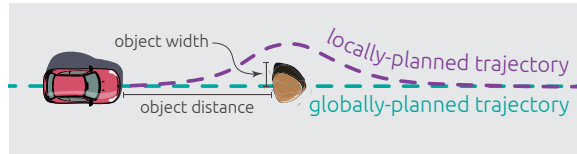


Figure 8.1: Elements of the collision avoidance problem after detecting a static obstacle

MPC has become increasingly popular in the field of automated collision avoidance thanks to its straightforward handling of constraints and its capacity to dynamically adjust to environmental changes by solving the control optimization problem in a receding horizon manner [12]. In the current state of the art, trajectory planning and vehicle control are commonly addressed through one of two architectural frameworks: hierarchical [160–166] or integrated [9, 167–171].

A hierarchical architecture offers greater flexibility in defining control problems and enables faster responses for real-time implementation, owing to the differing frequency and performance requirements at each level. However, the reference trajectory provided by the planner may not be attainable for the real plant. This is a critical issue in emergency cases where the feasible area for collision avoidance is limited and the vehicle is operating at its handling limits. Integrated planning and control circumvents this issue by treating both problems within a single optimization problem.

While integrated MPC design, in particular when used with Electronic Power Steering (EPS), allows the handling of two optimization problems in combination, it is essential to highlight a key distinction between the two architectures: in a hierarchical architecture, ad-

addressing the planning and control optimization problems can occur at different frequencies (e.g. planning at 5-10Hz and control at 50-100Hz). Conversely, an integrated architecture demands solving the integrated optimization problem at the control frequency, albeit with a higher computational demand compared to the control optimization problem.

An important source of computational complexity stems from the fact that the integrated MPC optimization problem is nonlinear, which requires employing computationally-demanding NonLinear Program (NLP) algorithms to find the – locally – optimal solution. In [169], Gaussian safe envelopes used in the integrated MPC problem are obtained via Gaussian processes regression; this formulation allows for efficient solution of the integrated MPC problem by Quadratic Programming (QP). Other examples of QP solution of the MPC optimization problem include [170] where constraining the decision space to the linear tire force range leads to a quadratic formulation of the problem, and [172] where the weights of the simplified quadratic problem are adapted online for improved control performance.

In [9], two models are serially cascaded to handle the two problems simultaneously, hence facilitating real-time solution of the optimization problem by NLP and a warm-start strategy. However, this strategy is limited in finding the optimal solution if a static obstacle suddenly appears on the road. Moreover, physics-based and local convexification approaches [165, 173] or explicit sub-optimal solution [167] have helped the real-time realization of integrated planning and control for normal driving. However, achieving real-time solutions in emergency scenarios remains a primary bottleneck of the current automated driving research.

Another technique in this area is incorporating parametric methods for trajectory planning into an integrated control optimization framework, which offers significant advantages by combining the strengths of both approaches: while the integrated architecture prevents the generation of unattainable trajectories for the control layer, the parametric formulation streamlines the optimization process significantly, facilitating fast convergence to a local optimum, which is essential for real-time planning and control. Parametric trajectories during a lane change maneuver are typically represented by polynomial curves at lower speeds [165, 166, 168, 170] or sigmoid [174] and tangent-hyperbolic functions [172] that replicate human behavior at higher speeds [175]. For instance, in the case of a sudden appearance of a static obstacle, the finite-state machine in [174] decides on simultaneous braking and steering to perform a lane change. The reference trajectory for such a maneuver is defined as a parametric sigmoid function and is optimized in real time via NLP.

Despite the fast-paced progress of the literature, there remains a gap in achieving real-time implementation of automated collision-avoidance in real-life emergency situations. This is primarily due to the *limited computational capacity of real systems* [12] and the low acceptance of the automated driving systems due to *lack of interpretability* [176].

In the context of avoiding collision after sudden appearance of a static obstacle, as well as limited memory and computation time, converging to – even a local – optimum via NLP is often not possible, which means that in practice the best feasible solution found before hitting a stopping criteria is used. As a result, the ‘quality’ of such a solution is sensitive to the provided initial guess. More specifically, the popular warm-start strategy that is often used in the literature, would be of limited value in such scenarios since the shifted solution

of the previous time step after detecting the obstacle is often far from a feasible solution to avoid the detected obstacle. Moreover, the limited computational resources restrict our ability to explore the decision space using other starting points. As a result, having a “good initial guess” is the practical way to improve the quality of the obtained solution by NLP. In this case, a good initial guess can be defined as an *interpretable* candidate solution, i.e. inspired by the normal reaction of a skilled driver to the sudden appearance of an obstacle ahead.

In this chapter, we tackle the problem of real-time collision avoidance with limited computational resources after the sudden appearance of a static obstacle. Therefore, we aim to provide experimental insights into the real-time implementation of MPC-based collision avoidance by confronting the computational limitation challenge head-on in real-life scenarios using an electric vehicle, and improving the computational efficiency of the problem by integration of a physics-based and human-like feedforward planner, to help convergence to a feasible solution in emergencies. We investigate multiple real-life emergency scenarios and analyze the effectiveness of our proposed approach in test cases that render previous solutions infeasible, emphasizing the need for our planning strategy. Hence, our approach integrates MPC more intelligently and provides effective solutions in the context of sudden appearance of a static obstacle.

The rest of this chapter is structured as follows: Section 8.2 describes the vehicle model, followed by an overview of the proposed control system design in Section 8.3. Section 8.4 then covers our proposed planning strategy. Details on system implementation are given in Section 8.5 and the results of the experimental tests are provided and analyzed in Section 8.6. Finally, Section 8.7 concludes this chapter.

## 8.2 VEHICLE MODEL

The single track vehicle model is shown in Fig. 8.2. First, we cover the kinematics of the vehicle and then we apply Newton’s second law to derive the governing equations of motion. Table 8.1 shows the variables and parameters used in this chapter.

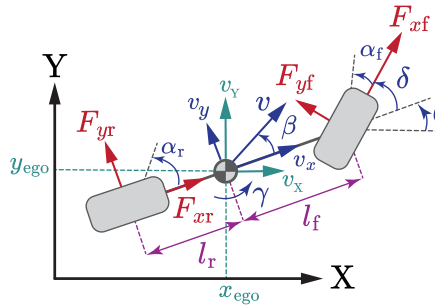


Figure 8.2: Single-track vehicle model

Table 8.1: List of variables and parameters

Variable	Definition	Units
$x_{\text{ego}}$	Global longitudinal position of the CoG	m
$y_{\text{ego}}$	Global lateral position of the CoG	m
$x_{\text{obs}}$	Global longitudinal position of the obstacle	m
$y_{\text{obs}}$	Global lateral position of the obstacle	m
$v_X$	Global longitudinal velocity of the CoG	m/s
$v_Y$	Global lateral velocity of the CoG	m/s
$v_x$	Local longitudinal velocity of the CoG	m/s
$v_y$	Local lateral velocity of the CoG	m/s
$\theta$	Yaw angle	rad
$\beta$	Sideslip angle	rad
$\dot{\gamma}$	Yaw rate	rad/s
$\delta$	Steering angle	rad
$\alpha_f$	Front slip angle	rad
$\alpha_r$	Rear slip angle	rad
$\kappa_f$	Front slip ratio	–
$\kappa_r$	Rear slip ratio	–
$F_{xf}$	Longitudinal force of the front tire	N
$F_{xr}$	Longitudinal force of the rear tire	N
$F_{yf}$	Lateral force of the front tire	N
$F_{yr}$	Lateral force of the rear tire	N
$F_{zf}$	Normal force on the front tire	N
$F_{zr}$	Normal force on the rear tire	N
Parameter	Definition	Value
$t_s$	Planning sampling time	0.2 s
$m$	Vehicle mass	925 kg
$I_{zz}$	Vehicle moment of inertia	617 kgm <sup>2</sup>
$J_{\omega r}$	Rotational inertia of the rear wheel	1.24 kgm <sup>2</sup>
$r$	Wheel radius	0.301 m
$l_f$	Distance between the front axle and the CoG	0.99 m
$l_r$	Distance between the rear axle and the CoG	0.71 m
$T_{\text{max}}$	Maximum torque	200Nm
$\delta_{\text{max}}$	Maximum steering angle	0.3rad
$\dot{\delta}_{\text{max}}$	Maximum steering angular speed	2618 rad/s
$C_x$	Longitudinal Pacejka tire parameter	1.5
$B_x$	Longitudinal Pacejka tire parameter	8.0
$C_y$	Lateral Pacejka tire parameter	1.4057
$B_y$	Lateral Pacejka tire parameter	7.1138

### 8.2.1 KINEMATIC MODEL

The trajectory of the ego vehicle in the global coordinates can be written as

$$\dot{x}_{\text{ego}} = v_X, \quad (8.1a)$$

$$\dot{y}_{\text{ego}} = v_Y, \quad (8.1b)$$

$$\dot{\theta} = \gamma, \quad (8.1c)$$

and the velocities in the local coordinate system attached to the vehicle body are obtained by

$$v_x = v_X \cos(\theta) - v_Y \sin(\theta), \quad (8.2a)$$

$$v_y = v_X \sin(\theta) + v_Y \cos(\theta). \quad (8.2b)$$

With the kinematic states and inputs defined by

$$s_{\text{kin}} = [x_{\text{ego}} \quad y_{\text{ego}} \quad v_X \quad v_Y]^T,$$

$$u_{\text{kin}} = [\dot{v}_X \quad \dot{v}_Y]^T,$$

the kinematic model of the vehicle can be written as

$$\dot{s}_{\text{kin}} = A s_{\text{kin}} + B u_{\text{kin}}, \quad (8.3)$$

where

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

## 8

### 8.2.2 EQUATIONS OF MOTION

The governing equations of motion for the vehicle model in Fig. 8.2 can be written as follows [8]:

$$\sum F_x = F_{xf} \cos(\delta) - F_{yf} \sin(\delta) + F_{xr}, \quad (8.4a)$$

$$\sum F_y = F_{xf} \sin(\delta) + F_{yf} \cos(\delta) + F_{yr}, \quad (8.4b)$$

$$\sum M_z = (F_{xf} \sin(\delta) + F_{yf} \cos(\delta)) l_f - F_{yr} l_r. \quad (8.4c)$$

Considering a small front steering angle, the dynamics of the single-track vehicle model is obtained as

$$\dot{v}_x = \frac{1}{m} (F_{xf} + F_{xr}) + v_y \gamma, \quad (8.5a)$$

$$\dot{v}_y = \frac{1}{m} (F_{yf} + F_{yr}) - v_x \gamma, \quad (8.5b)$$

$$\dot{\gamma} = \frac{1}{I_{zz}} (F_{yf} l_f - F_{yr} l_r). \quad (8.5c)$$



Since we expect that the evasive maneuver will generate forces beyond their maximum peak [177], we consider a nonlinear model for the lateral tire forces on the front and rear axles using the celebrated Pacejka tire model [90] as

$$F_{xf} = \mu F_{zf} \sin(C_x \arctan(B_x \kappa_f)), \quad (8.6a)$$

$$F_{xr} = \mu F_{zr} \sin(C_x \arctan(B_x \kappa_r)), \quad (8.6b)$$

$$F_{yf} = \mu F_{zf} \sin(C_y \arctan(B_y \alpha_f)), \quad (8.6c)$$

$$F_{yr} = \mu F_{zr} \sin(C_y \arctan(B_y \alpha_r)), \quad (8.6d)$$

with the front and rear slip angles respectively described by

$$\alpha_f = \delta - \beta - \frac{l_f \gamma}{v_x}, \quad (8.7a)$$

$$\alpha_r = \frac{l_r \gamma}{v_x} - \beta. \quad (8.7b)$$

### 8.3 PROPOSED CONTROL SYSTEM

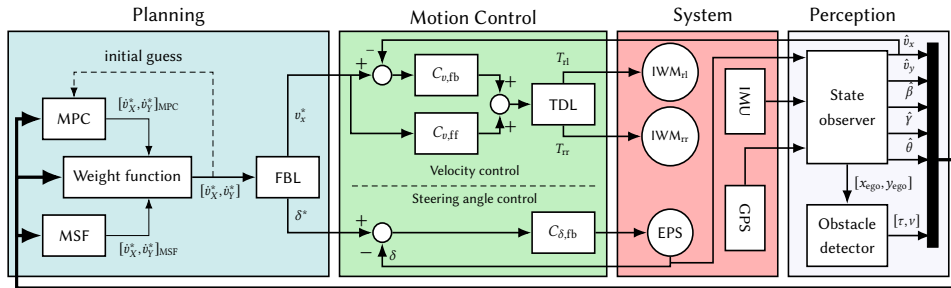


Figure 8.3: Closed-loop system architecture

Figure 8.3 shows the architecture of the proposed closed-loop system, consisting of the real system, as well as perception, planner, and controller modules, which will be described next.

**Perception** On the perception layer, an observer is incorporated to estimate the sideslip and yaw angles, while the obstacle detector obtains the position and width of the obstacle and returns the time to collision  $\tau$ , and the lateral steering index  $v$ , based on the measurements of the current state of the ego vehicle. The lateral steering index indicates the degree of extreme steering required to avoid a collision, and is covered in the next section in more detail.

**Planner** We propose a planner design strategy that leverages the capabilities of MPC in finding an optimal maneuver for collision avoidance, as well as an assistive Maximum Steering Feed-forward (MSF) planner designed to replicate a human-like response to the

detection of static obstacles. For the sake of computational efficiency, the kinematic model of the ego vehicle is used in the planner module. Therefore, both the MPC and the MSF planners provide their respective references for the longitudinal and lateral accelerations,  $\dot{v}_x^*$  and  $\dot{v}_y^*$ . A weight function then combines the two references and feeds the resulting reference signals to a FeedBack Linearization (FBL) controller to obtain the corresponding reference steering angle and longitudinal velocity.

**Controller** We use a Speed Controller (SPC) to obtain the required torque for tracking the reference longitudinal velocity. The output torque of the speed controller is then distributed via the Torque Distribution Law (TDL) to the rear left and rear right In-Wheel Motors (IWMs). In addition, the FeedBacked Electric Power Steering (FB-EPS) system tracks the steering reference by providing the steering angle signal to the steering motor.

## 8.4 DESIGN OF THE PLANNING STRATEGY

Given the measured and observed current state of the ego vehicle, along with  $\tau$  and  $v$ , the MPC and MSF planners provide their respective solutions,  $\dot{v}_x^*$  and  $\dot{v}_y^*$ , distinguished by their respective subscripts in Fig. 8.3. The same information is fed into the weight function that acts as a situation-assessment unit by distributing the weights between  $[\dot{v}_x^*, \dot{v}_y^*]_{\text{MPC}}$  and  $[\dot{v}_x^*, \dot{v}_y^*]_{\text{MSF}}$ . For instance, in cases where the solution of the MPC planner is close to the steering limits or if it fails to provide a solution in time, the weight function assigns a higher weight to the solution from the MSF planner. The design of each planner is discussed separately in the following sections.

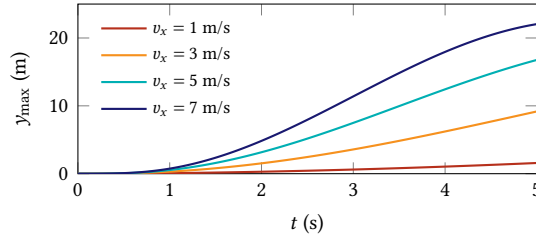


Figure 8.4: Maximum steering maneuver for different longitudinal velocities.

### 8.4.1 MSF PLANNER

We define the maximum steering maneuver  $y_{\max} : \mathbb{R}^2 \rightarrow \mathbb{R}$  to denote the resulting lateral position of the vehicle, as a function of time and its longitudinal velocity, after performing a maximum-possible evasive action without braking. This evasive steering is designed to mimic one of the human-like responses after detecting an obstacle: steering to the side as fast as possible.

The function  $y_{\max}$  is calculated for time  $t$  and velocity  $v_x$  by considering (8.1)-(8.7) to

solve

$$y_{\max}(t, v_x) = \int_0^t v_Y(\phi) d\phi, \quad (8.8a)$$

$$\text{s.t. } v_x(\phi) = v_x, \quad (8.8b)$$

$$\delta(\phi) = \min(\dot{\delta}_{\max}\phi, \delta_{\max}), \quad (8.8c)$$

$$v_y(0) = 0, \quad (8.8d)$$

$$\theta(0) = 0, \quad (8.8e)$$

$$\gamma(0) = 0, \quad (8.8f)$$

where  $v_Y$  is obtained by solving (8.8) and  $\dot{\delta}_{\max}$  and  $\delta_{\max}$  respectively represent the maximum steering rate and steering angles. The resulting lateral position is shown by solid lines in Fig. 8.4. Using  $y_{\max}$  and the width of the danger zone  $w$ , the lateral steering index  $\nu$  is defined by

$$\nu = \frac{w}{y_{\max}(\tau, v_x)}. \quad (8.9)$$

**Remark 8.1.** The zero initial conditions for the lateral velocity, yaw angle, and yaw rate in (8.8) represent a specific and extreme case. To define a more realistic model of the maximum steering maneuver,  $y_{\max}$  can be defined as a function of these initial conditions. However, this will lead to a higher-dimensional domain for the function  $y_{\max}$ , hence resulting in higher computational demand in the next steps. For the sake of computational efficiency, we use the zero initial conditions to merely account for the most extreme form of the maximum steering maneuver.

### 8.4.2 MPC PLANNER

We define the safety barrier  $y_{\text{safe}}$  using the sigmoid barrier function introduced in [174] in its extreme case as

$$y_{\text{safe}}(t) = \frac{w}{1 + \exp\left(\frac{x_{\text{obs}} - x_{\text{ego}}(t) - 4v_X(t)}{\epsilon \sqrt{w} v_X^2(t)}\right)}, \quad (8.10)$$

where  $w$  is the width of the danger zone, to be avoided by the ego vehicle's center of gravity and  $\epsilon = 1/\sqrt{8.8\mu g}$ . In the following example, we clarify the behavior of the  $y_{\text{safe}}$  function.

**Example 8.1.** Figure 8.5 shows a schematic view of the sigmoid barrier function (8.10) with the danger zone shown in red. For a given  $v_X$ ,  $y_{\text{safe}}$  is defined such that the vehicle's center of gravity has traveled  $w/2$  in the lateral direction 4 seconds before arriving at the longitudinal position of the danger zone. For instance, assuming a constant longitudinal speed for the vehicle in the global coordinates, with  $\tau = 6$  s shown in blue,  $y_{\text{safe}}$  passes through the lateral mid-point after 2 seconds. If the obstacle is closer, i.e.  $\tau = 5$  s as shown in orange, the resulting  $y_{\text{safe}}$  has the same curvature and ensures a  $4v_X$  distance before reaching  $w/2$ .

