THE SLOTHBOT: A ROBOTIC PLATFORM FOR ENVIRONMENTAL EXPLORATION AND SURVEILLANCE

By Carmen Jimenez Cortes

Directed by Dr. Magnus Egerstedt

Comillas Pontifical University (ICAI School of Engineering)

Official Master's Degree in Industrial Engineering (MII)

Georgia Institute of Technology

Dec 2021

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título ……THE SLOTHBOT : A ROBOTIC PLATFORM FOR ENVIRONMENTAL……

……EXPLORATION AND SURVEILLANCE……

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico …2° Mii…… es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: CJiménez Fecha: .18../ .10../ .21.

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: [illegible] Fecha: November 9, 2021 ……/ ……/ ……

THE SLOTHBOT: A ROBOTIC PLATFORM FOR ENVIRONMENTAL EXPLORATION AND SURVEILLANCE

By Carmen Jimenez Cortes

Directed by Dr. Magnus Egerstedt

Comillas Pontifical University (ICAI School of Engineering)

Official Master's Degree in Industrial Engineering (MII)

Georgia Institute of Technology

Dec 2021

RESUMEN

Introducción

El ritmo de desarrollo tecnológico e industrial, en conjunto con el impacto humano sobre la explotación de reursos, el cambio climático o la huella de dióxido de carbono ha llevado a nuestro medioambiente a encontrarse en una situación comprometida. Investigadores de todas las áreas están realizando un esfuerzo colectivo para mejorar este hecho antes de alcanzar un punto de no retorno. Uno de los ecosistemas más importantes a analizar son las copas de los árboles, dado que son donde la energía solar se convierte en hidratos de carbono para nutrir ecosistemas completos, y también son hogar de más del cincuenta por ciento de la biodiversidad terrestre.

Para obtener una representación precisa de estas, los datos y medidas deben ser recogidos continuamente y sobre áreas extensas y no estacionarias. Este factor presenta un desafío donde la robótica puede hacer una importante contribución. En este Trabajo de Fin de Master proponemos un robot que ayudará en la caracterización de las copas de los árboles. Esta misión puede ser expresada como la realización de dos tareas: exploración medioambiental y monitorización medioambiental. El factor que hace particularmente difíciles estas tareas es que ambas necesitan ser ejecutadas durante periodos de tiempo que superan la capacidad de la batería del robot, necesitan ser *persistificadas*. El trabajo realizado previamente en la persistificación de las tareas de robots utilizaba dinámicas de control afines. Cuando una dependencia entre el voltage de la batería y el control aplicado se incluye en el modelo, las dinámicas del sistema dejan de ser control afines y por tanto toda la teoría previa debe ser revisada.

Objetivos

Los objetivos que se han satisfecho en este Trabajo Fin de Master son:

1. Desarrollar una nueva versión del actual robot: The SlohtBot

2. Ampliar la teoría existente acerca de Barreras de Control para su uso con sistemas con dinámicas dependientes en la energía de control

3. Implementar nuevos algoritmos de control basados en Barreas de Control para la persitificación de tareas robóticas

4. Estudiar e implementar el método denominado *Smoothing Splines* para la caracterización de funciones escalares desconocidas, como la intensidad lumínica

5. Establecer las bases matemáticas para la aplicación recursiva del método *Smoothing Splines* y así mantener las dimensiones del problema a resolver constante cuando nuevos sets de datos son incluidos

Solución

La solución propuesta consiste en un robot para la exploración y vigilancia medioambiental, así como las teoría y derivaciones matemáticas necesarias para alcanzar estos objetvios.

El SlothBot [1], mostrado en Figure 1, es el robot diseñado para la exploración de los ecosistemas de las copas de los árboles. Su dinámica es unidimensional, dado que se encuentra instalado en un cable colgado entre dos árboles. La versión Figure 1 estuvo instalada en el Jardín Botánico de Atlanta por más de 13 meses. Durante este tiempo, el robot se encontraba transversando el cable mientras recogía medidas. Alguna de estas medidas eran: la calidad del aire, la intensidad lumínica, temperatura y humedad.

Para garantizar la supervivencia del robot en un entorno exterior y desconocido, es necesario que las variables de estado, posición y voltage de la batería, permanezcan dentro