Using a discretized model of the vehicle and considering a prediction horizon of length  $N_p$ , we denote the input signal and the resulting predicted states over the prediction window

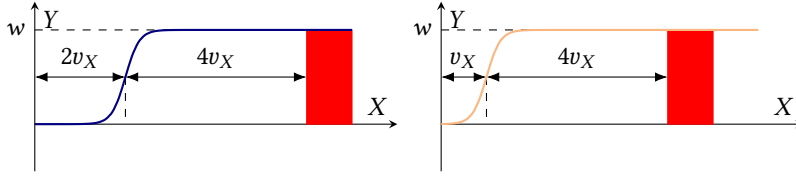


Figure 8.5: Sigmoid barrier function examples

by

$$\tilde{s}_{\text{kin}}(k) = [s_{\text{kin}}^T(k+1|k) \quad \dots \quad s_{\text{kin}}^T(k+N_p|k)]^T, \quad (8.11a)$$

$$\tilde{u}_{\text{kin}}(k) = [u_{\text{kin}}^T(k) \quad \dots \quad u_{\text{kin}}^T(k+N_p-1)]^T, \quad (8.11b)$$

where  $s_{\text{kin}}(k+i|k)$  for  $i \in \{1, \dots, N_p\}$  represents the predicted state at time step  $k+i$  based on the state measurement at time step  $k$ , i.e.  $s_{\text{kin}}(k|k)$ . We define four costs for each predicted step as

$$J_{\text{safe}}(k) = -\log(y_{\text{safe}}(k) - y_{\text{ego}}(k)), \quad (8.12a)$$

$$J_{\text{stable}}(k) = |v_Y(k)/v_X(k)|, \quad (8.12b)$$

$$J_{\text{brake}}(k) = |v_X(k) - v_{\text{des}}(k)|, \quad (8.12c)$$

$$J_{\text{steer}}(k) = |\dot{v}_Y(k)|. \quad (8.12d)$$

The function  $J_{\text{safe}}$  represents the barrier function for safe collision avoidance and the functions  $J_{\text{stable}}$ ,  $J_{\text{brake}}$ , and  $J_{\text{steer}}$  are defined to respectively minimize the sideslip angle for vehicle stability, the deviation from the desired velocity  $v_{\text{des}}$ , and the steering action in the planning optimization problem. The MPC cost function is then defined as

$$\mathcal{J}(k) = \sum_{i=1}^{N_p} \left[ \eta_1 J_{\text{safe}}(k+i) + \eta_2 J_{\text{stable}}(k+i) + \eta_3 J_{\text{brake}}(k+i) + \eta_4 J_{\text{steer}}(k+i) \right], \quad (8.13)$$

where the weights  $0 \leq \eta_j \leq 1$  with  $j \in \{1, \dots, 4\}$  yield a convex combination where  $\sum_{j=1}^4 \eta_j = 1$ . The resulting MPC planning optimization problem is given by

$$\tilde{u}_{\text{kin}}^*(k) = \underset{\tilde{u}_{\text{kin}}(k)}{\text{argmin}} \quad \mathcal{J}(k). \quad (8.14)$$

Note that the decision variable in (8.14) is the input vector and the state trajectory is defined within the cost function to get an unconstrained MPC optimization problem and to avoid infeasibility issues. MPC finds the optimal trajectory to avoid a collision by solving (8.14) at each time step in a receding-horizon fashion. This is done by solving the problem for the next  $N_p$  time steps, while providing the solution to the first step ahead to the controller.

**Example 8.2.** Consider a simple case of  $N_p = 2$  with  $u_{\text{kin}}(k) = u_{\text{kin}}(k+1)$ ,  $v_X(k) = v_{\text{des}}(k) = 5 \text{ m/s}$ ,  $x_{\text{ego}}(k) = y_{\text{ego}}(k) = y_{\text{obs}}(k) = 0 \text{ m}$ ,  $x_{\text{obs}} = w = 1 \text{ m}$  and  $v_Y(k) = 0 \text{ m/s}$ . The solution to (8.14) for input signals normalized on the bound  $[-1, 1]$  with the selections of  $\eta_j = 0.25$  is obtained as

follows:

$$\left. \begin{array}{ll} \eta_1 = 0.25, & \eta_2 = 0.25 \\ \eta_3 = 0.25, & \eta_4 = 0.25 \end{array} \right\} \Rightarrow u_{\text{kin}}^*(k) = \begin{bmatrix} 0.00 \\ 0.53 \end{bmatrix},$$

which indicates the optimal response would be no braking and steering rate equal to 53% of its maximum value. However, if the weight for  $J_{\text{safe}}$  is increased at the expense of reduction of the weights for  $J_{\text{brake}}$  and  $J_{\text{steer}}$ , the solution to (8.14) would change to

$$\left. \begin{array}{ll} \eta_1 = 0.50, & \eta_2 = 0.25 \\ \eta_3 = 0.05, & \eta_4 = 0.20 \end{array} \right\} \Rightarrow u_{\text{kin}}^*(k) = \begin{bmatrix} -0.35 \\ 1.00 \end{bmatrix},$$

corresponding to maximum steering rate with 35% of maximum braking.

### 8.4.3 FBL

To extract the reference steering angle and longitudinal velocity from  $[\dot{v}_X^*, \dot{v}_Y^*]$ , we obtain the required velocities in the local coordinate using (8.2) and we consider the vehicle dynamics in a feedback-linearization fashion as

$$\begin{aligned} \delta^*(k) &= \beta(k) + l_f \gamma(k) / v_x(k) \\ &\quad + F_{yf}^{-1} \left[ m \left( \dot{v}_y(k) + \gamma(k) v_x(k) \right) - F_{yr} \left( l_r \gamma(k) / v_x(k) - \beta(k) \right) \right] \\ v_x^*(k) &= \dot{v}_x(k) t_s + v_x(k). \end{aligned} \quad (8.15a)$$

## 8.5 SYSTEM IMPLEMENTATION

To evaluate the proposed planning strategy, we use an experimental vehicle FPEV2-Kanon driven by rear-IWMs shown in Fig. 8.6. The main parameters of the vehicle and the IWM are summarized in Table 8.1. The FPEV2-Kanon is equipped with an electric power steering system consisting of a speed-controlled Maxon brushless motor.

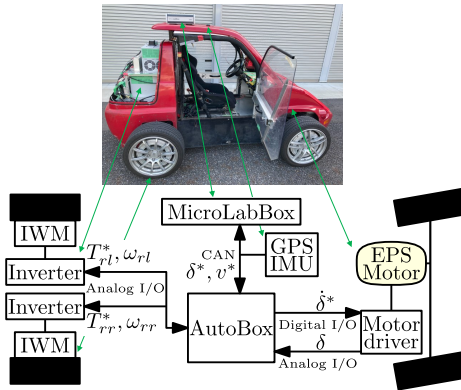


Figure 8.6: Experimental vehicle system configuration.

The motion control and planning modules are implemented in AutoBox (PPC 750GX 1 GHz, 32GB SDRAM program memory, 96MB SDRAM data storage) and MicroLabBox

Table 8.2: Optimization implementation parameters.

Parameter	Value
Maximum iteration	10
Maximum function evaluation	100
Constraint tolerance	0.001
Optimality tolerance	0.001
Step tolerance	0.001

(NXP QorIQ P5020 dual core 2GHz, 1GB DRAM, 128MB flash memory), respectively. The communication among these modules and the on-board Inertial Measuring Unit (IMU) and Global Positioning System (GPS) sensors is facilitated via a Controller Area Network (CAN), and set to 500kbps. The GPS, Autobox, and MicroLabBox measurements are respectively updated at a frequency of 1 Hz, 10kHz, and 5Hz. The MPC planner is implemented using `fmincon`'s Sequential Quadratic Programming (SQP) solver from the MATLAB R2017b Optimization toolbox. The optimization parameters are shown in Table 8.2.

The motion control module consists of two separate controllers for tracking the steering angle and the longitudinal velocity, as shown in Fig. 8.3. Considering the nominal plant transfer function

$$P_n(s) = \frac{1}{\left(J_{\omega r} + r^2 \frac{m}{2}(1 - \lambda_n)\right)s},$$

with the nominal slip ratio  $\lambda_n = 0.05$ , a Proportional Integral (PI) speed controller is designed by the placing the pole in  $-1\text{rad/s}$ .

## 8

## 8.6 RESULTS AND DISCUSSION

In this section, the results are presented and discussed in three categories: the relaxed cases with  $v_x = 5\text{m/s}$ , lateral assessment with  $v_x = 6\text{m/s}$ , and the extreme cases where  $v_x = 7\text{m/s}$ .

### 8.6.1 RELAXED CASES WITH $v_x = 5\text{M/S}$

Figure 8.7 shows the results for the first round of experimental tests. The vehicle accelerates to reach the longitudinal velocity of  $5\text{m/s}$  and then detects a static obstacle with  $\tau \in \{3, 4, 5\}$  seconds. The desired vehicle speed is set to  $5\text{m/s}$  as well, and to compensate for the lower  $\tau$  values, we select a smaller obstacle for lower  $\tau$  values to keep the  $\nu$  parameter in a narrow band. In all three scenarios, the vehicle manages to safely avoid colliding with the static obstacle with the help of the MPC and feedforward steering commands while keeping the velocity close to the desired value. In all three cases, the maximum steering command provided by the combined planner occurs after passing the obstacle. At this point, the planner decides to steer back to the initial lateral position since there is no longer a risk of colliding with the obstacle. However, the communication delays in reading the GPS signal result in a slight delay in responding as well, as can be observed in the cases with  $\tau = 4\text{s}$  and  $3\text{s}$ .

**Observation 8.1.** While MPC is prone to providing solutions close to the steering bounds in limited computation time, the combined planning strategy offers a smoother maneuver.

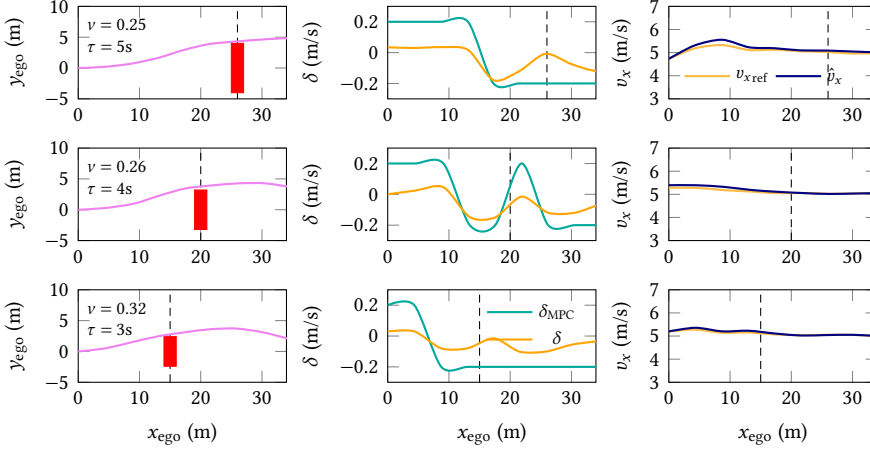


Figure 8.7: Experimental assessment of the  $\tau$  influence: vehicle trajectory for the desired velocity  $v_x = 5\text{m/s}$ .

### 8.6.2 LATERAL ASSESSMENT WITH $v_x = 6\text{M/S}$

Three examples for the next set of experiments are shown in Fig. 8.8. The distance to the static obstacle at the detection time is set to  $\tau = 3\text{s}$  for all three cases, while varying the value of  $\nu$  from 0.4 to 0.35. Similar to the relaxed cases, we observe that MPC is still “overreacting” due to the limited computation time, while the combined planner manages to avoid any collision without an extreme steering command. The desired velocity is set to  $6\text{m/s}$ , which is the same as the longitudinal velocity at the time of obstacle detection in the first two cases. However, in the third case, the vehicle detects the obstacle at the speed of  $4.5\text{m/s}$  which means that a much higher steering command is required to achieve the same lateral displacement. As a result, we observe that the planners opt for an increase in the longitudinal velocity to allow for a lower and safer steering command.

**Observation 8.2.** With a higher velocity, we observe that the GPS measurement lag can be more limiting in a swift return to the initial lateral position after overtaking the obstacle.

### 8.6.3 EXTREME CASES WITH $v_x = 7\text{M/S}$

Finally, we conduct the experiment for the extreme cases close to the maximum allowed longitudinal velocity on campus field with setting the desired vehicle speed to  $7\text{m/s}$ . Figure 8.9 shows the results for three cases with the same obstacle size, while changing the  $\tau$  and  $\nu$  values. In all three cases, we observe that the planners need to increase the longitudinal velocity as the steering limits are not sufficient to guarantee a safe collision-free maneuver.

**Observation 8.3.** Increasing the longitudinal velocity is a potential solution in scenarios where the steering limits are not sufficient to guarantee a safe collision-free maneuver.

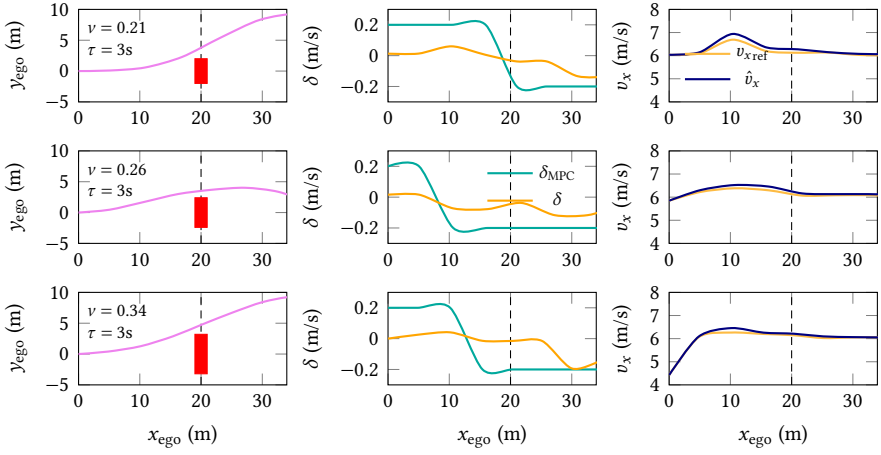


Figure 8.8: Experimental assessment of the  $\nu$  influence: vehicle trajectory for the desired velocity  $v_x = 6\text{m/s}$ .

In the most extreme case with  $\tau = 1.4\text{s}$  and  $\nu = 0.95$ , we observe that the MPC planner cannot find a feasible solution to avoid colliding with the static obstacle. However, the feedforward planner offers a solution close to the limits of steering, which helps in avoiding the highly-probable collision. While returning to the initial lateral position and the desired velocity after overtaking the obstacle.

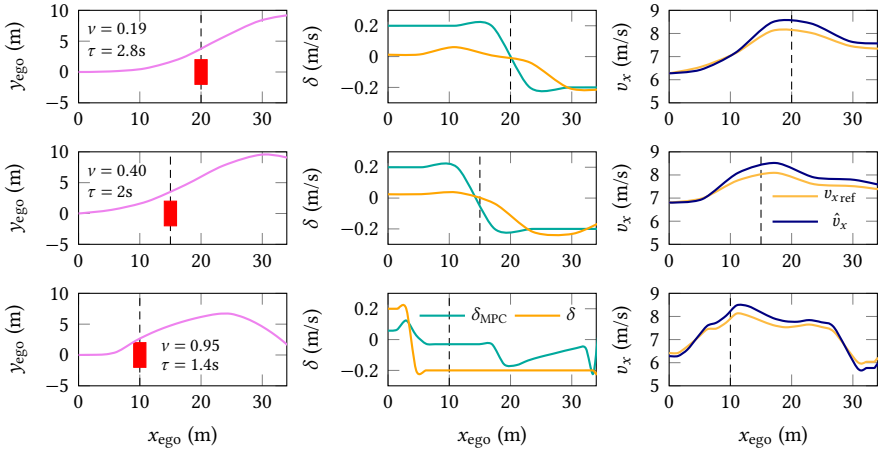


Figure 8.9: Experimental assessment of the hazard: vehicle trajectory for the desired velocity  $v_x = 7\text{m/s}$ .

#### 8.6.4 COMPARISON WITH MPC

In the final set of tests, we compare the collision avoidance capabilities of the MPC planner against our proposed combined strategy. Figure 8.10 shows two scenarios with  $v_x = 6\text{m/s}$



and  $\tau = 3\text{s}$  with two different obstacle widths. It can be observed that while MPC overreacts to the presence of the obstacle by providing extreme steering commands due to the limited computation time, the combined planner can safely avoid colliding with the obstacle while keeping the ego vehicle within the field limits.

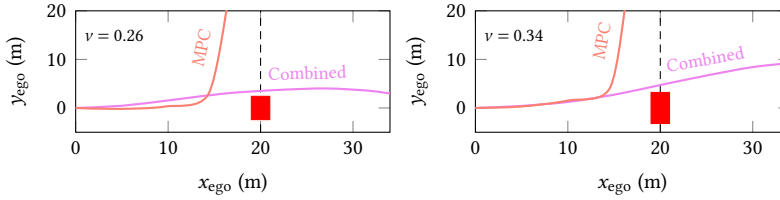


Figure 8.10: Comparing the MPC and combined planners

## 8.7 CONCLUSIONS

This chapter has offered experimental insights into real-time implementation of MPC for collision avoidance after the unexpected appearance of a static obstacle. Given the limitations of state-of-the-art nonlinear MPC in providing feasible solutions in real-time, we have proposed a human-inspired feedforward planner to support situations where the MPC optimization problem is either infeasible or converges to a poor local solution due to a poor initial guess. Our real-world experiments, conducted under various conditions and speeds using the FPEV2-Kanon electric vehicle, validate the effectiveness of our proposed planning strategy, also in comparison to a state-of-the-art MPC motion planner.

For future research, we suggest real-time experimental tests considering parametric uncertainties e.g. due to variations in friction coefficient.



## 9

# PROACTIVE COLLISION AVOIDANCE WITH STOCHASTIC OBSTACLE BEHAVIOR

*Then predictions could be scientific, ... only by ceasing to prophesy definitively.*

— Albert Camus, *The Rebel*

*Uncertainty in the behavior of other traffic participants is a crucial factor in collision avoidance for automated driving; here, stochastic metrics could avoid overly conservative decisions. This chapter introduces a Stochastic Model Predictive Control (SMPC) planner for emergency collision avoidance in highway scenarios to proactively minimize collision risk while ensuring safety through chance constraints. To guarantee that the emergency trajectory can be attained, we incorporate nonlinear tire dynamics in the prediction model of the ego vehicle. Further, we exploit Max-Min-Plus-Scaling (MMPS) approximations of the nonlinearities to avoid conservatism, enforce proactive collision avoidance, and improve computational efficiency in terms of performance and speed. Consequently, our contributions include integrating a dynamic ego vehicle model into the SMPC planner, introducing the MMPS approximation for real-time implementation in emergency scenarios, and integrating SMPC with hybridized chance constraints and risk minimization. We evaluate our SMPC formulation in terms of proactivity and efficiency in various hazardous scenarios. Moreover, we demonstrate the effectiveness of our proposed approach by comparing it with a state-of-the-art SMPC planner and we validate that the generated trajectories can be attained using a high-fidelity vehicle model in IPG CarMaker.*

9

## 9.1 INTRODUCTION

While robust (worst-case) approaches in Model Predictive Control (MPC) synthesis have been used in automated driving to ensure safe motion planning in uncertain dynamic environments [14, 15, 178, 179], they can lead to overly-conservative maneuvers [180] and eventually fail in reaching the main control objective. For instance, it is recognized that human drivers do not drive according to worst-case considerations: if they did, an urban driver may never merge into its desired lane when considering the worst-case scenario in predicting the behavior of other traffic participants [181], or a highway driver would activate unnecessary emergency braking when considering the worst-case scenario in predicting the behavior of a cut-in vehicle. Arguably, the way human drivers avoid overly-conservative maneuvers is by taking some stochastic metrics into account during the planning.

As an example, Fig. 9.1 shows a scenario of proactive collision avoidance: the ego vehicle (pink) is surrounded by other road users (green). If the front vehicle suddenly brakes, a conservative decision would be to decelerate as well to keep the distance. However, this decision could lead to collision with the rear vehicle. It would be much safer in this scenario for the ego vehicle to proactively avoid the collision by moving to the left lane while keeping a safe distance from all the surrounding road users. In summary, proactive collision avoidance can be understood by three key features: swift response to disturbance (i.e. danger), optimality in terms of safety, and avoiding propagation of hazard to future time steps, which translates into getting out of an emergency situation as fast as possible.

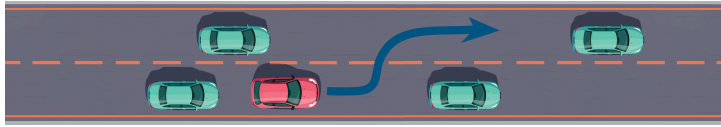


Figure 9.1: Example of proactive collision avoidance in a highway scenario: if its front vehicle suddenly brakes, the ego vehicle (pink) avoids front and rear-end collision with other road users (green) by safely moving to the left lane.

### 9

#### 9.1.1 MOTION PLANNING CHALLENGES IN DIFFERENT SCENARIOS

SMPC [182] is used in various collision-avoidance applications to generate a reference trajectory within a dynamic environment e.g. for mobile robots [183–185] or spacing control in vehicle platoons [186, 187]. In automated driving applications, [159] reviews different threat metrics for risk assessment during maneuvers from collision probability to time-to-collision or distance-to-collision between the ego vehicle and other road users. Since the challenges and requirements of stochastic motion planning in an uncertain environment depend on the driving scenario, two cases can be distinguished: urban or highway.