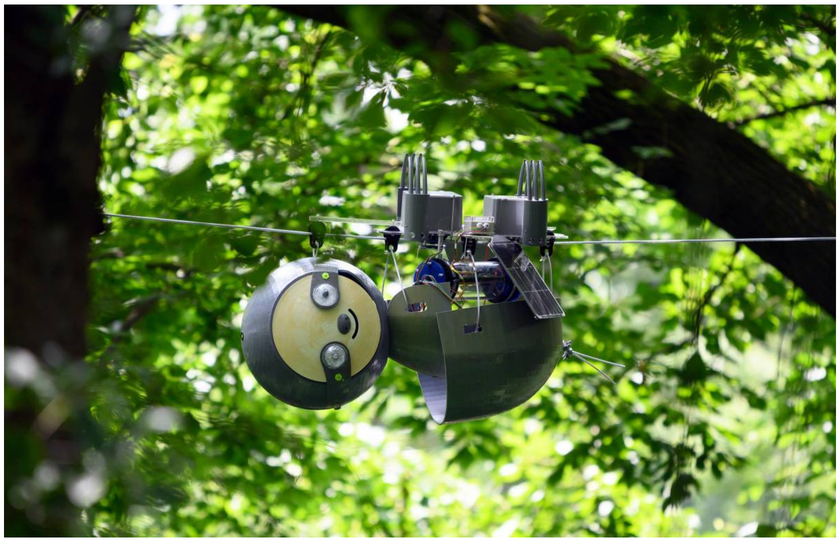


Figure 1: SlothBot en el Jardín Botánico de Atlanta

de unos límites seguros. Para ello, el modelo del sistema debe incluir tanto las dinámicas del robot como el efecto del medioambiente. El modelo empleado resulta:

$$\begin{pmatrix} \dot{p} \\ \dot{E} \end{pmatrix} = \begin{pmatrix} u \\ K(\frac{1}{1+\frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E) - \rho u^2 \end{pmatrix} \tag{1}$$

En Equation 1 se puede observar que al incluir una dependencia entre la batería del robot y la energía del control aparece un término cuadrático en la ecuación dinámica de la batería. Debido a ello, el sistema deja de ser control afín.

El problema de cumplir la misión del robot mientras se garantiza su supervivencia se ha expresado como la minimización del cuadrado de la norma de la diferencia entre el control nominal $\hat{u}$, que es el comando para satisfacer la misión del robot, y el control seguro a aplicar u. Este control seguro u satisface dos restricciones que garantizan que el sistema sea invariable. Estas restricciones son las Barreras de Control.

Expresando estos objetivos en terminos matemáticos, se obtiene el problema de optimización presentado en Equation 2:

$$\begin{aligned} &\underset{u}{minimize} \quad ||u - \hat{u}||^2 \\ &s.t. \\ &\quad - L_g h_1(X) u^2 \leq \gamma_1 h_1(X) + L_f h_1(X) \\ &\quad - L_g h_2(X) u \leq \gamma_2 h_1(X) + L_f h_2(X) \end{aligned} \tag{2}$$

Donde:

- $X = [E \quad p]$ es el vector de estado
- $L_f h_i, L_g h_i$ son las derivadas de Lie de la barrera h_i.

La función de la intensidad lumnínica $I(p)$ juega un papel fundamental en la dinámica de la batería, presentada en Equation 2. Obtener un modelo preciso de esta función permitiría al robot realizar misiones de exploración más ambicioses, ya que se conocería en adelanto que áreas le permitirían recargar su batería y cuales agotarla.

El método de los *Smoothing Splines* consiste en una ténica para la identificación de modelos, basada en la teoría de control clásica. Su objetivo es encontrar el control que hace que la salida de un sistema arbitrario se aproxime a una serie de valores en unos instantes de interpolación dados. Para obtener este control, se resuelve un problema de minimización sobre la energía de control empleada. La salida obtenida al aplicar dicho control es lo que se denomina un *Smoothing Spline*. Las medidas tomadas se consideran imperfectas y por tanto el método propuesto no busca un ajuste perfecto entre la salida y los puntos dados, sino un modelo que resuma todas las muestras de manera general. Esta estrategia conlleva una menor carga computacional y por tanto require un menor tiempo para resolver el problema de optimización.

La teoría detrás de este aplicación de los *Smoothing Splines* fue publicada con Egerstedt y Clide [2], y puede encontrare en el cuerpo de esta tesis.

Resultados

Para resolver el problema en Equation 2 puede aproximarse como un Problema Cuadrático con Restricciones Cuadráticas (QCQP), para el cual existen algoritmos de optimización convexa. El problema de todos estos algoritmos es su alta carga computacional y por tanto demanda energética, lo que los hace incompatibles con el objetivo de garantizar la longevidad del sistema. Para la aplicación del SlothBot, al poseer dinámicas unidimensionales, solo hay una entrada de control disponible, por lo que el problema resulta escalar tanto en la función de coste como en las restricciones. Esta nueva clase de problemas ha sido denominada Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problems. El hecho de que se trate de un problema escalar permite encontrar una solución analítica al mismo, que una vez implementada require una menor carga computacional que cualquiera de las posibles alternativas. La existencia de esta solución prueba qu el robot no acabará en una situación donde su batería no puede volver a ser recargada, bajo condiciones de operación nominales.

Esta solución analítica fue implementada en un simulador del Slothbot Figure 2. En esta simulación, el robot comienza en una posición $p = 0$ y con un 20% de su batería disponible, i.e. $E = 0.2$, durante la ejecución el robot recorre el cable indefinidamente. En Figure 3 se puede observar como las Barreras de Control modifican la velocidad, y por tanto la posición, del robot para garantizar que su batería no se agota o sobrecarga. Un análisis en mayor detalle de los resultados obtenidos puede encontrarse en el cuerpo de esta tesis.