In urban driving, the vehicles drive at lower speeds, which allows using kinematic models for the ego vehicle [188]. In addition, vehicles can decide among different actions such as turning to different streets at a junction, stopping to park, or merging into another lane [1]. Moreover, there is a variety of traffic participants from pedestrians and bicycles to different drivers with their own driving styles that significantly affect the decision-

making outcome [181]. Therefore, the prediction of other participants should be more comprehensive and intention-aware, and the research in this area has been focusing on robust estimation of feasible space [189, 190] and tractable MPC formulations in the presence of uncertainty in the behavior of other traffic participants [2, 3].

Conversely, the planning problem in highway scenarios faces two entangled challenges: ensuring that the generated trajectory can be attained and solving the planning problem is computationally efficient. On highways, an emergency maneuver at high speed would push the vehicle in the nonlinear regime. In this sense, ensuring that the generated reference trajectory can be attained requires considering the nonlinear tire dynamics within the ego vehicle model for a more accurate prediction of the available tire forces [13]. A common solution to avoid unattainable generated trajectories is to design an integrated planner/tracker incorporating a higher-fidelity prediction model of the ego vehicle, e.g. [9] proposes serially-cascaded models to allow using different sampling times and prediction horizons for the planning and tracking sub-problems. However, this technique is applicable to less-aggressive maneuvers only, since both prediction models for the planning and tracking sub-problems are simple. In this sense, hierarchical control design is still the most popular choice in the literature for emergency collision avoidance in highway driving [191, 192] and the kinematic single-track model is often selected as the ego vehicle prediction model [119, 192, 193]. On the other hand, incorporating nonlinear tire dynamics significantly increases the computational complexity of the MPC planning problem, which may prevent a proactive response to danger.

### 9.1.2 SOURCES OF UNCERTAINTY IN HIGHWAY DRIVING

In the highway collision avoidance literature, the stochasticity of the uncertain environment is expressed via chance constraints in the SMPC planning problem. After observing their initial position and velocity, the behavior of the obstacles is forecasted over a prediction horizon by considering a linear, often point mass, model [119]. Stochastic behavior of obstacles is then modeled by random variables in their prediction model such as their velocity [194, 195] or acceleration [183]. Sometimes, randomness in the lane change decision is considered as well [193]. In this sense, [196] expresses the trajectory of obstacles using a Markov jump system description, whereas [194] uses a hybrid obstacle model including stochastic switching decision between continuing along a straight path or following an arc trajectory. This uncertainty is then propagated over the prediction horizon e.g. by chaos-based approaches [171] or state updates via Kalman filter [119, 183], leading to chance constraints in the SMPC problem. The reference trajectory is found by minimizing a cost function which in the literature has been mainly defined as a convex (often quadratic) function of the states and inputs [197], such as the velocity-tracking error [119, 196] to enforce maintaining a constant longitudinal velocity. Unfortunately, in emergency maneuvers and hazardous scenarios, minimizing the probability of collision is more important than tracking errors. In this sense, [198] uses a potential field function for collision avoidance, but the obstacle behavior is not stochastic. However, the objective function of avoiding collision may have no closed form, such as in [194], due to the stochasticity of the switching decision. There, the objective function is constructed iteratively via reachable sets.

### 9.1.3 COMPUTATIONAL EFFICIENCY IN EMERGENCIES

At the same time, tractability is also crucial and must be traded with the accuracy of the model. For instance, in [19] a hybrid nonlinear prediction model is considered for the ego vehicle and the exponential growth in computational complexity is compensated by adapting the prediction horizon accordingly. Further, [199] suggests successive convexification to improve the initial guess for the Nonlinear Model Predictive Control (NMPC) problem to reduce the number of iterations and [200] uses a Mixed-Integer Linear Program (MILP) to find the feasible region and feed it into the nonlinear planning problem to find the optimal trajectory. To the best of our knowledge, no research has been done incorporating tire force dynamics for real-time emergency motion planning in highway scenarios, i.e. fast online solution of the planning optimization problem, while minimizing the probability of collision which leads to a highly- nonlinear formulation for the SMPC problem.

Hybrid modeling frameworks [18] such as MMPS formalism [24], are effective tools to reduce the computational complexity of the planning problem while incorporating the nonlinear behavior regime. In this sense, hybridization refers to the approximation of a nonlinear function, e.g. the prediction model, using a hybrid systems modeling framework. In case of a nonlinear control optimization problem, hybridization can lead to an MILP formulation of the problem that is computationally more efficient to solve, compared to a NonLinear Program (NLP). Sequential Quadratic Programming (SQP) and real-time iteration scheme have been used in the literature where the nonlinear dynamics is linearized at each time step [93]. However, that approach has limited capability to adequately capture the complexity of the nonlinear behavior along the prediction horizon. The fact that MILPs can be solved to global optimality in a finite number of iterations [201] makes them a suitable candidate to formulate the MPC planning problem.

### 9.1.4 CONTRIBUTIONS

In this chapter, we propose an SMPC motion planner for emergency collision avoidance in highway scenarios. We present a proactive planner design by minimizing the collision risk as well as improving safety using chance constraints in the SMPC formulation. To avoid generating unattainable trajectories, we incorporate nonlinear tire dynamics (accounting for the nonlinear tire behavior close to saturation limits) within the prediction model for the ego vehicle and we use MMPS approximation to reduce the computational complexity of the planning problem. As a result, the novelties in our work are twofold:

1. introducing the idea of MMPS approximation of the nonlinearities for real-time implementation, and
2. combining hybridized risk minimization within a stochastic MPC framework for highway path planning.

Moreover, we provide a comprehensive analysis of how various formulations of the MPC planner influence the conservatism and efficiency of the algorithm to proactively avoid a collision in hazardous scenarios and we compare our proposed approach to a method inspired by the state-of-the-art SMPC planner in [119] during various cases studies. To verify that the generated trajectories can be attained by our proposed SMPC planner, we simulate the maneuvers using a high-fidelity vehicle model in IPG CarMaker [202].

The chapter is structured as follows: Section 9.2 describes the formulation of the predictive planning problem. Then Section 9.3 briefly covers the MMPS approximation, and Section 9.4 explains our approach in reformulating and solving the SMPC problem. Simulation results and comparisons to the state-of-the-art SMPC planner and the built-in collision avoidance module in IPG CarMaker are presented in Section 9.5. Finally, we conclude this chapter in Section 9.6.

The notation in this work is rather standard. The state and input vectors at time step  $k$  are represented by  $s(k)$  and  $u(k)$ , respectively. We use a tilde symbol, e.g. as in  $\tilde{s}(k)$ , to denote the trajectory of a signal along the prediction horizon. The probability is expressed by the  $\Pr$  symbol.

## 9.2 PROBLEM FORMULATION

Given a predicted state trajectory  $\tilde{s}$  at control time step  $k$  along the next  $N_p$  steps as

$$\tilde{s}(k) = [s^T(k+1|k) \quad \dots \quad s^T(k+N_p|k)]^T, \quad (9.1)$$

the SMPC planning optimization problem can be formulated by the generic form

$$\min_{\tilde{s}(k), \tilde{u}(k)} J(\tilde{s}(k)), \quad (9.2a)$$

$$\text{s.t. } s(k+i|k) = f(s(k+i-1|k), u(k+i-1)), \quad \forall i \in \{1, \dots, N_p\}, \quad (9.2b)$$

$$g(s(k+i-1|k), u(k+i-1)) \leq 0, \quad \forall i \in \{1, \dots, N_p\}, \quad (9.2c)$$

$$\Pr(s(k+i|k) \in S_k) \geq 1 - \epsilon, \quad \forall i \in \{1, \dots, N_p\}, \quad (9.2d)$$

where  $J$  represents the cost function, usually formulated as deviations from a desired velocity or divergence from a globally-planned reference trajectory. Further, the planning problem is constrained to the prediction model of the ego vehicle  $f(\cdot)$  via (9.2b), general nonlinear constraints  $g(\cdot)$  (9.2c), and the chance constraints (9.2d) where  $S_k$  is the safe or confidence region in step  $k$  and  $\epsilon$  is the minimum acceptable probability for constraint violation and is selected to be close to 0. Based on the requirements for highway emergencies,  $J$ ,  $f$ ,  $g$  and  $S_k$ , often need to be selected in such a way that (9.2) would be an NLP, hence computationally expensive to solve in real time. As explained in Section 9.1, we use MMPS approximation of the nonlinearities to facilitate obtaining an MILP reformulation of (9.2) and to improve the computational efficiency. This is further discussed in the next section.

## 9.3 MMPS APPROXIMATION

As the name suggests, MMPS systems are modeled using max, min, plus, and scaling operators and are equivalent to continuous piecewise-affine systems [18]. Any MMPS function  $f_{\text{MMPS}}$  can be described by either a conjunctive or a disjunctive canonical form [28]:

$$f_{\text{con}}(\chi) = \min_{p=1, \dots, P} \max_{q=1, \dots, m_p} (\gamma_{p,q}^T \chi + \nu_{p,q}), \quad (9.3a)$$

$$f_{\text{dis}}(\chi) = \max_{q=1, \dots, Q} \min_{p=1, \dots, n_q} (\phi_{p,q}^T \chi + \omega_{p,q}), \quad (9.3b)$$

where  $\gamma$  and  $\phi$  are vectors,  $v$  and  $\omega$  are scalars, and  $P, Q, m_p$ , and  $n_q$  are integers determining the number of nested min and max operators.

A nonlinear (scalar) function  $f : \mathcal{D} \rightarrow \mathbb{R}$  can be approximated by an MMPS form  $[f]_{\text{MMPS}}$  in compact state domain  $\mathcal{D}$  via solving the nonlinear optimization problem

$$\min_{\mathcal{A}} \int_{\mathcal{D}} \frac{\|f(\chi) - [f]_{\text{MMPS}}(\chi)\|_2}{\|f(\chi)\|_2 + \epsilon_0} d\chi, \quad (9.4)$$

where  $[.]_{\text{MMPS}}$  represents the MMPS approximation of the corresponding argument with either forms in (9.3) and  $\mathcal{A}$  collects the decision variables for fixed values of  $P, Q, m_p$ , and  $n_q$  as

$$\mathcal{A} = \begin{cases} (\{\gamma_{p,q}\}, \{v_{p,q}\})_{\substack{p=1,\dots,P \\ q=1,\dots,m_p}} & \text{if } [f]_{\text{MMPS}} = f_{\text{con}} \\ (\{\phi_{p,q}\}, \{\omega_{p,q}\})_{\substack{q=1,\dots,Q \\ p=1,\dots,n_q}} & \text{if } [f]_{\text{MMPS}} = f_{\text{dis}} \end{cases}. \quad (9.5)$$

Note that  $\mathcal{A}$  is a tuple of vector and scalar sets since it is necessary to preserve their order in the MMPS forms. The positive value  $\epsilon_0 > 0$  added to the denominator in (9.4) serves to avoid division by very small values for  $\|f(\chi)\|_2 \approx 0$ .

In the next steps, we hybridize a suitable nonlinear prediction model for the ego vehicle by solving (9.4) for the nonlinear terms within the vehicle model and use our information of the shape and form of each nonlinearities to select their respective approximation forms in (9.3) and the values of the integer pairs  $(P, m_p)$  or  $(Q, n_q)$ . Problem (9.4) is a smooth NLP which can be solved by e.g. sequential quadratic programming and multi-start strategy.

## 9.4 PROBLEM REFORMULATION AND SOLUTION APPROACH

### 9.4.1 OBSTACLE VEHICLE MODEL

Given  $N_o$  obstacles on the road, the states of the  $\eta$ -th obstacle where  $\eta \in \{1, \dots, N_o\}$  at time step  $k$  are expressed by the stochastic vector  $z^{(\eta)}(k)$  defined as

$$z^{(\eta)}(k) = [x_{\text{obs}}^{(\eta)}(k) \quad y_{\text{obs}}^{(\eta)}(k) \quad \dot{x}_{\text{obs}}^{(\eta)}(k) \quad \dot{y}_{\text{obs}}^{(\eta)}(k)]^T, \quad (9.6)$$

with the Gaussian distribution

$$z^{(\eta)}(k) \sim \mathcal{N}(\xi^{(\eta)}(k), \Xi^{(\eta)}(k)), \quad (9.7)$$

where  $\xi$  and  $\Xi$  respectively indicate the mean vector and the covariance matrix as

$$\xi^{(\eta)}(k) = \begin{bmatrix} \xi_x^{(\eta)}(k) & \xi_y^{(\eta)}(k) & \xi_{\dot{x}}^{(\eta)}(k) & \xi_{\dot{y}}^{(\eta)}(k) \end{bmatrix}^T, \quad (9.8)$$

$$\Xi^{(\eta)}(k) = \begin{bmatrix} \sigma_x^{(\eta)}(k) & 0 & 0 & 0 \\ 0 & \sigma_y^{(\eta)}(k) & 0 & 0 \\ 0 & 0 & \sigma_{\dot{x}}^{(\eta)}(k) & 0 \\ 0 & 0 & 0 & \sigma_{\dot{y}}^{(\eta)}(k) \end{bmatrix}. \quad (9.9)$$



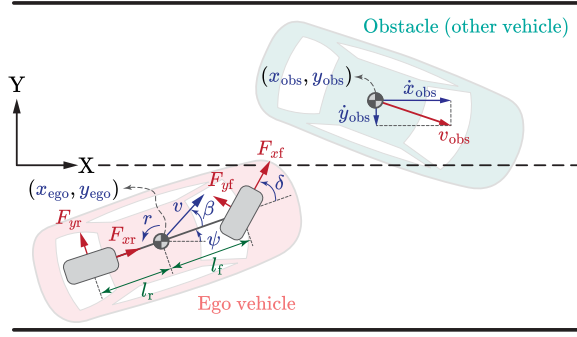


Figure 9.2: Model configuration for the ego vehicle and the obstacles on the road.

**Remark 9.1.** We use discretized double integrator dynamics to model the obstacle behavior and update variance and mean using Kalman updates. Note that the actual covariance matrix does not remain diagonal, but it is customary to consider a reduced or approximated covariance matrix including the diagonal elements of  $\Xi$  associated with the target states [203–206] for computational efficiency; an approach we use in this chapter as well.

More specifically, we use a point mass model [119] for the obstacles in Fig. 9.2, expressed by

$$z^{(\eta)}(k+1) = Az^{(\eta)}(k) + Bw^{(\eta)}(k) + v^{(\eta)}(k), \quad (9.10a)$$

$$w^{(\eta)}(k) = K \left( z_{\text{ref}}^{(\eta)}(k) - z^{(\eta)}(k) \right), \quad (9.10b)$$

where  $w$  represents the input signal as

$$w^{(\eta)}(k) = \begin{bmatrix} \ddot{x}_{\text{obs}}^{(\eta)}(k) & \ddot{y}_{\text{obs}}^{(\eta)}(k) \end{bmatrix}^T, \quad (9.11)$$

the  $A$  and  $B$  being state and input matrices resulting from discretized double integrator dynamics,  $v \sim \mathcal{N}(0_{4 \times 1}, \Xi_0)$  the process noise, and  $K$  being the Kalman feedback gain such that the obstacle tracks its corresponding reference state  $z_{\text{ref}}$ . The covariance matrix for each obstacle is updated at each time step in line with Kalman update by

$$\xi^{(\eta)}(k+1) = (A - BK)\xi^{(\eta)}(k)BKz_{\text{ref}}^{(\eta)}(k), \quad (9.12a)$$

$$\Xi^{(\eta)}(k+1) = (A - BK)\Xi^{(\eta)}(k)(A - BK)^T + \Xi_0^{(\eta)}. \quad (9.12b)$$

with  $\Xi_0$  being the initial estimate of the covariance matrix of the process noise. Using the Gaussian distribution in (9.7), we define  $p_k^{(\eta)}$  to express the probability density function for the presence of obstacle  $\eta \in \{1, \dots, N_o\}$  on the road as

$$p_k^{(\eta)}(x, y) = \frac{\exp \left( - \left( \frac{x - \xi_x^{(\eta)}(k)}{\sqrt{2}\sigma_x^{(\eta)}(k)} \right)^2 - \left( \frac{y - \xi_y^{(\eta)}(k)}{\sqrt{2}\sigma_y^{(\eta)}(k)} \right)^2 \right)}{2\pi\sigma_x^{(\eta)}(k)\sigma_y^{(\eta)}(k)}, \quad (9.13)$$

which is used to develop the probability function  $P$  for the state vector  $s(k)$  defined in (16) using a chi-squared distribution (see [119]) and taking into account the unsafe area  $\Omega^{(\eta)}$ , as

$$P^{(\eta)}(s(k)) = \Pr((x_{\text{ego}}(k), y_{\text{ego}}(k)) \in \Omega^{(\eta)}). \quad (9.14)$$

The unsafe set  $\Omega$  for each obstacle is defined as an area that the center of gravity of the ego vehicle must avoid, and it is an ellipse calculated by considering the position and size of both ego and obstacle vehicles as known parameters [159].

### 9.4.2 HYBRID EGO VEHICLE MODEL

The ego vehicle prediction model as shown in Fig. 9.2 is described by a dynamic bicycle model [12] with a small-angle assumption for  $\delta$  (reasonable in highway scenarios [15])

$$\dot{x}_{\text{ego}} = v \cos(\psi + \beta), \quad (9.15a)$$

$$\dot{y}_{\text{ego}} = v \sin(\psi + \beta), \quad (9.15b)$$

$$\dot{\psi} = r, \quad (9.15c)$$

$$\dot{v} = \frac{1}{m} [F_{\text{xf}} - F_{\text{yf}}\delta + F_{\text{xr}}] + v\beta r, \quad (9.15d)$$

$$\dot{\beta} = \frac{1}{mv} [F_{\text{yf}} + F_{\text{yr}}] - r, \quad (9.15e)$$

$$\dot{r} = \frac{1}{I_{zz}} [F_{\text{xf}}\delta l_f + F_{\text{yf}} l_f - F_{\text{yr}} l_r], \quad (9.15f)$$

$$\dot{\delta} = d_\delta, \quad (9.15g)$$

with  $F_{\text{xf}}$ ,  $F_{\text{xr}}$ , and  $d_\delta$  as inputs. All the variables and system parameters are described in Tables 9.1 and 9.2, and the state vector  $s$  at time step  $k$  is expressed by

$$s(k) = [x_{\text{ego}}(k) \quad y_{\text{ego}}(k) \quad \psi(k) \quad v(k) \quad \beta(k) \quad r(k) \quad \delta(k)]^T. \quad (9.16)$$

The tire forces should satisfy the tire saturation limits

$$F_{\text{xf}}^2 + F_{\text{yf}}^2 \leq (\mu F_{\text{zf}})^2, \quad (9.17a)$$

$$F_{\text{xr}}^2 + F_{\text{yr}}^2 \leq (\mu F_{\text{zr}})^2, \quad (9.17b)$$

also known as Kamm circle constraint [89]. Considering the slip angles

$$\alpha_f = \delta - \beta + \frac{l_f r}{v}, \quad (9.18a)$$

$$\alpha_r = \frac{l_r r}{v} - \beta, \quad (9.18b)$$

we describe the lateral tire forces by MMPS approximations of the Pacejka tire model [90] shown in Fig. 9.3 as

$$[F_y]_{\text{MMPS}} = F_{\text{max}} \min \left( \max \left( \frac{\alpha}{\alpha_s}, -1 \right), 1 \right), \quad (9.19)$$

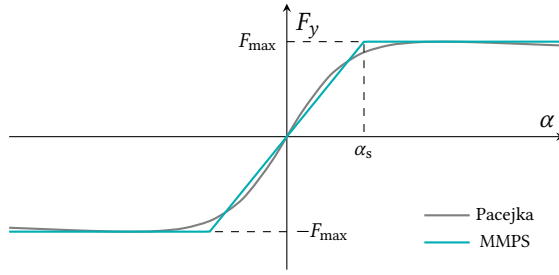


Figure 9.3: Pacejka tire model and its MMPS approximation

where the nonlinear function representing the tire forces on the front and rear axles is approximated by a parametric MMPS function where  $F_{\max}$  and  $\alpha_s$  respectively correspond to the maximum tire force and the saturation slip angle.

Substituting the front and rear slip angles in (9.19) gives the front and rear lateral tire forces as

$$[F_{yf}]_{\text{MMPS}} = F_{\max} \min \left( \max \left( \frac{\delta}{\alpha_s} - \frac{\beta}{\alpha_s} + \frac{l_f r}{\alpha_s v_0}, -1 \right), 1 \right), \quad (9.20a)$$

$$[F_{yr}]_{\text{MMPS}} = F_{\max} \min \left( \max \left( \frac{l_r}{\alpha_s v_0} r - \frac{1}{\alpha_s} \beta, -1 \right), 1 \right). \quad (9.20b)$$

Using the MMPS approximation of the other nonlinear terms in the ego vehicle model, we obtain an MMPS formulation for the ego vehicle model expressed by

$$\dot{x}_{\text{ego}} = \max\{v, v_0 [\cos(\psi + \beta)]_{\text{MMPS}}\}, \quad (9.21a)$$

$$\dot{y}_{\text{ego}} = v_0 [\sin(\psi + \beta)]_{\text{MMPS}}, \quad (9.21b)$$

$$\dot{\psi} = r, \quad (9.21c)$$

$$\dot{v} = \frac{F_{xf} + F_{xr}}{m} - \frac{[\delta F_{yf}]_{\text{MMPS}}}{m} + v_0 [\beta r]_{\text{MMPS}}, \quad (9.21d)$$

$$\dot{\beta} = \frac{[F_{yf} + F_{yr}]_{\text{MMPS}}}{mv_0} - r, \quad (9.21e)$$

$$\dot{r} = \frac{l_f \delta_0 F_{xf}}{I_{zz}} + \frac{l_f [F_{yf}]_{\text{MMPS}}}{I_{zz}} - \frac{l_r [F_{yr}]_{\text{MMPS}}}{I_{zz}}, \quad (9.21f)$$

$$\dot{\delta} = d_{\delta}, \quad (9.21g)$$

Figure 9.4 presents three examples of the nonlinear terms vs. their MMPS approximations. To find these formulations, we have used information on the form of the nonlinear function and we have selected the number of max and min operators accordingly. For instance, in Fig. 9.4a, we use three hyperplanes and two max and min operators based on the cosinusoidal shape of the nonlinear function.

**Remark 9.2.** Considering the orders of magnitude of variations of the longitudinal velocity over the prediction horizon, the velocity  $v$  in (9.15b), (9.15d) and (9.15e) can be approximated as

a fixed parameter over the prediction horizon and can be taken equal to the current measured velocity. Moreover, in cases where  $v$  is multiplied by cosine terms with values close to 1, we take the maximum value between the velocity  $v$  and the MMPS approximation with  $v = v_0$  in (9.21a) to ensure the inclusion of numerically significant effects resulting from variations in  $v$  when  $\psi + \beta \approx 0$  in (9.21a). A similar approach is used for  $\delta$  in (9.15f) where its variations are included in the MMPS tire forces and the current steering angle is used as a parametric coefficient for the first term.

**Remark 9.3.** After MMPS approximation of the continuous-time model of the ego vehicle, (9.21) can be discretized e.g. using forward Euler method and a proper sampling time to be incorporated in the SMPC formulation in (9.2).

Further, the Kamm circle constraints in (9.17) are approximated using MMPS function in Figures 9.4c. Note that due to different ranges of  $F_{xf}$  and  $F_{xr}$ , the front and rear force magnitudes are approximated by the maximum of respectively three and four affine functions, to appropriately capture the form of the nonlinear function. The maximum tire forces on the front and rear axles are functions of the online measurements of the friction coefficient  $\mu$ , which we assume available via a friction estimator [9, 12], as

$$F_{\max} = \min(\mu F_{zf}, \mu F_{zr}).$$

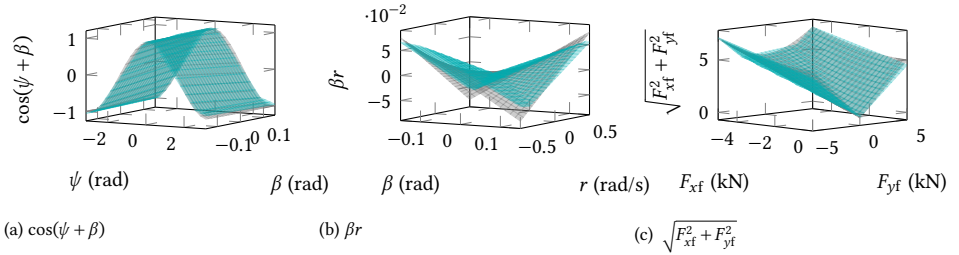


Figure 9.4: Plots of example nonlinear terms in the ego vehicle prediction model and their MMPS approximations

### 9.4.3 CHANCE CONSTRAINTS AND COLLISION RISK FUNCTION

To hybridize the probability function  $P$  in (9.14), we approximate it by the MMPS function  $[P]_{\text{MMPS}}$  as illustrated in Fig. 9.5. The MMPS approximation  $[P]_{\text{MMPS}}$  is a probability function as well and is used as a chance constraint in the SMPC formulation.

Since the chance constraints must be bounded in such a way that the probability of constraint violation is very low (to improve safety), the accuracy of the MMPS approximation is more important in regions close to  $P = 0$ . Therefore, we obtain  $[P]_{\text{MMPS}}$  by approximating the Gaussian probability density function (9.13) on a compact domain  $D$  defined by the road boundaries via solving (9.4) and imposing the constraint

$$\int_D [p_k]_{\text{MMPS}}^{(\eta)}(x, y) = 1,$$

Table 9.1: System variables and their bounds in the case study

Variable	Definition	Unit	Bounds
$x_{ego}$	Longitudinal position of the ego vehicle	m	$[0, \infty]$
$x_{obs}$	Longitudinal position of the obstacle	m	$[0, \infty]$
$\dot{x}_{obs}$	Longitudinal velocity of the obstacle	m/s	$[5, 50]$
$y_{ego}$	Lateral position of the ego vehicle	m	$[-6, 6]$
$y_{obs}$	Lateral position of the obstacle	m	$[-6, 6]$
$\dot{y}_{obs}$	Lateral velocity of the obstacle	m/s	$[-5, 5]$
$v$	Velocity of the ego vehicle	m/s	$[5, 50]$
$\beta$	Sideslip angle	rad	$[-0.2, 0.2]$
$\psi$	Yaw angle	rad	$[-\pi, \pi]$
$r$	Yaw rate	rad/s	$[-0.5, 0.5]$
$\delta$	Steering angle (road)	rad	$[-0.2, 0.2]$
$F_{xf}$	Longitudinal force on the front axis	N	$[-5000, 0]$
$F_{xr}$	Longitudinal force on the rear axis	N	$[-5000, 5000]$
$F_{yf}$	Lateral force on the front axis	N	–
$F_{yr}$	Lateral force on the rear axis	N	–
$F_{zf}$	Normal load on the front axis	N	–
$F_{zr}$	Normal load on the rear axis	N	–
$\alpha_f$	Front slip angle	rad	–
$\alpha_r$	Rear slip angle	rad	–

Table 9.2: System parameters

Parameter	Definition	Value	Unit
<i>Fixed parameters</i>			
<i>(IPG CarMaker BMW vehicle model)</i>			
$m$	Vehicle mass	1970	kg
$I_{zz}$	Inertia moment about z-axis	3498	kg/m <sup>2</sup>
$l_f$	CoG* to front axis distance	1.4778	m
$l_r$	CoG to rear axis distance	1.4102	m
$F_{zf}$	Normal load on the front axis	7926	N
$F_{zr}$	Normal load on the rear axis	8303	N
$\alpha_s$	Saturation slip angle	0.09	rad
$t_s$	Planner sampling time	0.2	s
$N_p$	Prediction horizon	10	–
<i>Varying parameters</i>			
<i>(measured online)</i>			
$F_{max}$	Maximum tire force	–	N
$\mu$	Friction coefficient	–	–
$s_0$	Initial EV state vector	–	–
$x_{ref}$	Globally-planned reference trajectory	–	–
$v_0$	Initial velocity	–	m/s
$\delta_0$	Initial steering angle	–	rad

\*Center of Gravity

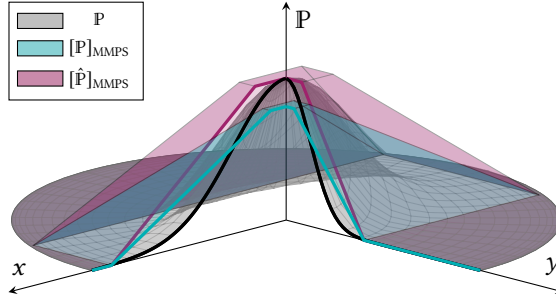


Figure 9.5: Conceptual illustration of the Gaussian probability function  $P$ , of its MMPS approximation and of the MMPS proxy functions. The approximations are valid in the compact domain  $\mathcal{D}$ .

which gives the parametric form for  $[P]_{\text{MMPS}}$  from (9.14) as

$$[P]_{\text{MMPS}}(s(k)) = \max \left( \min_{p=1,\dots,5} (\phi_p^T s(k) + \omega_p), 0 \right), \quad (9.22)$$

with  $\phi_p$  being affine functions of  $\xi_x(k)$ ,  $\xi_y(k)$ ,  $\sigma_x(k)$  and  $\sigma_y(k)$ . Similar to  $P$ , the MMPS approximation  $[P]_{\text{MMPS}}$  is a probability function that is used in the chance constraints.

However,  $[P]_{\text{MMPS}}$  under-estimates  $P$  in regions close to the peak of  $P$ , which is not desired for deriving the collision risk function. To improve safety, we use the MMPS function  $[\hat{P}]_{\text{MMPS}}$  in Fig. 9.5 as a proxy of  $[P]_{\text{MMPS}}$  to obtain the risk of collision for each point on the road in the presence of other road users. This time, we find  $\hat{\phi}_p$  by approximating  $p_k$  via (9.4) constrained to

$$[\hat{p}_k]_{\text{MMPS}}^{(\eta)}(x, y) \geq p_k(x, y), \quad \forall (x, y) \in \mathcal{D},$$

which gives the proxy function

$$[\hat{P}]_{\text{MMPS}}(s(k)) = \max \left( \min_{p=1,\dots,5} (\hat{\phi}_p^T s(k) + \hat{\omega}_p), 0 \right), \quad (9.23)$$

serving as an over-estimation of  $P$  based on  $[\hat{p}_k]_{\text{MMPS}}$ . Since  $[\hat{p}_k]_{\text{MMPS}}$  is not a probability density function,  $[\hat{P}]_{\text{MMPS}}$  is only used to calculate the risk as the cost and does not serve as an approximation of the probability in evaluating the chance constraint. This separation allows to avoid conservatism in  $[P]_{\text{MMPS}}$  within the constraints while seeking safer trajectories by minimizing the over-approximation  $[\hat{P}]_{\text{MMPS}}$ .

For each time step, the collision risk depends on the probability of the presence of other road users in  $(x, y)$ . Therefore, the collision risk of  $\tilde{s}(k)$  can be defined as

$$P(\tilde{s}(k)) = \frac{1}{N_p} \sum_{i=1}^{N_p} \max_{\eta=\{1,\dots,N_o\}} [\hat{P}]_{\text{MMPS}}^{(\eta)}(s(k+i|k)). \quad (9.24)$$

**Remark 9.4.** The max operator in (9.24) can be replaced by a sum across the presence probability of all the  $N_o$  road users. However, this sum may result in a more conservative

estimation of the collision risk (The same argument can be deduced using Boole's inequality). For instance, if there are two obstacles with a safe corridor in between where  $\mathbb{P}^{(1)} = \mathbb{P}^{(2)} = \rho$ , the sum would give a risk of approximately  $2\rho$  for this area, whereas in a real situation, the chance of two vehicles getting closer is low; furthermore, the real presence probability for both obstacles would be even lower than  $\rho$  which is an estimate that does not take into account the effect of the presence of one obstacle on the decisions of other road users.

#### 9.4.4 SMPC OPTIMIZATION PROBLEM

We incorporate the presence probability of obstacles into the MPC planner in two ways: first, we ensure a very low probability for the collision by constraining  $[\mathbb{P}]_{\text{MMPS}}^{(\eta)}$  to be less than a small threshold  $\epsilon > 0$ . Secondly, we minimize the collision risk function  $P$  from (9.24) in the objective function to not only ensure this safety level, but also to converge to the safest attainable trajectory and to prevent getting close to high-risk areas in a predictive manner. This in fact will lead to a more proactive response to danger during a hazardous scenario, which will be illustrated in an example case later.

The stochastic MPC motion planner is formulated as follows: given a globally-planned reference velocity profile  $\tilde{v}_{\text{ref}}$  and the initial states  $s_0$ , we find the optimal trajectory  $\tilde{s}_{\text{ref}}$  by solving

$$\min_{\tilde{s}, \tilde{u}} \quad P(\tilde{s}) + \lambda_v \|\tilde{v} - \tilde{v}_{\text{ref}}\|_1 + \lambda_u \|\tilde{u}\|_1 + \lambda_\tau \|\tilde{\tau}\|_1, \quad (9.25a)$$

$$\text{s.t.} \quad s(k+i|k) = [f_{\text{ego}}]_{\text{MMPS}}(s(k+i-1|k), u(k+i-1)), \quad \forall i \in \{1, \dots, N_p\}, \quad (9.25b)$$

$$[g]_{\text{MMPS}}(s(k+i-1|k), u(k+i-1)) \leq 0, \quad \forall i \in \{1, \dots, N_p\}, \quad (9.25c)$$

$$\tau(i) = \min_{j=1, \dots, N_{\text{lane}}} \{ |y_{\text{ego}}(k+i|k) - y_{c_j}| \}, \quad \forall i \in \{1, \dots, N_p\}, \quad (9.25d)$$

$$[\mathbb{P}]_{\text{MMPS}}^{(\eta)}(s(k+i|k)) \leq \epsilon, \quad \forall \eta \in \{1, \dots, N_o\}, \quad \forall i \in \{1, \dots, N_p\}, \quad (9.25e)$$

where  $[f_{\text{ego}}]_{\text{MMPS}}$  represents the discretized form of the MMPS system dynamics in (9.21) and similarly,  $[g]_{\text{MMPS}}$  approximates the nonlinear constraints such as the Kamm circle. The objective is to minimize the cost in (9.25a) which consists of the collision probability, the deviation from the reference velocity, and the control effort. Moreover, the lane-center deviation  $\tau$  is defined over the prediction horizon as (9.25d) which allows switching to a “better” lane (among  $N_{\text{lane}}$  lanes) if necessary. Here,  $y_{c_j}$  values represent the center line in lanes 1 and 2 for as two available lanes for the vehicle on the road and can be easily extended to include more lanes. Constraints (9.25b) and (9.25e) respectively account for the prediction model of the ego vehicle and the chance constraints. The Proactive SMPC (P-SMPC) problem is solved via Algorithms 7 and 8.

**Algorithm 7** Probability function development

---

**Require:**  $Z(k), \Xi_0, N_p$   $\triangleright Z$  contains states of all the obstacles  
**for**  $\eta \in \{1, \dots, N_o\}$  **do**  
 $z^{(\eta)}(k) \leftarrow \eta^{\text{th}}$  column in  $Z(k)$   
 $\Xi^{(\eta)}(k|k) \leftarrow \Xi_0$   
**for**  $i \in \{1, \dots, N_p\}$  **do**  $\triangleright$  obstacle prediction  
 $\Xi^{(\eta)}(k+i|k) \xleftarrow{(9.12)} \Xi_0, \Xi^{(\eta)}(k+i-1|k)$   
 $\xi^{(\eta)}(k+i|k) \xleftarrow{(9.10)} z^{(\eta)}(k+i-1|k)$   
 $p_i^{(\eta)} \xleftarrow{(9.13)} \xi^{(\eta)}(k+i|k), \Xi^{(\eta)}(k+i|k)$   
 $P^{(\eta)}(.) \xleftarrow{\text{develop using (9.14)}} p_i^{(\eta)}$   
 $[P]_{\text{MMPS}}^{(\eta)}(.) \xleftarrow{(9.22)} P^{(\eta)}(.)$   
 $[\hat{P}]_{\text{MMPS}}^{(\eta)}(.) \xleftarrow{(9.23)} P^{(\eta)}(.)$   
**end for**  
**end for**  
**return**  $[P]_{\text{MMPS}}^{(\eta)}(.), [\hat{P}]_{\text{MMPS}}^{(\eta)}(.) \quad \forall \eta \in \{1, \dots, N_o\}$

---

**Algorithm 8** The P-SMPC planner

---

**Require:**  $s(k), [f_{\text{ego}}]_{\text{MMPS}}, Z(k), \Xi_0, \tilde{v}_{\text{ref}}, N_p, y_c$   
 $N_{\text{lane}} \leftarrow \text{length of } y_c$   
 $[P]_{\text{MMPS}}(.), [\hat{P}]_{\text{MMPS}}(.) \xleftarrow{\text{Algorithm 7}} Z(k), \Xi_0, N_p$   
 $P(.) \xleftarrow{\text{develop using (9.24)}} [\hat{P}]_{\text{MMPS}}(.), N_p$   
 $\tilde{s}^* \leftarrow \text{solve (9.25)}$   $\triangleright$  the planning optimization problem  
**return**  $\tilde{s}^*$

---

**Remark 9.5.** The chance constraints in the SMPC literature [182] are often expressed by the generic form in (9.2d). In our planner formulation, we use (9.25e) as a more tractable formulation of chance constraints, which is essentially equivalent to bounding the constraint violation probability in (9.14) or its MMPS approximation (9.22) by a small value  $\epsilon$ . Note that  $[P]_{\text{MMPS}}$  over-estimates  $P$  for probabilities close to zero as shown in Fig. 9.5, and that in (9.25e) we make sure the collision probability is smaller than  $\epsilon$  for all the states in  $\mathcal{D}$  and all the time steps within the prediction horizon.

## 9.5 SIMULATIONS AND RESULTS

In this section, we evaluate the control performance of our proposed P-SMPC planner on two aspects: proactivity of the planner, and attainable generated trajectories. Here, we select  $\epsilon = 0.001$  which is the tightest bound investigated in [119]. The P-SMPC optimization problem defines 10 continuous and 20 binary decision variables per prediction step to model the ego vehicle. Further, each detected obstacle adds up to 6 binary variables per



prediction step to allow for hybrid representation of the collision probability function associated with it.

The proactivity assessment is done in four highway scenarios where we investigate the effect of collision-risk minimization in the objective function (9.25a) in our P-SMPC planner against the optimization formulation inspired by the state-of-the-art [119] indicated as Regular SMPC (R-SMPC) planner where the collision-risk is not included in the objective function and the collision is avoided by only considering the left-hand side of (9.25e). Note that R-SMPC is not the same planner as in [119] since it incorporates the MMPS approximation of the nonlinearities, but we only change the objective function while keeping the same dynamic prediction model for both planners for a fair comparison and a better analysis of the risk-minimization effects. Further, we simulate the SMPC optimization problem in its nonlinear form as Nonlinear SMPC (N-SMPC) to compare the computation time against its MILP counterpart, P-SMPC. However, N-SMPC becomes infeasible in the complex scenario, which is discussed in more detail later.

To assess if the generated trajectory can be attained, we provide the reference trajectories provided by the P-SMPC planner to a high-fidelity vehicle model in IPG CarMaker [202] and compare the position and velocity trajectories of the ego vehicle with their references.

The control frequency for all the simulations is set to 1kHz in accordance with the real-life applications where the computational capabilities limit the operational frequency of (digital) controllers [12]. The SMPC problems are all designed with sampling time of 0.2s and  $N_p = 10$ . We solve the MILPs using the GUROBI [154] optimizer and the NLPs using the SQP solver in fmincon in a MATLAB R2020b environment. For a fair comparison between the two solvers, we provide the object codes to speed-up the solution time of the NLPs, which in our simulations, has resulted in up to 20 times faster convergence compared to providing the objective and the constraints as MATLAB functions. The simulations were run on a PC with a 8-core(s) Intel Xeon 3.60 GHz CPU and 8 GB RAM on Windows 10 64-bit the codes are available from [207].