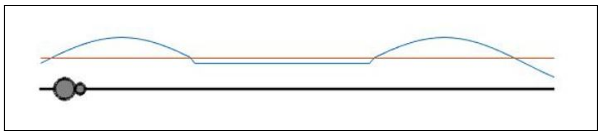


Figure 2: Intensidad Lumínica $I(p)$ (azul), Intensidad Lumínica umbral I_c (naranja), y SlothBot simulado

Los resultados obtenidos cuando el método basado en *Smoothing Splines* se aplica a la

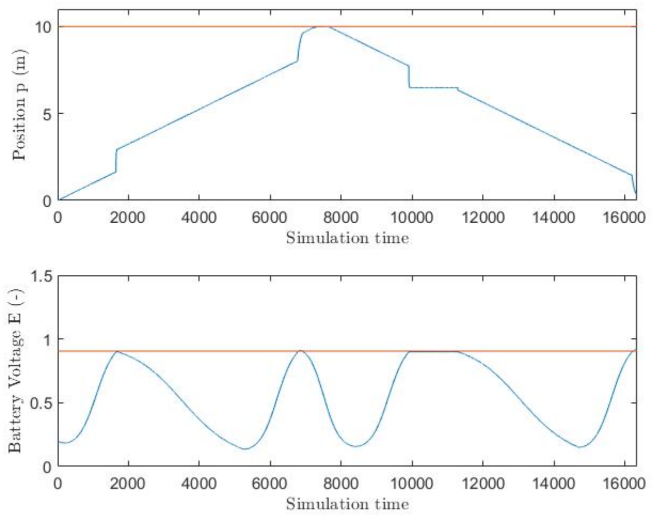


Figure 3: Variables de estado en simulación. Gráfico superior: Posición p. Gráfico inferior: Voltage de la Batería E

identificación del modelo de la intensidad lumínica puede observarse en Figure 4, Figure 5 y Figure 6.

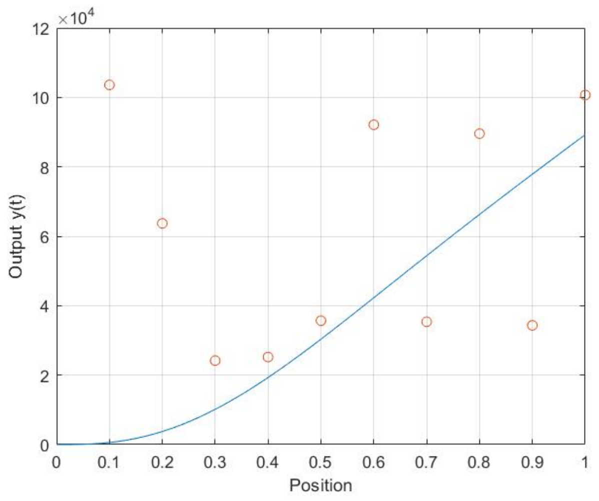


Figure 4: Waypoints y smoothing spline con 10 medidas

Debido a que este método require de calcular Gramianos y matrices inversas, cuando nuevos sets de datos son incluidos el tiempo de ejecución crece exponencialmente. Este factor muestra como una revisión del método que permita mantener las dimensiones del problema constante es necesaria. La solución propuesta consiste en la aplicación recursiva

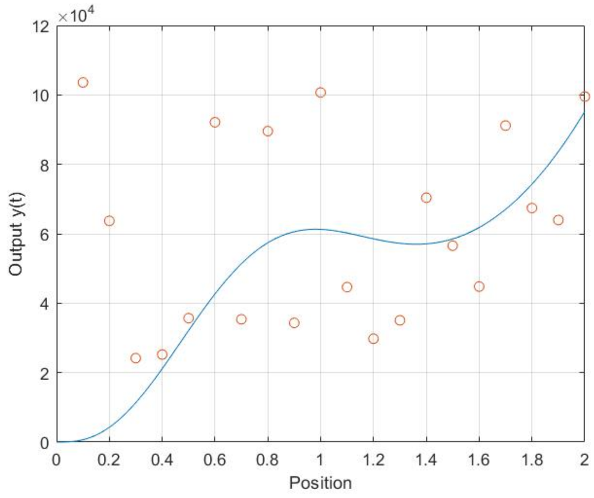


Figure 5: Waypoints y smoothing spline con 20 medidas

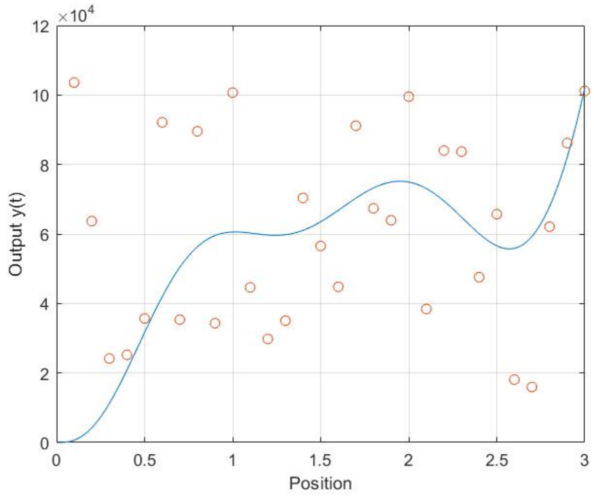


Figure 6: Waypoints y smoothing spline con 30 medidas

de los *Smoothing Splines*. Este nuevo enfoque utiliza cada nuevo set de datos para modificar la solución previa en lugar de resolver un nuevo problema. La terоría asi como los fundamentos matemáticos de la solución se encuentran incluidos en esta tesis.