### 9.5.1 PROACTIVITY ASSESSMENT

In real life, some of the most dangerous situations on a highway are sudden appearance of a static object or extreme deceleration of a front vehicle. Therefore, we define different conceptual scenarios with slow-moving vehicles in all of them to present scenarios where the obstacle is so slow (or even static) that slowing down to keep the distance for collision avoidance is either impossible for the ego vehicle or extremely dangerous. As a result, we can test the ability of the planner in finding a safe, yet aggressive, evasive maneuver to avoid the collision. For this, the initial longitudinal velocity of the ego vehicle is considered to be 22m/s ( $\approx 80$ km/h), while the dynamic obstacles are assumed to have initial velocities between 8 to 11m/s ( $\approx 30$  to 40km/h). Nevertheless, we select the scenarios in a way to represent challenging, yet possible cases where e.g. other drivers do not aim to collide with each other, but may behave carelessly.

We use four conceptual scenarios to assess the solutions of the P-SMPC planner:

1. **Single obstacle:** A slow-moving obstacle is in front of the ego vehicle on the same lane. We expect the ego vehicle to avoid collision with this obstacle by performing an evasive maneuver, instead of slowing down to keep a safe distance.

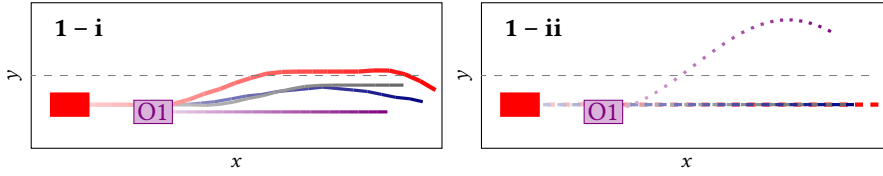
2. **Dynamic corridor:** In addition to an obstacle in the lane as the single-obstacle scenario, there is another slow-moving vehicle on the other lane to present a situation where the ego vehicle needs to pass through a corridor between two dynamic obstacles with stochastic behavior. Here we expect the ego vehicle to pass that corridor along an optimal trajectory.
3. **Static/dynamic corridor:** This scenario is similar to the dynamic corridor, except here we have a static object on the road instead of another slow-moving obstacle.
4. **Complex scenario:** Here we assess the planner in a situation where there are four slow-moving vehicles (two on each lane) and one static object present on the road. There exists a safe corridor between the dynamic and static obstacles, in which we expect the planner to find an optimal trajectory.

Moreover, each scenario is investigated twice: first as realization (i) where the obstacles behave ideally as the P-SMPC planner calculates  $z_{\text{ref}}$ , i.e. they keep their longitudinal velocity and lateral position, and secondly as realization (ii) where some/all of them either change their speed or their lane. Note that in realization (ii), the obstacle's intention to change lanes is not known a priori to the ego vehicle, as a result, the SMPC planners keep the assumption that the obstacle behaves as realization (i).

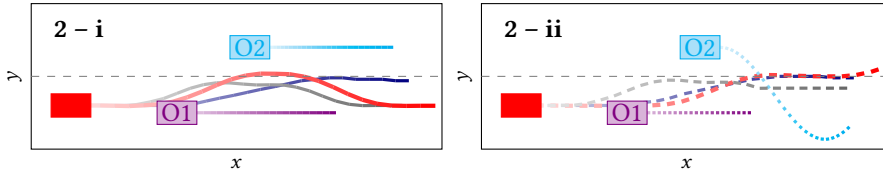
In total, we have conducted 400 Monte-Carlo simulations by perturbing the initial speed and the longitudinal distance between the ego vehicle and the obstacles with uniform sampling within a  $\pm 5\%$  range as an acceptable bound from the literature [185]. In Fig. 9.6, four examples are selected as most clear cases to showcase the efficacy of our approach in a more clear way. The statistical information regarding the Monte-Carlo simulations can be found in Fig. 9.11a. The ego vehicle is shown in red, while the obstacles are labeled by the letter "O" and a number to distinguish among them. The solid lines represent the case where the obstacles move according to the obstacle prediction model and keep their longitudinal velocity and lateral position. The dashed lines correspond to a case where the obstacles behave differently than the obstacle prediction model in P-SMPC, e.g. some of the obstacles on the road are accelerating/decelerating or intending to change their lanes. The solid red line shows the generated reference trajectory in cases with realization (i), while the dashed one shows the solution in realization (ii). R-SMPC and N-SMPC results are shown respectively in gray and blue in a similar fashion.

### SINGLE OBSTACLE

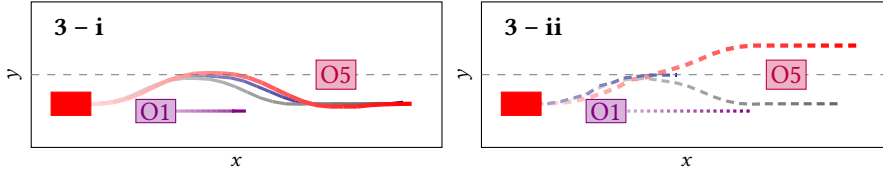
In the first scenario (Fig. 9.6a), the P-SMPC and R-SMPC planners avoid collision when the obstacle behaves as predicted by an evasive maneuver. However, the P-SMPC planner keeps a larger distance with a higher speed compared to the R-SMPC planner that converges to a trajectory that only satisfies the chance constraints (left-hand side of (9.25e)) and favors a solution that is closer to the middle of the lane. Note that the higher average velocity is visible by comparing the length of the red and gray trajectories. If the obstacle intends to change lanes which is not known to the ego vehicle a priori, the P-SMPC and R-SMPC planners both keep on assuming that the obstacle will keep its lateral position in each control step, but after the initial control steps, the planners converge to trajectories on the same lane as the obstacle merges into the next one. The difference between the planners is that the P-SMPC planner converges to slightly higher speed (since the red dashed line



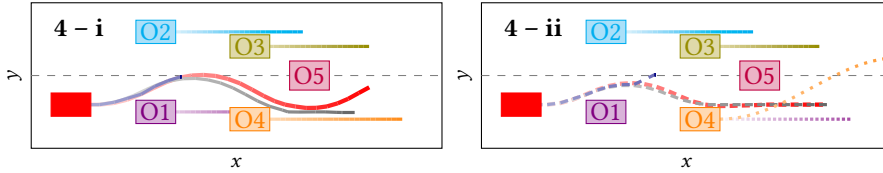
(a) Scenario 1: single obstacle



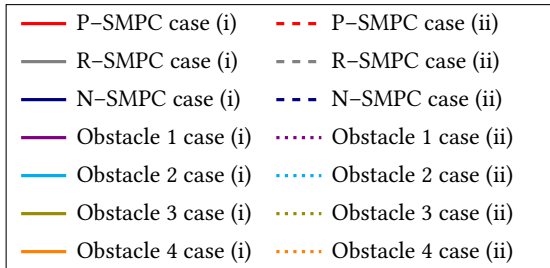
(b) Scenario 2: dynamic corridor



(c) Scenario 3: static/dynamic corridor



(d) Scenario 4: complex scenario



(e) Legend

Figure 9.6: Simulation results for proactivity assessment of the planners. The ego vehicle is shown by a red rectangle and the fading represents the trajectory evolution over time. Note that obstacle 5 (O5) is static.

extends more to the right) to keep more distance from the obstacle. In this sense, P-SMPC is more proactive as it manages to get out of the hazardous situation while ensuring a higher safety level.

### DYNAMIC CORRIDOR

Figure 9.6b shows the simulation results during the dynamic corridor scenario where both obstacles are moving. If the obstacles behave as predicted by the ego vehicle and intend to keep driving on the same lane, the P-SMPC and R-SMPC planners avoid the collision by overtaking O1 and returning to the center of the right lane. Here, the P-SMPC planner keeps more distance with O1 since it succeeds in finding a trajectory that has a lower collision risk than the left-hand side of (9.25e). However, if O2 actually intends to move to the right lane, after a few control steps when the ego vehicle observes the updated lateral position of O2, P-SMPC keeps more distance from the center of the right lane and eventually merges into the left lane as it detects this area to be the safest option. It should be noted that this is possible due to allowing switching between lanes in (9.25d). Otherwise, the planners would keep aiming for staying on the right lane which means driving on the center line between the two lanes until the right lane is risk-free. The R-SMPC planner, however, is not able to use this potential since it keeps a closer trajectory to the obstacles and does not search for other trajectories with lower collision-risk, as long as (9.25e) is satisfied. As a result, P-SMPC is more proactive in the sense of avoiding the propagation of hazard to the next time steps.

### STATIC/DYNAMIC CORRIDOR

In the dynamic/static corridor scenario, both the P-SMPC and R-SMPC planners avoid colliding with the obstacles by overtaking O1 as shown in Fig. 9.6c, where the P-SMPC planner keeps a larger distance with the “more uncertain” obstacle (O1). However, if O1 intends to increase its longitudinal velocity, the R-SMPC planner still converges to the same trajectory since it still satisfies the (9.25e), whereas P-SMPC changes lanes to the safer track and avoids the collision by overtaking the static obstacle O5 from the left. Similar to the dynamic corridor, this may lead to hazard propagation to the next steps, a problem which P-SMPC mitigates by proactive collision avoidance via finding a solution with a lower collision risk for future time steps.

## 9

### COMPLEX SCENARIO

Figure 9.6d shows the simulations for the complex scenario. If obstacles behave as predicted by the ego vehicle, the P-SMPC and R-SMPC planners manage to find a solution within the attainable corridor to avoid collision with the road users. In the final control steps, the left lane is empty and safer, therefore the P-SMPC planner decides to merge to the left lane, whereas R-SMPC keeps the same lane. However, if O1 steers to the right and O4 intends to merge into the left lane, the P-SMPC planner decides to stay in the same lane as the right lane is the safer one and suggests a similar trajectory as planned by the R-SMPC planner. Figure 9.7 shows the force plots during the complex scenario as an example to show the capability of the SMPC to operate close to the tire saturation limits. Note that the velocity of the ego vehicle during the maneuvers is not always constant and is discussed in more detail in the next section, accompanied by corresponding plots.

Note that the N-SMPC planner reaches infeasibility before the end of simulations in the last three cases, which leads to incomplete trajectories. This phenomenon is a result of using a warm start strategy (or solution using limited and insufficient number of initial guesses) which in turn leads to accumulation of errors after a few time steps as follows: in the complex scenario, the ego vehicle detects the obstacles 2 seconds before reaching their current position, e.g. O4 is detected after the ego initiates steering to avoid colliding with O1. Using the shifted solution of the previous time step in such cases leads to a poor result: as the previous solution was to go back to the initial lateral position after overtaking O1, by detecting O4, the planner converges to a solution that suggest going back to the initial lateral position after overtaking O4. Conversely, R-SMPC and P-SMPC planners are able to find a better solution thanks to their search for a global optimum, which is to brake and steer to the center of the lane to keep more distance from O4. In the next time steps, O5 is detected, and R-SMPC and P-SMPC manage to find a trajectory to steer to the center of the lane faster now that an obstacle is in the way. However, the poor solution in the previous time steps from the N-SMPC planner has resulted in higher longitudinal velocity. Therefore, the time to collision with O5 is shorter and it is infeasible to find a trajectory to avoid colliding with O5 with the current velocity.

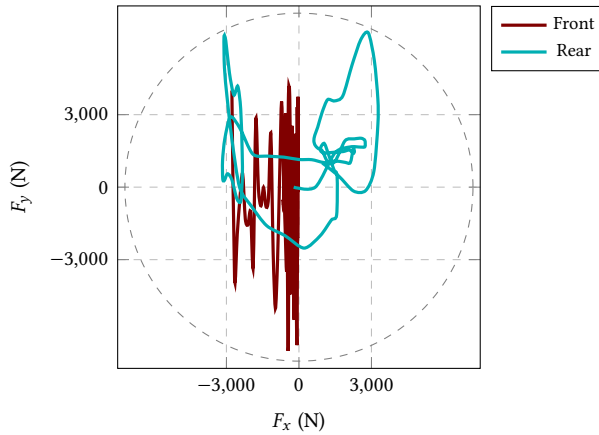


Figure 9.7: Force plot of the complex maneuver with the Kamm circle shown by dashed line.

### 9.5.2 ASSESSMENT OF ATTAINABLE TRAJECTORIES

To assess if the trajectories generated by the planner can be attained, we check whether they can be tracked by a high-fidelity vehicle model. In the proactivity test, eight reference trajectories were generated in total by the P-SMPC planner. To avoid repetition, we select four of these trajectories as distinct maneuvers and we simulate the high-fidelity BMW model in IPG CarMaker [202] to track them. It should be noted that the other trajectories produced similar results. The selected maneuvers are:

- a) **Constant-speed overtake:** scenario (2-i), the solid red line in Fig. 9.6b,

- b) **Decelerating overtake:** scenario (3-i), the solid red line in Fig. 9.6c,
- c) **Double overtake:** scenario (3-ii), the dashed red line in Fig. 9.6c, and
- d) **Lane change:** scenario (2-ii), the dashed red line in Fig. 9.6b.

In each simulation, we give the velocity vector in the four maneuvers to the longitudinal controller in IPG as the reference velocity profile, and provide the steering angles to the lateral controller for lateral motion. Figure 9.8 shows comparisons of the  $x_{ego}$ ,  $y_{ego}$ , and  $v$  trajectories obtained by the P-SMPC planner and the resulting trajectory of the IPG vehicle.

**Remark 9.6.** We start each IPG simulation from  $x_{ego} = 0m$  and run a steady, constant velocity maneuver for 200m to allow for the IPG model to stabilize before tracking the reference maneuver. As a result, the attainability tests start at  $x_{ego} = 200m$ .

Figure 9.8 shows that the reference trajectories provided by P-SMPC planner are attainable for the high-fidelity IPG model to track, with slight mismatch along the X axis, which is reasonable considering the larger complexity of the higher-fidelity model in IPG CarMaker, as compared to the prediction model in the P-SMPC planner.

### 9.5.3 COMPARISON WITH IPG MOTION PLANNER

As the final step, we showcase the proactivity and efficiency of the P-SMPC planner by comparing its behavior against the built-in collision avoidance module in IPG CarMaker simulation environment. The test scenario is similar to the complex scenario in Fig. 9.6d where one static and four slow-moving obstacles are present on the road. This time, we decrease the obstacle velocities even further down to 2-7 m/s. Moreover, we simulate a sudden braking by the last obstacle on the road until it stops in a dangerous way.

For a fair comparison, we set a “normal” but “risk-taking” driver behavior in IPG by selecting a standard driver and the maximum overtaking rate, which means that the driver always favors evading the obstacles rather than braking. This case shows how an overly-conservative planning strategy can lead to higher risk and propagating the hazard to other road users.

In the proactive collision avoidance case, we first ran the simulation in MATLAB and used the same TestRun in CarMaker. We arrange the maneuver in IPG such that the IPG driver merely tracks the speed profile and the steering wheel angle generated by the P-SMPC planner in MATLAB. Notably, we intentionally excluded considerations of other traffic participants in this scenario to prevent any interference with the operation of the IPG motion planner. The video of the comparison simulation is accessible online from <https://youtu.be/UacmQDjQ2vI>.

Figure 9.9a compares the velocity profiles for the overly-conservative IPG motion planner and the P-SMPC planner. While the P-SMPC planner manages to keep the velocity close to the cruising speed, the IPG planner dangerously brakes in multiple occasions. This issue becomes more critical when the IPG planner decides for a full stop behind the last obstacle on the right lane as shown in Fig. 9.10a: on the other hand, the P-SMPC planner manages to safely guide the ego vehicle outside of the risky zone between two slow-moving vehicles by taking a proactive strategy to overtake the stopping vehicle as

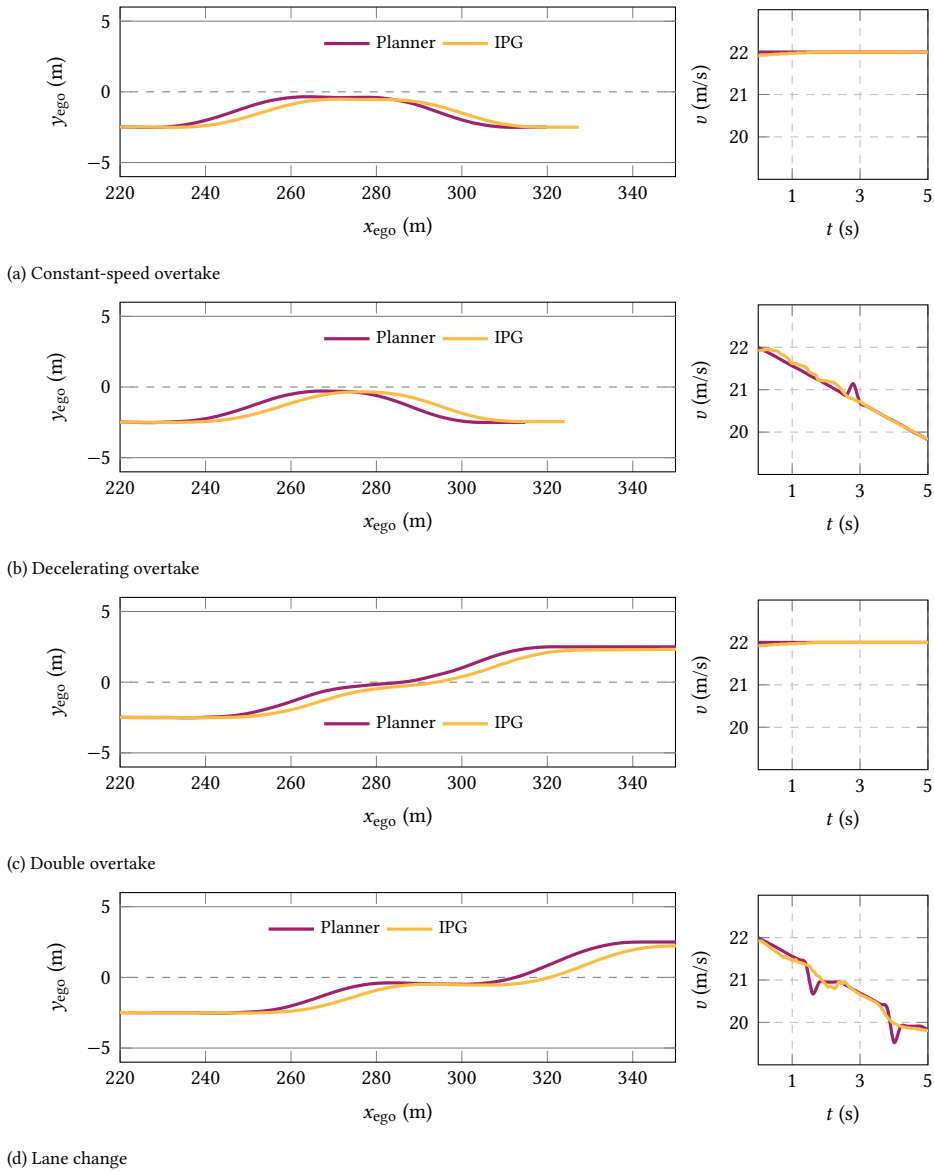
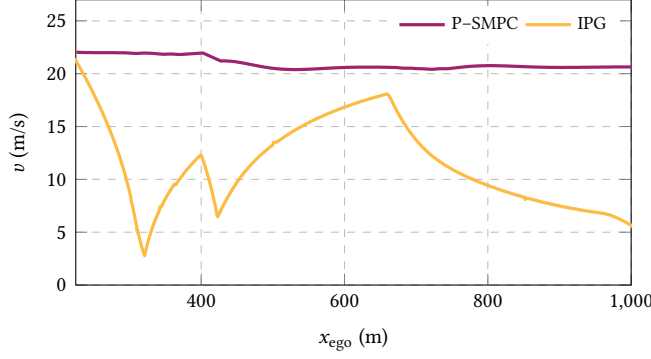
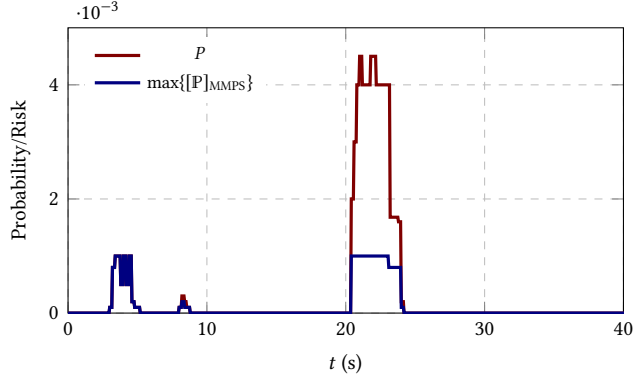


Figure 9.8: Simulation results for attainability assessment of the P-SMPC planner.

well as by keeping a safe distance from the other slow-moving obstacle on the left lane in Fig. 9.10b.



(a) Velocity profiles of the ego vehicle along the road



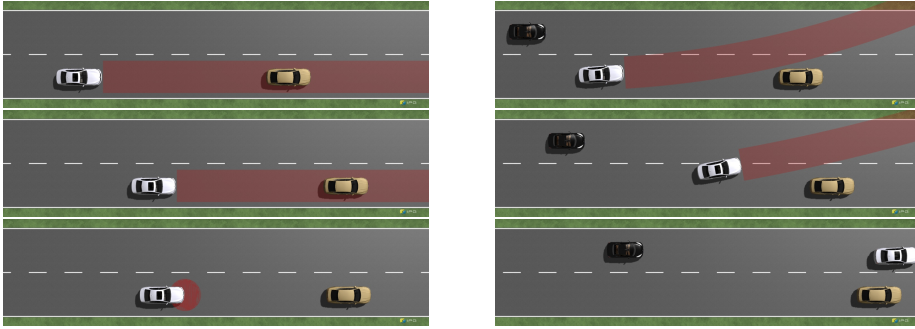
(b) Risk and maximum  $[P]_{\text{MMPS}}$  values for P-SMPC

Figure 9.9: Plots of comparative test between overly-conservative and proactive collision avoidance.

#### 9.5.4 PERFORMANCE ANALYSIS AND DISCUSSION

In the previous sections, we showed the proactivity of our proposed P-SMPC motion planner by comparing its performance against the state-of-the-art SMPC formulation (R-SMPC) and the built-in motion planner in a high-fidelity modeling and simulation platform. To gain a more clear view of the planning performance of P-SMPC, we have collected the data from all the aforementioned simulations and plotted the time evolution of chance constraints and the risk function values and the density histogram for computation time in Fig. 9.11. Since the simulations have various lengths in terms of time, we have scaled their data to a risky zone and a safe zone in Figures 9.11a and 9.11b to allow for a meaningful comparison. The risky zone represents the section of the simulations where the ego vehicle observes sudden appearance of the obstacles and ends when it does not detect any obstacles





(a) Overly-conservative collision avoidance: the ego vehicle slows down to keep distance until a full stop behind the obstacle. (b) Proactive collision avoidance by P-SMPC: the ego vehicle manages to get out of the risky zone before its front vehicle stops.

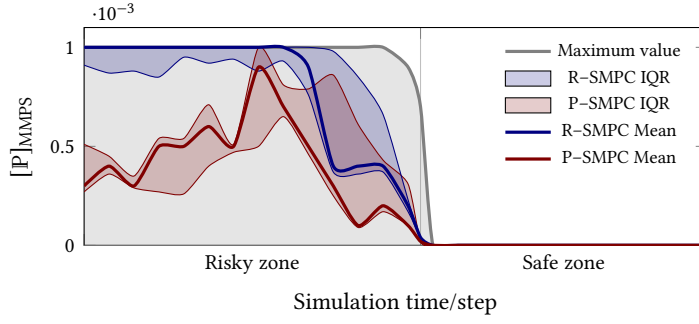
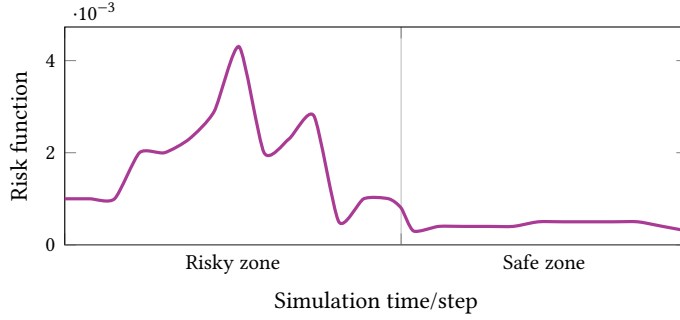
Figure 9.10: Snapshots of overly-conservative (a) and proactive (b) collision avoidance planning strategies.

ahead on the road.

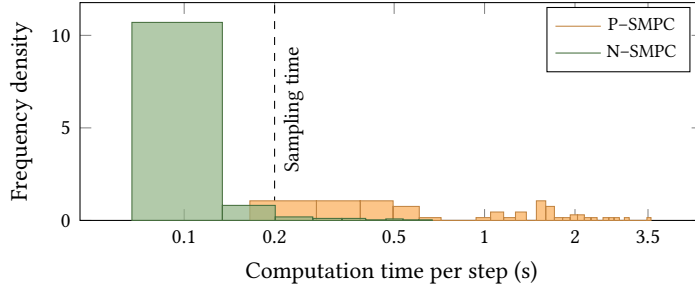
Figure 9.11a shows the statistical information of  $[P]_{\text{MMPS}}$  values in the Monte-Carlo simulation results for the R-SMPC and P-SMPC planners. The maximum values for both planners is 0.001 (0.1%) as shown in gray. Both planners show a reduction of the maximum  $[P]_{\text{MMPS}}$  value by getting out of the risky zone. However, the mean for  $[P]_{\text{MMPS}}$  values for P-SMPC are significantly lower than the mean values for R-SMPC, which shows the effectiveness of minimizing a risk function based on over-approximation of the  $P$  within the SMPC formulation. The peak in the mean value for P-SMPC corresponds to the riskiest time steps during the simulation, which occur where the vehicle is closest to the obstacle, e.g. during an overtaking maneuver. Further, the InterQuartile Range (IQR) distance for the planners is shown by the width of a shaded area around the mean values, using their corresponding colors.

The risk function values for P-SMPC planner are plotted in Fig. 9.11b. Since the risk function is an over-approximation of  $P$ , its value are higher than  $[P]_{\text{MMPS}}$ . Nevertheless, the P-SMPC planner manages to keep the risk function below 0.0045 (0.45%) at all times in Fig. 9.9b due to its predictive proactive collision avoidance. In addition, while convergence to a global optimum cannot be guaranteed for an NLP, an MILP solver can reach its global optimum when it is given sufficient time. As a result, the MILP formulation of the (originally nonlinear) SMPC planning optimization problem improves the computational efficiency by a speed-up in computations and a better coverage of the decision space.

Lastly, the density histogram for computation time per planning step is shown in Fig. 9.11c. Compared to the planner sampling time of 0.2s, the MILP solver could find the global optimum 96% of the times within 0.15s (75% of the time step) on the PC used for the test and only 4% of the times required more than 0.2s to find the global optimum. This shows the computational efficiency of the P-SMPC planning formulation, which can be further improved by imposing a time limit for the solver (and trading the global optimality) or running the simulations on a faster machine. Note that this level of computational efficiency is achieved for the assumed model and approximation accuracy adopted in this

(a) Evolution of  $[P]_{\text{MMPS}}$  values

(b) Evolution of maximum risk function values (P-SMPC)



(c) Density histogram for computation times. The N-SMPC computation times for steps that the NLP was infeasible are not considered and the data only account for the duration of sampling times where the planner converged to a solution.

Figure 9.11: Performance analysis of the P-SMPC planner in terms of safety and computation time. The data in these plots represent the density histograms of their corresponding variables considering all the performed simulations in this study.

chapter. For a more comprehensive study of control performance vs. computational speed trade-off in hybridization of NMPC using MMPS formalism, the reader is referred to our previous study [85, 86].

## 9.6 CONCLUSIONS

This chapter has presented a novel SMPC motion planner for emergency collision avoidance during hazardous highway scenarios. The proposed planner proactively avoids collision by static and dynamic obstacles on a highway by avoiding conservatism and swift response to sudden appearance of road users with uncertain behavior, thus improving the safety of the ego vehicle.

The novelties of our proposed approach can be summarized as follows: first, the proactive SMPC planner uses a tractable formulation of chance constraints for safe collision avoidance, while minimizing a risk function formulated as an over-estimation of the probabilities while facilitating the incorporation of a dynamic model for the ego vehicle as well as exploiting the tire-force potential close to the vehicle handling limits. Secondly, hybrid approximations of the nonlinearities in the system dynamics by the MMPS formalism are used to allow for an MILP formulation of the SMPC problem and facilitate real-time implementation and convergence to the global optimum. Safety, proactivity, and computational efficiency of our proposed planner were shown via various simulations of emergency scenarios and compared against the state-of-the-art SMPC formulation and a high-fidelity vehicle modeling and simulation environment.

For future work, we aim at improving the model for dynamic obstacles on the road and extending the uncertainty regarding the intention of the other road users. While the model employed in this chapter for the obstacles helped obtain an efficient computational accuracy-speed trade-off, more comprehensive models of obstacle behavior are influential for implementation of levels 4 and 5 of automated driving. Further, we aim at integrated planning and control design for emergency scenarios for improved accuracy and computational efficiency, in addition to investigating an efficient control structure to integrate our proposed SMPC planner with hybrid vehicle control and a friction estimator to account for the uncertainties of the environment as well. Moreover, in-depth calibration of probability bounds, investigation of suboptimality bounds, feasibility analysis of the SMPC problem for different probability formulations, and proof of recursive feasibility will be important topics for our future research, as well as designing a back-up mode in cases where the feasibility of the planning optimization problem cannot be guaranteed.



# 10

## CONCLUSIONS

*Sometimes the answer to a problem is neither complex nor simple, just unexpected.*

— Rouzbeh Moein, *The Cold Coffee of Mr. Writer*

This chapter serves as the conclusion to the thesis, starting with a summary of the research contributions. We then provide recommendations for future research, along with a discussion on the automated driving outlook, keeping an eye on its societal relevance and technological progress.

### 10.1 SUMMARY OF RESEARCH CONTRIBUTIONS

In this thesis, we have addressed the problem of controlling evasive maneuvers in hazardous scenarios for automated driving. In particular, we approached the problem focusing on a proactive response to hazard, taking into account the need for swift and optimal response to unforeseen emergencies on highways during a driving task. In summary, our research contributions can be categorized as follows:

- **two cut-based PieceWise Affine (PWA) approximation approaches** have been proposed to allow for a flexible approximation of multi-dimensional nonlinear systems in Chapters 2 and 3,
- **an open-source toolbox named H4MPC** has been introduced in Chapter 4 to offer a user-friendly interface for hybridizing Model Predictive Control (MPC) problems,
- **a sensitivity analysis of PWA approximations** was conducted in Chapter 5 for a polytopically-constrained NonLinear Program (NLP) to obtain quantitative bounds on the distance between the original and the approximated local minima,
- **a hybridization benchmark for MPC optimization problems** using the Max-Min-Plus-Scaling (MMPS) formalism has been presented in Chapter 6, followed by a comparative assessment of the computational performance of the hybridized MPC optimization problems in Chapter 7,

- **experimental insights into real-time implementation** have been discussed for emergency collision avoidance after the sudden appearance of a static obstacle in Chapter 8,
- **a proactive Stochastic Model Predictive Control (SMPC) approach for collision avoidance** in highway scenarios has been presented to minimize the collision risk and enhances safety using chance constraints in Chapter 9.

## 10.2 FUTURE RESEARCH SUGGESTIONS

For the continued progress of automated driving research, it is essential that future research explores certain aspects in greater detail. For instance, the following points should be considered:

**Realistic scenarios acknowledge a wide range of uncertainties.** Therefore, it is essential to introduce further levels of uncertainty, such as tire-road friction coefficients, pneumatic pressure of the tires, and load distribution in the ego vehicle. Additionally, unforeseen events, such as abrupt weather changes or road blockage caused by a sudden snow accumulation, should be considered in future research steps.

**Hazard extends beyond configuration of the vehicles.** Refining, as well as expanding the definition of hazardous cases is essential for advancing automated driving research. Although various threat assessment measures have been proposed in the literature, the successful implementation of automated vehicles demands a broader understanding of what constitutes a hazardous scenario. For example, in a Level 3 automated vehicle, a danger alarm triggered by detecting a pedestrian may cause the human driver to panic and lose control; resulting in a situation with a higher collision risk than the mere presence of the pedestrian. Therefore, hazards must account for not only metrics such as time/distance to collision but also other influential parameters, such as the state of the driver.

**Human-replacing technology calls for a human-inspired design.** In this context, studying human behavior, such as the decision-making process of average, as well as highly-skilled drivers like Formula 1 participants, allows for incorporating crucial insights, from modeling to the application of more realistic probability distributions, into the automated driving planning and control systems.

**It is time for nomenclature convergence.** The development of more comprehensive and improved approximation techniques is becoming increasingly important, especially as the complexity of the control problems rises with the targeted automation level. The new approximation methods should focus on delivering faster results while minimizing computational complexity. In light of this, closer collaboration with the learning community is essential as the fields of function approximation and learning often deal with similar types of optimization problems. However, the use of different terminologies (e.g. error function vs. loss function, parameter estimation vs training, and adaptive approximation vs. online learning) in each field creates a significant barrier. This mismatch in nomenclature

not only makes communication difficult but also causes researchers to miss out on insights from the other field. To create true synergy, these gaps should be bridged by aligning the vocabularies among the fields of optimization, approximation, and learning.

**Technology matters.** Technological advancements have substantial impact on the progress of the field. To name a few, enhancing sensing technologies, hardware processing speeds and memory capacity, and reducing overall energy consumption are all significant aspects of advancing automated driving. Additionally, rethinking mechanical design could provide further advancements in this area as well. For instance, similar to how adaptive suspension transformed chassis control in ground vehicles or load-sensing bearings developed by SKF significantly improved slip-control, introducing concepts such as moving loads in vehicle dynamics could enhance safety during evasive maneuvers.

## 10.3 OUTLOOK AND RECOMMENDATIONS

Moving beyond the technical aspects, the successful implementation of automated driving necessitates addressing the following higher-level considerations, as they touch on the broader view of the future of automated driving and its integration into everyday life.

**Global problems require global solutions.** The implementation of automated driving presents a global challenge: a Level 5 automated vehicle must perform well in diverse environments—from the highways of San Diego and Santiago to the congested streets of Dar es Salaam, Kathmandu, and Amsterdam, and even the rural areas surrounding Brisbane. Achieving this will require the collaboration among different global sectors, as each region comes with its own set of regulations, levels of public trust, and environmental challenges such as weather patterns and infrastructure variations.

**Let us not forget who automated cars are made for.** It is crucial to investigate the foundations of public trust and ensure that it is addressed within a global, diversity-informed framework. This means that the diverse backgrounds and needs of future users must be considered at every stage, from system design to production. For instance, understanding how individuals from different cultural, social, and demographic backgrounds perceive and trust autonomous vehicles is essential.

**Humans do not trust black boxes.** In other words, people are less likely to trust something they do not understand. To ensure the acceptance of automated driving systems, it is crucial to provide the public with clear and transparent information about how these technologies work. In this sense, research findings must be communicated effectively, not only via open science, but also through public talks and workshops to address public concerns and align with their expectations.





## BIBLIOGRAPHY

- [1] S. Liu, K. Zheng, L. Zhao, and P. Fan. A driving intention prediction method based on hidden Markov model for autonomous driving. *Computer Communications*, 157:143–149, 2020.
- [2] I. Batkovic, M. Ali, P. Falcone, and M. Zanon. Safe trajectory tracking in uncertain environments. *IEEE Transactions on Automatic Control*, 68(7):4204–4217, 2023.
- [3] X. Zhang, A. Liniger, and F. Borrelli. Optimization-based collision avoidance. *IEEE Transactions on Control Systems Technology*, 29(3):972–983, 2021.
- [4] Introduction: From highly automated to autonomous driving. In G. Dimitrakopoulos, A. Tsakanikas, and E. Panagiotopoulos, editors, *Autonomous Vehicles*, pages xiii–xxiii. Elsevier, 2021.
- [5] Society of Automotive Engineers (SAE International). Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems, Sep, 3, 2017. [https://web.archive.org/web/20170903105244/https://www.sae.org/misc/pdfs/automated\\_driving.pdf](https://web.archive.org/web/20170903105244/https://www.sae.org/misc/pdfs/automated_driving.pdf).
- [6] I. Dickson. How the supply chain is pioneering autonomous driving, 2013. [Online; accessed October 2, 2024].
- [7] U. Ju and S. Kim. Willingness to take responsibility: Self-sacrifice versus sacrificing others in takeover decisions during autonomous driving. *Heliyon*, 10(9):e29616, 2024.
- [8] N. Chowdhri, L. Ferranti, F. S. Iribarren, and B. Shyrokau. Integrated nonlinear model predictive control for automated driving. *Control Engineering Practice*, 106:104654, 2021.
- [9] V. A. Laurence and J. C. Gerdes. Long-horizon vehicle motion planning and control through serially cascaded model complexity. *IEEE Transactions on Control Systems Technology*, 30(1):166–179, 2022.
- [10] J. Wurts, J. L. Stein, and T. Ersal. Design for real-time nonlinear model predictive control with application to collision imminent steering. *IEEE Transactions on Control Systems Technology*, 30(6):2450–2465, 2022.
- [11] H. Liu, L. Zhang, P. Wang, and H. Chen. A real-time NMPC strategy for electric vehicle stability improvement combining torque vectoring with rear-wheel steering. *IEEE Transactions on Transportation Electrification*, 8(3):3825–3835, 2022.

- [12] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti. Model predictive path tracking control for automated road vehicles: A review. *Annual Reviews in Control*, 55:194–236, 2023.
- [13] G. Bellegarda and Q. Nguyen. Dynamic vehicle drifting with nonlinear MPC and a fused kinematic-dynamic bicycle model. *IEEE Control Systems Letters*, 6:1958–1963, 2022.
- [14] Manan S. Gandhi, Bogdan Vlahov, Jason Gibson, Grady Williams, and Evangelos A. Theodorou. Robust model predictive path integral control: analysis and performance guarantees. *IEEE Robotics and Automation Letters*, 6(2):1423–1430, 2021.
- [15] S. Dixit, U. Montanaro, M. Dianati, D. Oxtoby, T. Mizutani, A. Mouzakitis, and S. Fallah. Trajectory planning for autonomous high-speed overtaking in structured environments using robust MPC. *IEEE Transactions on Intelligent Transportation Systems*, 21(6):2310–2323, 2020.
- [16] Q. Shi, Ji. Zhao, A. E. Kamel, and I. Lopez-Juarez. MPC based vehicular trajectory planning in structured environment. *IEEE Access*, 9:21998–22013, 2021.
- [17] H. Atoui, O. Sename, V. Milanés, and J. J. Martinez. LPV-based autonomous vehicle lateral controllers: A comparative analysis. Technical Report hal-03197895, 2021. <https://hal.archives-ouvertes.fr/hal-03197895>.
- [18] J. Lunze and F. Lamnabhi-Lagarrigue. *Handbook of Hybrid Systems Control: Theory, Tools, Applications*. Cambridge University Press, 2009.
- [19] M. Pčolka, E. Žáčková, S. Čelikovský, and M. Šebek. Toward a smart car: Hybrid nonlinear predictive controller with adaptive horizon. *IEEE Transactions on Control Systems Technology*, 26(6):1970–1981, 2018.
- [20] Y. Zheng and B. Shyrokau. A real-time nonlinear MPC for extreme lateral stabilization of passenger vehicles. In *IEEE International Conference on Mechatronics (ICM 2019)*, pages 519–524, 2019.
- [21] X. Sun, Y. Cai, S. Wang, X. Xu, and L. Chen. Optimal control of intelligent vehicle longitudinal dynamics via hybrid model predictive control. *Robotics and Autonomous Systems*, 112:190–200, 2019.
- [22] L. Gharavi, B. De Schutter, and S. Baldi. Parametric piecewise-affine approximation of nonlinear systems: A cut-based approach. *IFAC-PapersOnLine*, 56(2):6666–6671, 2023.
- [23] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.
- [24] B. De Schutter, T. van den Boom, J. Xu, and S. S. Farahani. Analysis and control of max-plus linear discrete-event systems: An introduction. *Discrete Event Dynamic Systems: Theory and Applications*, 30:25–54, 2020.