Contribuciones

Las principales contribuciones de esta tesis son:

- Extender la teoría de la Barreras de Control para sistemas en forma no afín

- Un nuevo robot para monitorización medioambiental, más pequeño y más fácil de mantener y reproducir
- Aplicación de la teoría de *Smoothing Splines* para la identificación del modelo de la Intensidad Lumínica
- Teoría y fundamentos matemáticos para la aplicación recursiva de *Smoothing Splines*

Bibliografía

[1] Y. E. Gennaro Notomista and M. Egerstedt, "The SlothBot: A Novel Design for a Wire-Traversing Robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, 2019.
[2] M. B. Egerstedt and C. F. Martin, "Optimal trajectory planning and smoothing splines," Automatica, vol. 37, no. 7, 2001.

SUMMARY

Introduction

The pace of technological and industrial development, in addition to Human's impact on resource depletion, climate change or carbon dioxide footprint have lead our environment to a compromised situation. Researchers of all areas are making a collective effort to improve it before a non-returning point is reached. One of the most important ecosystems to analyze are tree canopies, as they are where solar energy becomes carbohydrates to fuel ecosystems, and house of more than fifty percent of our terrestrial biodiversity.

For an accurate representation of the surveilled tree-tops ecosystem, data needs to be gathered continuously and over large and non-stationary areas. This presents a challenge where robotics can constitute a valuable contribution. With this Master Thesis we aim to provide a robotic platform that will help in the characterization of tree canopies. This mission can be expressed as the satisfaction of two tasks: Environmental exploration and surveillance. These two objectives are particularly challenging because they need to be executed over time periods that exceed the robot's total battery capacity, i.e. they need to be *persistified*. Previous work on persistification of robotic tasks used control affine dynamics, but when a dependency between the robot's battery voltage and the control input is included, the dynamics are no longer control affine, and all previous theory needs to be reviewed.

Objectives

The objectives addressed with this Master Thesis have been:

1. Develop a new version of the robotic platform: The SlothBot

2. Derive Control Barrier Functions (CBFs) theory for systems with energy dependent dynamics

3. Implement new control algorithms based on CBFs for robotic's tasks persistification

4. Study and implement the application of Smoothing Splines for unknown scalar valued functions characterization, such as the Light Intensity

5. Stablish the mathematical foundations of Recursive Smoothing Splines, a method that aims to keep the problem size constant when new data sets are added

Proposed Solution

The proposed solution consists of a robotic platform for environmental exploration and surveillance as well as the theoretical mathematical derivations and control laws needed to achieve such objectives.

”The SlothBot” [1], shown in Figure 7, is the robot developed for tree-canopies exploration. Its motion is one dimensional, as it was deployed on a wire hanged between two trees. This version of the Slothbot was installed at the tree canopies of the Atlanta Botanical Garden for more than 13 months. During these months the robot was traversing the wire while taking environmental measurements. Some of these measurements are: air quality, light intensity, temperature, and humidity.

In order to ensure the robot's survival in an unknown and outdoors environment we need that both the battery voltage and the robot's position remain within safe values. Therefore, the system model had to include both the dynamics of the robot itself and also the impact of the environment on the robot. Doing so, the system model results:

$$\begin{pmatrix} \dot{p} \\ \dot{E} \end{pmatrix} = \begin{pmatrix} u \\ K\left(\frac{1}{1+\frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E\right) - \rho u^2 \end{pmatrix} \tag{3}$$

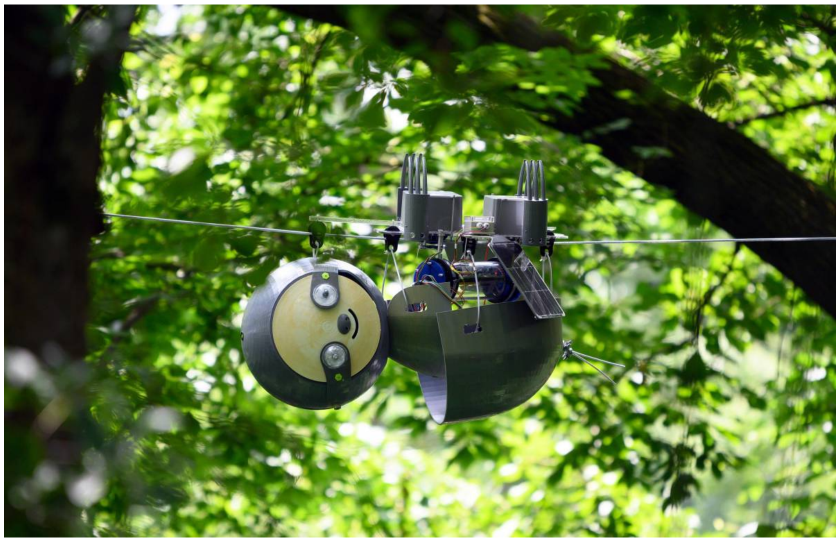


Figure 7: SlothBot at the Atlanta Botanical Garden

In Equation 3 it can be seen how when a dependency of the robot's battery voltage on the control input is included, a quadratic term appears in the battery's dynamic equation. Hence, the system is no longer control affine.

The problem of satisfying the robot's mission while ensuring its survival is formulated as minimizing the squared norm of the difference between the nominal control $\hat{u}$, that encodes the robots mission, and the safe control u. The safe control u satisfies the two constraints that guarantee the system remains forward invariant. These constraints are indeed Control Barrier Functions.

Expressing these objectives in mathematical terms, the optimization problem shown in Equation 3.16 is obtained:

$$\begin{aligned}
&\underset{u}{minimize} \quad ||u - \hat{u}||^2 \\
&s.t. \\
&\quad - L_g h_1(X) u^2 \leq \gamma_1 h_1(X) + L_f h_1(X) \\
&\quad - L_g h_2(X) u \leq \gamma_2 h_1(X) + L_f h_2(X)
\end{aligned} \tag{4}$$

Where:

- $X = [E \quad p]$ is the state vector
- $L_f h_i, L_g h_i$ are the Lie Derivatives of the barrier function h_i.

The light intensity function $I(p)$ plays a fundamental role in the battery dynamics presented in Equation 3. Having an accurate representation of such function would allow the robot to perform more ambitious exploration missions, as it will be known in advance which areas help charging its battery, and which ones deplete it.

Smoothing Splines are a model fitting technique based on classical control theory. Its goal is to find the control input that drives the output of an arbitrary system to a set of waypoints at certain interpolation time instants. In order to choose the control input, a minimization problem on the control energy is solved. The resulting output when applying this controller is a Smoothing Spline. Measurements are not considered to be perfect and due to this fact, the Smoothing Splines method does not aim to obtain a perfect fit, but a model that overall summarizes all data points available. This approach leads to a less computational expensive, and thus fastest, solution to the optimization problem.

The mathematical derivation, based on the published work by Egerstedt and Clide [2] can be found in the thesis.

Results

Solving the problem in Equation 4 can be approached as a Quadratically Constrained Quadratic Program (QCQP), for which complex convex optimization algorithms exist. A drawback of all these algorithms is their high computational needs, and thus a high energy consumption, what makes them incompatible with the longevity goal of this project. For the SlothBot application, due to its one dimensional dynamics, there is only one control input available, resulting in a problem with a scalar objective function and scalar nested constraints. We denote this new class of problems Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problems. The fact that this is a scalar problem allows for an analytical solution, which requires less computational resources compared to all the mentioned alternatives. This solution proves that the robot would not end up in a situation in which the battery can not recharge or discharge, under regular operation conditions.

This analytical solution was implemented in a simulated SlothBot Figure 8. During this simulation, the robot starts its at position $p = 0$ and with a 20% of its battery, i.e. $E = 0.2$, and traverses the wire forward and backwards. Figure 9 shows how the CBFs modify the speed and thus the position of the robot to prevent its battery from depleting or overcharge. A more in depth analysis of the presented results can be found in this thesis.

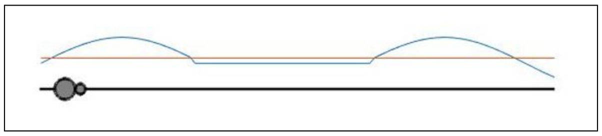


Figure 8: Light intensity function $I(p)$ (blue), light intensity threshold I_c (orange), and simulated SlothBot

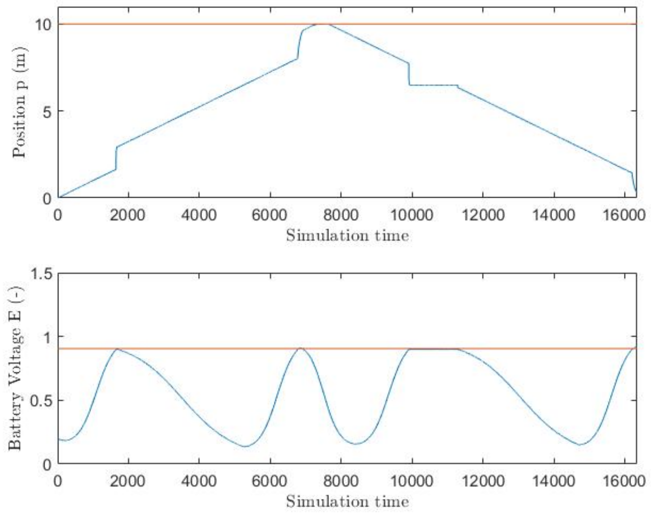


Figure 9: State variables when traversing the wire forward and backwards. Upper plot: Position p. Lower plot: Battery Voltage E

The results obtained when using Smoothing Splines to characterize the light intensity model on the robot's wire are shown in Figure 10, Figure 11, and Figure 12.