- [25] R. M. F. ra Cândido, L. Hardouin, M. Lhommeau, and R. S. Mendes. Conditional reachability of uncertain max plus linear systems. *Automatica*, 94:426–435, 2018.
- [26] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.
- [27] Richard Oberdieck and Efstratios N. Pistikopoulos. Explicit hybrid model-predictive control: The exact solution. *Automatica*, 58:152–159, 2015.
- [28] B. De Schutter and T. J.J. van den Boom. MPC for continuous piecewise-affine systems. *Systems and Control Letters*, 52(3-4):179–192, 2004.
- [29] G. Ferrari-Trecate, F. A. Cuzzola, D. Mignone, and M. Morari. Analysis of discrete-time piecewise affine and hybrid systems. *Automatica*, 38(12):2139–2146, 2002.
- [30] M. Hovd and S. Olaru. Parameter-dependent PWQ Lyapunov function stability criteria for uncertain piecewise linear systems. *Modeling, Identification and Control*, 39:15–21, 2018.
- [31] V. Sessa, L. Iannelli, F. Vasca, and V. Acary. A complementarity approach for the computation of periodic oscillations in piecewise linear systems. *Nonlinear Dynamics*, 85:1255–1273, 2016.
- [32] D. Ito, T. Ueta, T. Kousaka, and K. Aihara. Bifurcation analysis of the Nagumo-Sato model and its coupled systems. *International Journal of Bifurcation and Chaos*, 26, 2016.
- [33] T. Suzuki and K. Aihara. Nonlinear system identification for prostate cancer and optimality of intermittent androgen suppression therapy. *Mathematical Biosciences*, 245:40–48, 2013.
- [34] B. Joseph-Duran, M. N. Jung, C. Ocampo-Martinez, S. Sager, and G. Cembrano. Minimization of sewage network overflow. *Water Resources Management*, 28:41–63, 2014.
- [35] X. Sun, We. Hu, Y. Cai, P.K. Wong, and L. Chen. Identification of a piecewise affine model for the tire cornering characteristics based on experimental data. *Nonlinear Dynamics*, 101:857–874, 2020.
- [36] D. Jagga, M. Lv, and S. Baldi. Hybrid adaptive chassis control for vehicle lateral stability in the presence of uncertainty. In *Mediterranean Conference on Control and Automation (MED 2018)*, pages 529–534, 2018.
- [37] S. Azuma, J. Imura, and T. Sugie. Lebesgue piecewise affine approximation of nonlinear systems. *Nonlinear Analysis: Hybrid Systems*, 4(1):92–102, 2010.
- [38] S. Kersting and M. Buss. Recursive estimation in piecewise affine systems using parameter identifiers and concurrent learning. *International Journal of Control*, 92(6):1264–1281, 2019.

- [39] Y. Du, F. Liu, J. Qiu, and M. Buss. Online identification of piecewise affine systems using integral concurrent learning. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(10):4324–4336, 2021.
- [40] L. Bako. Subspace clustering through parametric representation and sparse optimization. *IEEE Signal Processing Letters*, 21(3):356–360, 2014.
- [41] E. Khanmirza, M. Nazarahari, and A. Mousavi. Identification of piecewise affine systems based on fuzzy PCA-guided robust clustering technique. *Eurasip Journal on Advances in Signal Processing*, 2016:1–15, 2016.
- [42] L. Q. Thuan, T. van den Boom, and S. Baldi. Online identification of continuous bimodal and trimodal piecewise affine systems. In *European Control Conference (ECC 2016)*, pages 1075–1070, 2016.
- [43] A. Hartmann, J. M. Lemos, R. S. Costa, J. Xavier, and S. Vinga. Identification of switched ARX models via convex optimization and expectation maximization. *Journal of Process Control*, 28:9–16, 2015.
- [44] Delft High Performance Computing Centre (DHPC). *DelftBlue Supercomputer (Phase 1)*, 2022. ARK: ark:/44463/DelftBluePhase1.
- [45] F. Lauer and G. Bloch. *Hybrid System Identification: Theory and Algorithms for Learning Switching Models*, volume 478 of *Lecture Notes in Control and Information Sciences*. Springer International Publishing, 2019.
- [46] V. V. Gorokhovik, O. I. Zorko, and G. Birkhoff. Piecewise affine functions and polyhedral sets. *Optimization*, 31(3):209–221, 1994.
- [47] F. Bayat, T. A. Johansen, and A. A. Jalali. Flexible piecewise function evaluation methods based on truncated binary search trees and lattice representation in explicit MPC. *IEEE Transactions on Control Systems Technology*, 20(3):632–640, 2012.
- [48] A. Gersnoviez, M. Brox, and I. Baturone. High-speed and low-cost mplementation of explicit model predictive controllers. *IEEE Transactions on Control Systems Technology*, 27(2):647–662, 2019.
- [49] D. Bertsimas and J. Dunn. *Machine Learning Under a Modern Optimization Lens*. Dynamic Ideas LLC, 2019.
- [50] F. Lauer. On the complexity of piecewise affine system identification. *Automatica*, 62:148–153, 2015.
- [51] P. Brox, J. Castro-Ramirez, M. C. Martinez-Rodriguez, E. Tena, C. J. Jimenez, I. Baturone, and A. J. Acosta. A programmable and configurable ASIC to generate piecewise-affine functions defined over general partitions. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(12):3182–3194, 2013.
- [52] X. Sun, P. Wu, Y. Cai, S. Wang, and L. Chen. Piecewise affine modeling and hybrid optimal control of intelligent vehicle longitudinal dynamics for velocity regulation. *Mechanical Systems and Signal Processing*, 162, 2022.

- [53] G. F. V. Amaral, C. Letellier, and L. A. Aguirre. Piecewise affine models of chaotic attractors: The Rössler and Lorenz systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 16(1), 2006.
- [54] T. Ohtsuki, T. Fujisawa, and S. Kumagai. Existence theorems and a solution algorithm for piecewise-linear resistor networks. *SIAM Journal on Mathematical Analysis*, 8(1):69–99, 1977.
- [55] C. Wen and X. Ma. A max-piecewise-linear neural network for function approximation. *Neurocomputing*, 71:843–852, 2008.
- [56] A. Bemporad. A piecewise linear regression and classification algorithm with application to learning and model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 68(6):3194–3209, 2023.
- [57] X. Tang and Y. Dong. Expectation maximization based sparse identification of cyber-physical system. *International Journal of Robust and Nonlinear Control*, 31(6):2044–2060, 2021.
- [58] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal. Identification of hybrid systems: A tutorial. *European Journal of Control*, 13:242–260, 2007.
- [59] A. Moradvandi, R.E.F. Lindeboom, E. Abraham, and B. De Schutter. Models and methods for hybrid system identification: a systematic survey. *IFAC-PapersOnLine*, 56(2):95–107, 2023.
- [60] S. Paoletti, I. Savelli, A. Garulli, and A. Vicino. A bilevel programming framework for piecewise affine system identification. In *IEEE Conference on Decision and Control (CDC)*, pages 7376–7381. IEEE, 2019.
- [61] J. F. Bard. *Practical bilevel optimization: algorithms and applications*. Springer, 2011.
- [62] P. Mattsson, D. Zachariah, and P. Stoica. Recursive identification method for piecewise ARX models: A sparse estimation approach. *IEEE Transactions on Signal Processing*, 64(19):5082–5093, 2016.
- [63] P. Mattsson, D. Zachariah, and P. Stoica. Recursive nonlinear-system identification using latent variables. *Automatica*, 93:343–351, 2018.
- [64] V. Breschi, D. Piga, and A. Bemporad. Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, 73:155–162, 2016.
- [65] Z. Jin, Q. Shen, and S. Z. Yong. Mesh-based piecewise affine abstraction with polytopic partitions for nonlinear systems. *IEEE Control Systems Letters*, 5(5):1543–1548, 2021.
- [66] A. Oliveri, A. Canepa, L. Queirolo, and M. Storace. An algorithm for automatic domain partitioning of piecewise-affine model predictive control laws. In *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, pages 217–220, 2013.

- [67] E. Amaldi, S. Coniglio, and L. Taccari. Discrete optimization methods to fit piecewise affine models to data points. *Computers & Operations Research*, 75:214–230, 2016.
- [68] Y. Du, F. Liu, J. Qiu, and M. Buss. A semi-supervised learning approach for identification of piecewise affine systems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 67(10):3521–3532, 2020.
- [69] F. Comaschi, B. A. G. Genuit, A. Oliveri, W. P. M. H. Heemels, and M. Storace. FPGA implementations of piecewise affine functions based on multi-resolution hyperrectangular partitions. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 59(12):2920–2933, 2012.
- [70] A. Bemporad, A. Oliveri, T. Poggi, and M. Storace. Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations. *IEEE Transactions on Automatic Control*, 56(12):2883–2897, 2011.
- [71] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.
- [72] J. Xu, T. J.J. van den Boom, B. De Schutter, and S. Wang. Irredundant lattice representations of continuous piecewise affine functions. *Automatica*, 70:109–120, 2016.
- [73] L. Gharavi, B. De Schutter, and S. Baldi. Parametric piecewise-affine approximation of nonlinear systems: A cut-based approach. *IFAC-PapersOnLine*, 56(2):6666–6671, 2023.
- [74] T. Zaslavsky. *Facing Up to Arrangements: Face-Count Formulas for Partitions of Space by Hyperplanes*, volume 154. American Mathematical Society, 1975.
- [75] L. E. Blumenson. A derivation of n-dimensional spherical coordinates. *The American Mathematical Monthly*, 67(1):63–66, 1960.
- [76] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10. Springer Science & Business Media, 1987.
- [77] P. Orlik and H. Terao. *Arrangements of hyperplanes*. Springer Science & Business Media, 2013.
- [78] T. Fujisawa and E. S. Kuh. Piecewise-linear theory of nonlinear networks. *SIAM Journal on Applied Mathematics*, 22(2):307–328, 1972.
- [79] L. Gharavi, B. De Schutter, and S. Baldi. H4MPC: A hybridization toolbox for model predictive control in automated driving. In *IEEE International Conference on Advanced Motion Control (AMC)*, pages 1–6, 2024.
- [80] N. Groot, B. De Schutter, and H. Hellendoorn. Integrated model predictive traffic and emission control using a piecewise-affine approach. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):587–598, 2013.
- [81] E. Asarin, T. Dang, and A. Girard. Hybridization methods for the analysis of nonlinear systems. *Acta Informatica*, 43(7):451, 2007.

- [82] F.D. Torrisi and A. Bemporad. HYSDEL-a tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Transactions on Control Systems Technology*, 12(2):235–249, 2004.
- [83] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad. Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain. *IEEE Transactions on Control Systems Technology*, 21(4):1236–1248, 2013.
- [84] N. Guo, X. Zhang, Y. Zou, B. Lenzo, and T. Zhang. A computationally efficient path-following control strategy of autonomous electric vehicles with yaw motion stabilization. *IEEE Transactions on Transportation Electrification*, 6(2):728–739, 2020.
- [85] L. Gharavi, B. De Schutter, and S. Baldi. Efficient MPC for emergency evasive maneuvers, part I: Hybridization of the nonlinear problem. *arXiv preprint arXiv:2310.00715*, 2023.
- [86] L. Gharavi, B. De Schutter, and S. Baldi. Efficient MPC for emergency evasive maneuvers, part II: Comparative assessment for hybrid control. *arXiv preprint arXiv:2310.00716*, 2023.
- [87] L. Gharavi. Hybridization Toolbox for Model Predictive Control. 4TU.ResearchData, 2023.
- [88] C. G. Bobier-Tiu, C. E. Beal, J. C. Keggelman, R. Y. Hindiyyeh, and J. C. Gerdes. Vehicle control synthesis using phase portraits of planar dynamics. *Vehicle System Dynamics*, 57(9):1318–1337, 2019.
- [89] R. Rajamani. *Vehicle Dynamics and Control*. Springer Science & Business Media, 2011.
- [90] H. Pacejka. *Tire and Vehicle Dynamics*. Elsevier, 2005.
- [91] A. Kripfganz and R. Schulze. Piecewise affine functions as a difference of two convex functions. *Optimization*, 18(1):23–29, 1987.
- [92] L. Gharavi, B. De Schutter, and S. Baldi. Sensitivity analysis for piecewise-affine approximations of nonlinear programs with polytopic constraints. *IEEE Control Systems Letters*, (XX), 2024.
- [93] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl. From linear to nonlinear MPC: bridging the gap via the real-time iteration. *International Journal of Control*, 93(1):62–80, 2020.
- [94] D. W. Griffith, L. T. Biegler, and S. C. Patwardhan. Robustly stable adaptive horizon nonlinear model predictive control. *Journal of Process Control*, 70:109–122, 2018.
- [95] L. Dong, J. Yan, X. Yuan, H. He, and C. Sun. Functional nonlinear model predictive control based on adaptive dynamic programming. *IEEE Transactions on Cybernetics*, 49(12):4206–4218, 2019.



- [96] V. M Zavala and L. T. Biegler. The advanced-step NMPC controller: Optimality, stability and robustness. *Automatica*, 45(1):86–93, 2009.
- [97] J. Xu, X. Huang, X. Mu, and S. Wang. Model predictive control based on adaptive hinging hyperplanes model. *Journal of Process Control*, 22(10):1821–1831, 2012.
- [98] L. Gharavi, A. Dabiri, J. Verkuijlen, B. De Schutter, and S. Baldi. Proactive emergency collision avoidance for automated driving in highway scenarios. *arXiv preprint*, 2310.17381, 2023.
- [99] Y. Chen, M. Bruschetta, D. Cuccato, and A. Beghi. An adaptive partial sensitivity updating scheme for fast nonlinear model predictive control. *IEEE Transactions on Automatic Control*, 64(7):2712–2726, 2019.
- [100] M. Ławryńczuk, P. M. Marusak, P. Chaber, and D. Seredyński. Initialisation of optimisation solvers for nonlinear model predictive control: Classical vs. hybrid methods. *Energies*, 15(7):2483, 2022.
- [101] T. Ghandriz, B. Jacobson, P. Nilsson, and L. Laine. Trajectory-following and off-tracking minimisation of long combination vehicles: A comparison between nonlinear and linear model predictive control. *Vehicle System Dynamics*, pages 1–34, 2023.
- [102] J. W. Daniel. Stability of the solution of definite quadratic programs. *Mathematical Programming*, 5(1):41–53, 1973.
- [103] H. X. Phu. Minimizing convex functions with bounded perturbations. *SIAM Journal on Optimization*, 20(5):2709–2729, 2010.
- [104] C. Büskens and H. Maurer. Sensitivity analysis and real-time optimization of parametric nonlinear programming problems. In M. Grötschel, S. O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems*, pages 3–16. Springer Berlin Heidelberg, 2001.
- [105] I. Subotic, A. Hauswirth, and F. Dorfler. Quantitative sensitivity bounds for nonlinear programming and time-varying optimization. *IEEE Transactions on Automatic Control*, 67(6):2829–2842, 2022.
- [106] S. Shin and V. M. Zavala. Diffusing-horizon model predictive control. *IEEE Transactions on Automatic Control*, 68(1):188–201, 2023.
- [107] H. Pirnay, R. López-Negrete, and L. T. Biegler. Optimal sensitivity based on IPOPT. *Mathematical Programming Computation*, 4(4):307–331, 2012.
- [108] M. A. Muller, D. Angeli, and F. Allgower. On necessity and robustness of dissipativity in economic model predictive control. *IEEE Transactions on Automatic Control*, 60(6):1671–1676, 2015.
- [109] Y. Puranik and N. V. Sahinidis. Domain reduction techniques for global NLP and MINLP optimization. *Constraints*, 22(3):338–376, 2017.



- [110] D. Whitley, S. Rana, J. Dzuber, and K. E. Mathias. Evaluating evolutionary algorithms. *Artificial Intelligence*, 85(1):245–276, 1996.
- [111] L.O. Chua and A.-C. Deng. Canonical piecewise-linear representation. *IEEE Transactions on Circuits and Systems*, 35(1):101–111, 1988.
- [112] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti. Model predictive path tracking control for automated road vehicles: A review. *Annual Reviews in Control*, 2022.
- [113] Y. Huang and Y. Chen. Vehicle lateral stability control based on shiftable stability regions and dynamic margins. *IEEE Transactions on Vehicular Technology*, 69(12):14727–14738, 2020.
- [114] J. K. Subosits and J. C. Gerdes. Impacts of model fidelity on trajectory optimization for autonomous vehicles in extreme maneuvers. *IEEE Transactions on Intelligent Vehicles*, 6(3):546–558, 2021.
- [115] V. Z. Patterson, F. E. Lewis, and J. C. Gerdes. Optimal decision making for automated vehicles using homotopy generation and nonlinear model predictive control. In *IEEE Intelligent Vehicles Symposium*, pages 1045–1050, 2021.
- [116] M. Metzler, D. Tavernini, P. Gruber, and A. Sorniotti. On prediction model fidelity in explicit nonlinear model predictive vehicle stability control. *IEEE Transactions on Control Systems Technology*, 29(5):1964–1980, 2021.
- [117] D. Tavernini, M. Metzler, P. Gruber, and A. Sorniotti. Explicit nonlinear model predictive control for electric vehicle traction control. *IEEE Transactions on Control Systems Technology*, 27(4):1438–1451, 2019.
- [118] K. Oh and J. Seo. Development of an adaptive and weighted model predictive control algorithm for autonomous driving with disturbance estimation and grey prediction. *IEEE Access*, 10:35251–35264, 2022.
- [119] T. Brudigam, M. Olbrich, D. Wollherr, and M. Leibold. Stochastic model predictive control with a safety guarantee for autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 8(1):22–36, 2023.
- [120] M. Rokonzaman, N. Mohajer, and S. Nahavandi. Effective adoption of vehicle models for autonomous vehicle path tracking: a switched MPC approach. *Vehicle System Dynamics*, pages 1–24, 2022.
- [121] K. Zhang, J. Sprinkle, and R. G. Sanfelice. Computationally aware control of autonomous vehicles: a hybrid model predictive control approach. *Autonomous Robots*, 39:503–517, 2015.
- [122] T. Zhao, E. Yurtsever, R. Chladny, and G. Rizzoni. Collision avoidance with transitional drift control. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 907–914, 2021.

- [123] M. Corno, G. Panzani, F. Roselli, M. Giorelli, D. Azzolini, and S. M. Savaresi. An LPV approach to autonomous vehicle path tracking in the presence of steering actuation nonlinearities. *IEEE Transactions on Control Systems Technology*, 29(4):1766–1774, 2020.
- [124] M. Amir and T. Givargis. Hybrid state machine model for fast model predictive control: Application to path tracking. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 185–192, 2017.
- [125] E. Pérez, C. Ariño, F. X. Blasco, and M. A. Martínez. Explicit predictive control with non-convex polyhedral constraints. *Automatica*, 48(2):419–424, 2012.
- [126] P. Tøndel, T. Johansen, and A. Bemporad. An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, 39(3):489–497, 2003.
- [127] N. Kochdumper and M. Althoff. Computing non-convex inner-approximations of reachable sets for nonlinear continuous systems. In *IEEE Conference on Decision and Control (CDC)*, pages 2130–2137, 2020.
- [128] X. Miao, Y. Song, Z. Zhang, and S. Gong. Successive convexification for ascent trajectory replanning of a multistage launch vehicle experiencing nonfatal dynamic faults. *IEEE Transactions on Aerospace and Electronic Systems*, 58(3):2039–2052, jun 2022.
- [129] P. Scheffe, T. Mario Henneken, M. Kloock, and B. Alrifaae. Sequential convex programming methods for real-time optimal trajectory planning in autonomous vehicle racing. *IEEE Transactions on Intelligent Vehicles*, 2021.
- [130] B. Açıkmeşe, J. M. Carson, and L. Blackmore. Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113, 2013.
- [131] R. Deits and R. Tedrake. Computing large convex regions of obstacle-free space through semidefinite programming. In *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*, pages 109–124. 2015.
- [132] K. Okamoto and P. Tsiotras. Optimal stochastic vehicle path planning using covariance steering. *IEEE Robotics and Automation Letters*, 4(3):2276–2281, 2019.
- [133] L. Yao. Nonparametric learning of decision regions via the genetic algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(2):313–321, 1996.
- [134] J. Xu. Morphological decomposition of 2-D binary shapes into conditionally maximal convex polygons. *Pattern Recognition*, 29(7):1075–1104, 1996.