The problem is that when new data sets are added, the computation of the Gramian and

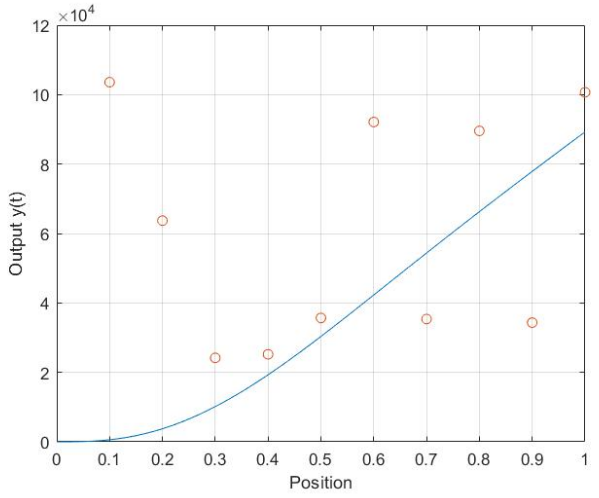


Figure 10: Waypoints and smoothing spline with 10 samples

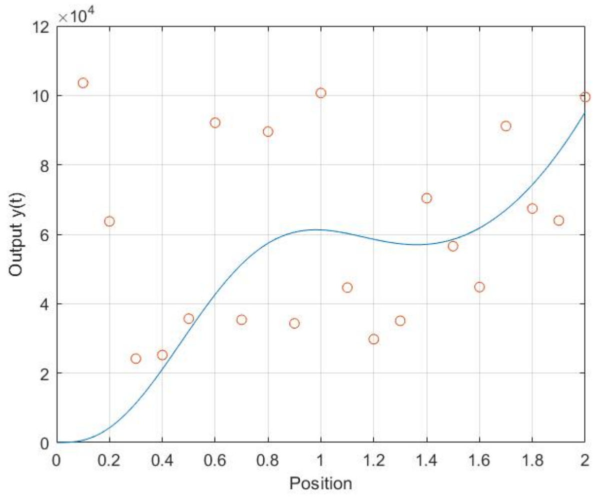


Figure 11: Waypoints and smoothing spline with 20 samples

the matrices inverse make the execution time grow exponentially. This fact shows the need for a revision of the method that would help to maintain the size of the problem fixed. The proposed solution consists of Recursive Smoothing Splines, that use each new data set to modify the previous spline instead of solving a new problem. The theory and mathematical derivation of this method can be found in the thesis.

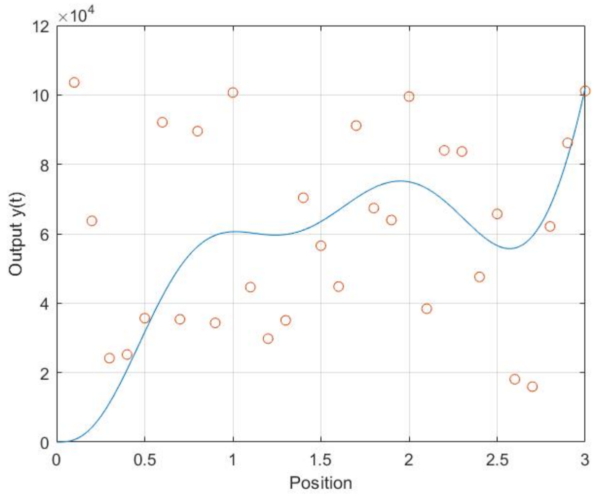


Figure 12: Waypoints and smoothing spline with 30 samples

Contributions

The main contributions of this Master Thesis are:

- Extension of Control Barrier Functions theory for systems with no control-affine dynamics
- New, smaller, and easier to build and maintain robotic platform for environmental exploration and surveillance
- Smoothing Splines approach for Light Intensity model identification
- Recursive Smoothing Splines state of art theory and mathematical foundations

References

[1] Y. E. Gennaro Notomista and M. Egerstedt, "The SlothBot: A Novel Design for a Wire-Traversing Robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, 2019.

[2] M. B. Egerstedt and C. F. Martin, "Optimal trajectory planning and smoothing splines," Automatica, vol. 37, no. 7, 2001.

ACKNOWLEDGMENTS

Six years and four months after I entered Alberto Aguilera 23 for the first time, my engineering degree at Comillas Pontifical University finally comes to an end. Seventeen years old me would have never imagined that she would be writing her Master Thesis four thousands miles away from home, living in Midtown Atlanta, while starting her Ph.D. program at one of the most prestigious engineering institutions in United States, Georgia Institute of Technology. This whole journey wouldn't have been possible without the love and encouragement of all the great people I had by my side during these years.

I need to start by thanking my parents, who have always been my main support during my entire life. Thanks for not letting me give up when I was feeling defeated, and thanks for your endless patience when I was overwhelmed...

Special thanks to my Master Thesis director and Ph.D. co-advisor Dr. Magnus Egerstedt for giving me the opportunity of being part of his research. I am very grateful for the opportunity and your trust to help me promote from my original M.S. program to the Ph.D., helping me getting closer to my dreamed career as a future controls researcher and faculty member. Working these past months with you has been an honor, and I have no doubt that University of California Irvine is going to incredibly benefit from your talent.

Thanks to all the professors from Comillas Pontifical University, that have been fully supportive in this process. I appreciate every recommendation letter, supportive emails... Thanks to Dr. Juan Carlos del Real and Alberto Zanmatti for your patience and help figuring out all the academic logistics that have made this possible. I promise I will do my best in these next years to make our institution proud.

They say friends are the family you choose, and in that sense I have the most amazing family one could ever dream of. I feel extremely grateful that, even tough it has already been ten months since I left Spain, I can still feel all of you very close. Thanks for always be there to celebrate my achievements and share the good things that are happening in your lives. You all are one of my biggest inspirations and motivations to keep working harder and trying to become better, both as a person and professionally, every single day.

And what can I say about Atlanta... Thanks for treating me so well this first year. I still can't believe the amount of great moments I have lived, and all the amazing people I have met, some of which have become my closest friends after just a few months... Thanks to all Crossfit Midtown members for welcoming me with open arms and making me feel at home. Special shout out to our headcoach and owner Mike T. Aaron, you are such an inspiring mentor and have built a wonderful community. I couldn't be happier of being now a small part of it as well.

I cannot conclude this acknowledgments without thanking my Ph.D. advisor Dr. Sam Coogan for receiving me in your lab to continue with my Ph.D. program, I can't wait for all the exciting projects to come in these next years.

Once again, thank y'all... I promise I'll make you proud.

TABLE OF CONTENTS

Acknowledgments xix

List of Tables xxv

List of Figures xxvii

Chapter 1: Introduction and Background 1

1.1 Motivation 1

1.2 Start of Art 3

1.3 Previous work 6

Chapter 2: Hardware and Software resources 7

2.1 Sensors, actuators and microcontrollers 7

2.1.1 Sensors 7

2.1.2 Actuators 9

2.1.3 Microcontrollers 10

2.2 Software tools 11

2.3 Mechanical Design 12

Chapter 3: Control Framework 13

3.1 System model 14

3.2 Control Barrier Functions 16

3.2.1 Control Barrier Functions fundamentals 16

3.2.2 Higher Degree Control Barrier Functions 17

3.2.3 First approach - Second order relative degree CBFs 18

3.2.4 Second approach - Candidate CBFs for energy dependent dynamics 20

3.3 Optimization problem statement 21

3.4 Analytical solution to the general Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problem 22

3.4.1 Parameterized minimization problem 23

3.4.2 Analysis of parameterized nested quadratic and linear constraints . . 24

3.4.3 Constraints values for the SlothBot dynamics 28

3.5 Simulation results of the analytic solution for QCSLQC problem in the SlothBot application 33

Chapter 4: Smoothing Splines for Light Intensity model estimation 35

4.1 Smoothing Splines 35

4.1.1 Theoretical derivation 36

4.1.2 Implementation of the algorithm 38

4.1.3 Application to Light Intensity Model Fitting 41

4.1.4 Simulation results for the SlothBot application 42

4.2 Recursive Splines 45

4.2.1 Problem Statement 45

4.2.2 Projections 46

4.2.3 Two-point boundary problem 48

4.2.4 Application to a dynamical system 51

4.2.5 Recursive Splines - Continuous Case 54

4.2.6 Recursive Splines - Discrete Case 59

Chapter 5: Conclusions, contributions and future work 67

5.1 Conclusions and Main Results 67

5.2 Contributions 69

5.3 Future Work 71

Appendices 72

Appendix A: Lie Derivatives 73

Appendix B: Lipschitz Continuity 75

Appendix C: Alignment with Sustainable Development Goals (SDGs) 76

References 78

LIST OF TABLES

4.1 Illuminance values for different outdoor conditions 43

LIST OF FIGURES

1 SlothBot en el Jardín Botánico de Atlanta v

2 Intensidad Lumínica $I(p)$ (azul), Intensidad Lumínica umbral I_c (naranja), y SlothBot simulado vii

3 Variables de estado en simulación. Gráfico superior: Posición p. Gráfico inferior: Voltage de la Batería E viii

4 Waypoints y smoothing spline con 10 medidas viii

5 Waypoints y smoothing spline con 20 medidas ix

6 Waypoints y smoothing spline con 30 medidas ix

7 SlothBot at the Atlanta Botanical Garden xiii

8 Light intensity function $I(p)$ (blue), light intensity threshold I_c (orange), and simulated SlothBot xv

9 State variables when traversing the wire forward and backwards. Upper plot: Position p. Lower plot: Battery Voltage E xv