- [135] L. Yao and K. S. Weng. Learning decision regions based on adaptive ellipsoids. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 22(1):41–73, 2014.
- [136] X. Wei, M. Liu, Z. Ling, and H. Su. Approximate convex decomposition for 3D meshes with collision-aware concavity and tree search. *ACM Transactions on Graphics*, 41(4):1–18, 2022.
- [137] R. Bulbul and A. U. Frank. AHD: The alternate hierarchical decomposition of nonconvex polytopes (generalization of a convex polytope based spatial data model). In *International Conference on Geoinformatics*, pages 1–6, 2009.
- [138] Q. Zhang, I. E. Grossmann, A. Sundaramoorthy, and J. M. Pinto. Data-driven construction of convex region surrogate models. *Optimization and Engineering*, 17:289–332, 2016.
- [139] P. Duhr, A. Sandeep, A. Cerofolini, and C. H. Onder. Convex performance envelope for minimum lap time energy management of race cars. *IEEE Transactions on Vehicular Technology*, 71(8):8280–8295, 2022.
- [140] T. Fu, H. Zhou, and Z. Liu. NMPC-based path tracking control strategy for autonomous vehicles with stable limit handling. *IEEE Transactions on Vehicular Technology*, 71(12):12499–12510, 2022.
- [141] N. Zhang, S. Yang, G. Wu, H. Ding, Z. Zhang, and K. Guo. Fast distributed model predictive control method for active suspension systems. *Sensors*, 23(6):3357, 2023.
- [142] G. Zhu, H. Jie, and W. Hong. Nonlinear model predictive path tracking control for autonomous vehicles based on orthogonal collocation method. *International Journal of Control, Automation and Systems*, 21:257–270, 2023.
- [143] M. M. Ghazaei Ardakani, B. Olofsson, A. Robertsson, and R. Johansson. Model predictive control for real-time point-to-point trajectory generation. *IEEE Transactions on Automation Science and Engineering*, 16(2):972–983, 2019.
- [144] M. Diehl, H. J. Ferreau, and N. Haverbeke. Efficient numerical methods for nonlinear MPC and moving horizon estimation. *Nonlinear Model Predictive Control: Towards New challenging Applications*, pages 391–417, 2009.
- [145] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal. Combined speed and steering control in high-speed autonomous ground vehicles for obstacle avoidance using model predictive control. *IEEE Transactions on Vehicular Technology*, 66(10):8746–8763, 2017.
- [146] Y. Vaupel, N. C. Hamacher, A. Caspari, A. Mhamdi, I. G. Kevrekidis, and A. Mitsos. Accelerating nonlinear model predictive control through machine learning. *Journal of Process Control*, 92:261–270, 2020.

- [147] B. C. van Huijgevoort, S. Weiland, and S. Haesaert. Temporal logic control of nonlinear stochastic systems using a piecewise-affine abstraction. *IEEE Control Systems Letters*, 7:1039–1044, 2023.
- [148] T. Besselmann and M. Morari. Hybrid parameter-varying model predictive control for autonomous vehicle steering. *European Journal of Control*, 14(5):418–431, 2008.
- [149] M. Rokonzaman, N. Mohajer, S. Nahavandi, and S. Mohamed. Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intelligent Transport Systems*, 15(5):646–670, 2021.
- [150] H. Zhao, H. Yang, Z. Wang, and H. Li. Nonlinear switched model predictive control with multiple Lyapunov functions for trajectory tracking and obstacle avoidance of nonholonomic systems. *International Journal of Robust and Nonlinear Control*, 33(11):6171–6187, 2023.
- [151] H. Dugoff, P. S. Fancher, and L. Segel. An analysis of tire traction properties and their influence on vehicle dynamic performance. *SAE Transactions*, 79:1219–1243, 1970.
- [152] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes. Safe driving envelopes for path tracking in autonomous vehicles. *Control Engineering Practice*, 61:307–316, 2017.
- [153] J. Y. Goh, T. Goel, and J. C. Gerdes. Toward automated vehicle control beyond the stability limits: drifting along a general path. *Journal of Dynamic Systems, Measurement and Control*, 142(2):021004, 2020.
- [154] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.
- [155] K. Holmström, A. O. Göran, and M. M. Edvall. User’s guide for TOMLAB 7. Technical report, Tomlab Optimization Inc, 2010.
- [156] F. Garrido and P. Resende. Review of decision-making and planning approaches in automated driving. *IEEE Access*, 10:100348–100366, 2022.
- [157] M. Antkiewicz, M. Kahn, M. Ala, K. Czarnecki, P. Wells, A. Acharya, and S. Beiker. Modes of automated driving system scenario testing: Experience report and recommendations. *SAE International Journal of Advances and Current Practices in Mobility*, 2(4):2248–2266, 2020.
- [158] E. De Gelder, H. Elrofai, A. K. Saberi, J. Paardekooper, O. Op Den Camp, and B. De Schutter. Risk quantification for automated driving systems in real-world driving scenarios. *IEEE Access*, 9:168953–168970, 2021.
- [159] J. Dahl, C. De Campos, G. R. and Olsson, and J. Fredriksson. Collision avoidance: A literature review on threat-assessment techniques. *IEEE Transactions on Intelligent Vehicles*, 4(1):101–113, 2019.
- [160] J. Fan, Y. Zhan, and J. Liang. A hierarchical control strategy for reliable lane changes considering optimal path and lane-changing time point. *IET Intelligent Transport Systems*, page itr2.12460, 2023.

- [161] X. T. Yang, K. Huang, Z. Zhang, Z. A. Zhang, and F. Lin. Eco-driving system for connected automated vehicles: Multi-objective trajectory optimization. *IEEE Transactions on Intelligent Transportation Systems*, 22(12):7837–7849, 2021.
- [162] R. Chai, A. Tsourdos, S. Chai, Y. Xia, A. Savvaris, and C. L. P. Chen. Multiphase overtaking maneuver planning for autonomous ground vehicles via a desensitized trajectory optimization approach. *IEEE Transactions on Industrial Informatics*, 19(1):74–87, January 2023.
- [163] T. Qie, W. Wang, C. Yang, Y. Li, Y. Zhang, W. Liu, and C. Xiang. An improved model predictive control-based trajectory planning method for automated driving vehicles under uncertainty environments. *IEEE Transactions on Intelligent Transportation Systems*, 24(4):3999–4015, 2023.
- [164] Y. Wang, X. Cao, and Y. Hu. A trajectory planning method of automatic lane change based on dynamic safety domain. *Automotive Innovation*, 6(3):466–480, 2023.
- [165] F. Wang, T. Shen, M. Zhao, Y. Ren, Y. Lu, B. Feng, and G. Yin. Lane-change trajectory planning and control based on stability region for distributed drive electric vehicle. *IEEE Transactions on Vehicular Technology*, 73(1):504–521, 2024.
- [166] H. Dong, Q. Wang, W. Zhuang, G. Yin, K. Gao, Z. Li, and Z. Song. Flexible eco-cruising strategy for connected and automated vehicles with efficient driving lane planning and speed optimization. *IEEE Transactions on Transportation Electrification*, 10(1):1530–1540, 2024.
- [167] X. Shang and A. Eskandarian. Emergency collision avoidance and mitigation using model predictive control and artificial potential function. *IEEE Transactions on Intelligent Vehicles*, 8(5):3458–3472, 2023.
- [168] H. Guo, C. Shen, H. Zhang, H. Chen, and R. Jia. Simultaneous trajectory planning and tracking using an MPC method for cyber-physical systems: A case study of obstacle avoidance for an intelligent vehicle. *IEEE Transactions on Industrial Informatics*, 14(9):4273–4283, 2018.
- [169] K. Li, Z. Yin, Y. Ba, Y. Yang, Y. Kuang, and E. Sun. An integrated obstacle avoidance controller based on scene-adaptive safety envelopes. *Machines*, 11(2):303, 2023.
- [170] S. Li and Q. Zhao. Research on the emergency obstacle avoidance strategy of intelligent vehicles based on a safety distance model. *IEEE Access*, 11:7124–7134, 2023.
- [171] Y. K. Nakka and S. Chung. Trajectory optimization of chance-constrained nonlinear stochastic systems for motion planning under uncertainty. *IEEE Transactions on Robotics*, 39(1):203–222, 2023.
- [172] Z. Sun, R. Wang, X. Meng, Y. Yang, Z. Wei, and Q. Ye. A novel path tracking system for autonomous vehicle based on model predictive control. *Journal of Mechanical Science and Technology*, 38(1):365–378, 2024.

- [173] H. Sun, S. Zhang, L. Dai, and S. V. Raković. Locally convexified rigid tube MPC. *IET Control Theory & Applications*, 17(4):446–462, 2023.
- [174] M. Ammour, R. Orjuela, and M. Basset. A MPC combined decision making and trajectory planning for autonomous vehicle collision avoidance. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):24805–24817, 2022.
- [175] S. Yang, H. Zheng, J. Wang, and A. Kamel. A personalized human-like lane-changing trajectory planning method for automated driving system. *IEEE Transactions on Vehicular Technology*, 70(7):6399–6414, 2021.
- [176] H. Lu, Y. Liu, M. Zhu, C. Lu, H. Yang, and Y. Wang. Enhancing interpretability of autonomous driving via human-like cognitive maps: A case study on lane change. *IEEE Transactions on Intelligent Vehicles*, pages 1–11, 2024.
- [177] M. Galvani, F. Biral, B. M. Nguyen, and H. Fujimoto. Four wheel optimal autonomous steering for improving safety in emergency collision avoidance manoeuvres. In *IEEE International Workshop on Advanced Motion Control (AMC)*, pages 362–367, 2014.
- [178] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, 66(2):952–964, 2017.
- [179] I. Batkovic, U. Rosolia, M. Zanon, and P. Falcone. A robust scenario MPC approach for uncertain multi-modal obstacles. *IEEE Control Systems Letters*, 5(3):947–952, 2021.
- [180] Y. Chen, U. Rosolia, W. Uellacker, N. Csomay-Shanklin, and A. D. Ames. Interactive multi-modal motion planning with branch model predictive control. *IEEE Robotics and Automation Letters*, 7(2):5365–5372, 2022.
- [181] K. Liu, N. Li, H. E. Tseng, I. Kolmanovsky, and A. Girard. Interaction-aware trajectory prediction and planning for autonomous vehicles in forced merge scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 24(1):474–488, 2023.
- [182] M. Cannon, B. Kouvaritakis, and X. Wu. Probabilistic constrained MPC for multiplicative and additive stochastic uncertainty. *IEEE Transactions on Automatic Control*, 54(7):1626–1632, 2009.
- [183] H. Zhu and J. Alonso-Mora. Chance-constrained collision avoidance for MAVs in dynamic environments. *IEEE Robotics and Automation Letters*, 4(2):776–783, 2019.
- [184] S. X. Wei, A. Dixit, S. Tomar, and J. W. Burdick. Moving obstacle avoidance: A data-driven risk-aware approach. *IEEE Control Systems Letters*, 7:289–294, 2023.
- [185] R. Chai, A. Tsourdos, A. Savvaris, S. Wang, Y. Xia, and S. Chai. Fast generation of chance-constrained flight trajectory for unmanned vehicles. *IEEE Transactions on Aerospace and Electronic Systems*, 57:1028–1045, 2021.
- [186] D. Moser, R. Schmied, H. Waschl, and L. del Re. Flexible spacing adaptive cruise control using stochastic model predictive control. *IEEE Transactions on Control Systems Technology*, 26(1):114–127, 2018.

- [187] H. Kazemi, H. N. Mahjoub, A. Tahmasbi-Sarvestani, and Y. P. Fallah. A learning-based stochastic MPC design for cooperative adaptive cruise control to handle interfering vehicles. *IEEE Transactions on Intelligent Vehicles*, 3(3):266–275, 2018.
- [188] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, 2016.
- [189] M. Muñoz Sánchez, D. Pogosov, E. Silvas, D. C. Mocanu, J. Elfring, and M. J. G. van de Molengraft. Situation-aware drivable space estimation for automated driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):9615–9629, 2022.
- [190] M. Bujarbaruah, X. Zhang, M. Tanaskovic, and F. Borrelli. Adaptive stochastic MPC under time-varying uncertainty. *IEEE Transactions on Automatic Control*, 66(6):2840–2845, 2021.
- [191] V. Fors, P. Anistratov, B. Olofsson, and L. Nielsen. Predictive force-centric emergency collision avoidance. *Journal of Dynamic Systems, Measurement and Control*, 143(8):1–12, 2021.
- [192] R. Hajiloo, M. Abroshan, A. Khajepour, A. Kasaiezadeh, and S. K. Chen. Integrated steering and differential braking for emergency collision avoidance in autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):3167–3178, 2021.
- [193] T. Benciolini, D. Wollherr, and M. Leibold. Non-conservative trajectory planning for automated vehicles by estimating intentions of dynamic obstacles. *IEEE Transactions on Intelligent Vehicles*, 8(3):2463–2481, 2023.
- [194] N. Malone, H. Chiang, K. Lesser, M. Oishi, and L. Tapia. Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field. *IEEE Transactions on Robotics*, 33(5):1124–1138, 2017.
- [195] L. Petrović, I. Marković, and I. Petrović. Mixtures of Gaussian processes for robot motion planning using stochastic trajectory optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(12):7378–7390, 2022.
- [196] M. Schuurmans, A. Katriniok, C. Meissen, H. E. Tseng, and P. Patrinos. Safe, learning-based MPC for highway driving under lane-change uncertainty: A distributionally robust approach. *Artificial Intelligence*, 320:103920, 2023.
- [197] S. Paternain and A. Ribeiro. Stochastic artificial potentials for online safe navigation. *IEEE Transactions on Automatic Control*, 65(5):1985–2000, 2020.
- [198] J. Wang, Y. Yan, K. Zhang, Y. Chen, M. Cao, and G. Yin. Path planning on large curvature roads using driver-vehicle-road system based on the kinematic vehicle model. *IEEE Transactions on Vehicular Technology*, 71(1):311–325, 2022.

- [199] L. Schäfer, S. Manzinger, and M. Althoff. Computation of solution spaces for optimization-based trajectory planning. *IEEE Transactions on Intelligent Vehicles*, 8(1):216–231, 2023.
- [200] F. Eiras, M. Hawasly, S. V. Albrecht, and S. Ramamoorthy. A two-stage optimization-based motion planner for safe urban driving. *IEEE Transactions on Robotics*, 38(2):822–834, 2022.
- [201] A. Caregnato-Neto, M. R. O. A. Maximo, and R. J. M. Afonso. Real-time motion planning and decision-making for a group of differential drive robots under connectivity constraints using robust MPC and mixed-integer programming. *Advanced Robotics*, 37(5):356–379, 2023.
- [202] CarMaker. IPG Automotive.
- [203] Y. Chen, C. Hu, and J. Wang. Motion planning with velocity prediction and composite nonlinear feedback tracking control for lane-change strategy of autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 5(1):63–74, 2020.
- [204] C. Kuo, A. Schaarschmidt, Y. Cui, T. Asfour, and T. Matsubara. Uncertainty-aware contact-safe model-based reinforcement learning. *IEEE Robotics and Automation Letters*, 6(2):3918–3925, 2021.
- [205] M. Tiger, D. Bergström, A. Norrstig, and F. Heintz. Enhancing lattice-based motion planning with introspective learning and reasoning. *IEEE Robotics and Automation Letters*, 6(3):4385–4392, 2021.
- [206] T. Brüdigam, M. Olbrich, D. Wollherr, and M. Leibold. Stochastic model predictive control with a safety guarantee for automated driving: Extended version. *arXiv*, (2009.09381), 2022.
- [207] L. Gharavi. Proactive Motion Planning Codes for Emergency Collision Avoidance in Highway Scenarios. 4TU.ResearchData, 2023.



# CURRICULUM VITÆ

## Leila GHARAVI

[ lejlɑ ɣærævi ]



Born on May 5, 1995 in Iran, Leila obtained her B.Sc. and M.Sc. degrees in Mechanical Engineering from Amirkabir University of Technology, with research experience in automatic manufacturing and production, vibration analysis, control of nonlinear dynamics, and soft rehabilitation robotics.

She started her Ph.D. research in February 2021 at Delft Center for Systems and Control, Delft University of Technology, The Netherlands. Her doctoral research focuses on nonlinear and hybrid systems, optimization, and model-predictive control, with applications to adaptive and proactive control of automated vehicles in hazardous scenarios. Moreover, Leila was a visiting research fellow at the e-Mobility and Control lab in The University of Tokyo in Japan from February to April 2024.

Leila thrives on exploring complexity in every aspect of her life. She delights in solving intricate jigsaw puzzles, finding joy in uncovering hidden patterns, and channels her creativity into arts and crafts projects that challenge her problem-solving skills. Her love for sociopolitical literature reflects this same curiosity, as she seeks to understand the intricacies of human life.

Openly vocal about diversity, Leila advocates for awareness and inclusion across all dimensions of human identity, including neurological, gender, orientation, racial, and cultural aspects. Through her work and her voice, she strives to foster understanding and create a more inclusive world.



# LIST OF PUBLICATIONS

## JOURNALS

9. J. Verkuijlen, **L. Gharavi**, A. Dabiri, B. De Schutter, and S. Baldi, Advancing Safety in Highway Automated Driving: a Stochastic Model Predictive Approach to Emergency Motion Planning, *Under review*.
8. **L. Gharavi**, S. Baldi, Y. Hosomi, T. Sato, B.M. Nguyen, and H. Fujimoto, Dodging the Moose: Experimental Insights in Real-Life Automated Collision Avoidance, *Under review*.
7. **L. Gharavi**, B. De Schutter and S. Baldi, Iterative Cut-Based PWA Approximation of Multi-Dimensional Nonlinear Systems, *Under review*.
6. **L. Gharavi**, A. Dabiri, J. Verkuijlen, B. De Schutter and S. Baldi, Proactive emergency collision avoidance for automated driving in highway scenarios, *IEEE Transactions on Control Systems Technology*, early access.
5. **L. Gharavi**, B. De Schutter and S. Baldi, Efficient MPC for Emergency Evasive Maneuvers, Part II: Comparative Assessment for Hybrid Control, *Under review*.
4. **L. Gharavi**, B. De Schutter and S. Baldi, Efficient MPC for Emergency Evasive Maneuvers, Part I: Hybridization of the Nonlinear Problem, *Under review*.
3. **L. Gharavi**, C. Liu, B. De Schutter and S. Baldi, Sensitivity analysis for piecewise-affine approximations of nonlinear programs with polytopic constraints, *IEEE Control Systems Letters*, 8, pp. 1271-1276, 2024.
2. **L. Gharavi**, M. Zareinejad, and A. Ohadi, Continuum analysis of a soft bending actuator dynamics, *Mechatronics*, 83, p. 102739, 2022.
1. **L. Gharavi**, M. Zareinejad, and A. Ohadi, Dynamic Finite-Element analysis of a soft bending actuator, *Mechatronics*, 81, p. 102690, 2022.

## CONFERENCES

3. **L. Gharavi**, C. Liu, B. De Schutter and S. Baldi, Sensitivity analysis for piecewise-affine approximations of nonlinear programs with polytopic constraints, *IEEE International Conference Decision and Control (CDC)*, 2024.
  2. **L. Gharavi**, B. De Schutter and S. Baldi, H4MPC: a hybridization toolbox for model predictive control in automated driving, *IEEE International Conference on Advance Motion Control (AMC)*, pp. 1-6, 2024.
  1. **L. Gharavi**, B. De Schutter and S. Baldi, Parametric piecewise-affine approximation of nonlinear systems: a cut-based approach, *IFAC-PapersOnLine*, 56 (2), pp. 6666-6671, 2023.
- Included in this thesis.