10 Waypoints and smoothing spline with 10 samples xvi

11 Waypoints and smoothing spline with 20 samples xvi

12 Waypoints and smoothing spline with 30 samples xvii

1.1 SlothBot at the Atlanta Botanical Garden 2

1.2 SlothBot 2.0 new prototype at GRITS Lab 3

2.1 Humidity, temperature and altitude sensor BME 680 7

2.2 Current, and Power sensor INA219 8

2.3 Air Quality sensor SGP30 8

2.4 Light luminosity Lux sensor TSL2591 8

2.5 Onboard Voltage Regulators S18V20F6 9

2.6 Lidar sensor Stemadu TFMini Plus 9

2.7 Motors 499:1 25Dx73L mm LP 6V with 48 CPR Encoder 10

2.8 Motor Drivers TB9051FTG 10

2.9 Raspberry Pi Zero W 10

2.10 Teensy 3.2. 11

2.11 CAD model of the original SlothBot 12

2.12 CAD model of the new SlothBot design 12

3.1 Representation of a positively invariant set from [17] 17

3.2 Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 1$ 29

3.3 Terms α (Blue) and β (Orange) of the linear constraint 30

3.4 Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 10$ 31

3.5 Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 100$ 32

3.6 Light intensity function $I(p)$ (blue), light intensity threshold I_c (orange), and simulated SlothBot 33

3.7 State variables when traversing the wire forward and backwards. Upper plot: Position p. Lower plot: Battery Voltage E 34

4.1 Waypoints and smoothing spline with different smoothing parameter ρ: 1/1000000 (Blue), 1/100000 (Yellow), and 2/100000 (Purple) 39

4.2 Waypoints and smoothing spline with different smoothing parameter ρ: 1/1000000 (Blue), 1/100000 (Yellow), and 2/100000 (Purple) 40

4.3 Waypoints and smoothing spline with $\rho = 1/1000000$ for system 1 (Blue), and system 2 (Yellow) 41

4.4 Waypoints and smoothing spline with 10 samples 42

4.5 Waypoints and smoothing spline with 20 samples 42

4.6 Waypoints and smoothing spline with 30 samples 43

4.7 SlothBot simulator interface. The blue curve represents the real light intensity and the red line the value for I_c 44

4.8 Real light intensity model (Yellow), samples gathered by the SlothBot (Orange circles), and model obtained with smoothing splines (Blue) 44

4.9 Projection: Geometric construction 47

4.10 V_ξ, V_0, and $V_0^\perp$ representation 49

CHAPTER 1
INTRODUCTION AND BACKGROUND

This first chapter introduces how robotics can contribute to environmental research and why they constitute a revolutionary tool in ecosystems characterization above ground level, section 1.1. Also, we include a review of the state of art, section 1.2, for both environmental applications of robotics, and also the most used tools for model identification nowadays. Finally, section 1.3 introduces the state of art for SlothBot project itself, the use of Control Barrier Functions, and Smoothing Splines, our proposed techniques for identifying the unknown and scalar-valued light intensity function.

1.1 Motivation

Nowadays, researchers of all technical areas are concerned about our environment compromised situation, and robotics can be a valuable contribution to the collective effort of improving this situation. Ecologists recognize that there is a lack of knowledge about all phenomena occurring on tree canopies and that they need means of gathering data above ground level [3] [4]. Tree canopies are profoundly important to global climate regulation and forest microclimates. They are also where solar energy becomes carbohydrates to fuel entire ecosystems, and they are the house of more than fifty percent of our terrestrial biodiversity [5].

Using robots reduces the risk for researchers, as at this moment most data gathering methods from treetops require a human presence, using ladder structures or construction cranes. Thanks to robotics, this situation improves drastically. Researchers would only have to analyze the information collected by the robot, without having to expose themselves to such risky heights.

Our contribution to the characterization of ecosystems above ground level can be expressed as completing two tasks: environmental exploration and environmental surveillance. One of the most challenging aspects of these two tasks is that both require to be executed over a long period of time, exceeding the robot's battery capacity. They need to be *persistified*. Nowadays, longevity is one of the most compromising factors when deploying any robotic system, specially in a natural and unknown territory.

Exploration implies a long execution time because of the size and dynamic nature of the environment, and surveillance is subjected to the duration of the observed phenomena. The use of mobile sensors, i.e. robots, has the advantage of improving coverage and data gathering compared to standard static sensors. However, mobile sensors imply a higher power consumption, which again compromises the longevity of the system outdoors.

The robotic platform that has been developed for this environmental mission can be found in Figure 1.1. It is named the SlothBot, and it is indeed a robotic sloth. This robot has been hanged at the Atlanta Botanical Garden for more than a year, until July 2021, gathering air quality, light intensity and pollution measurements. As part of this Master Thesis, a new, more compact and efficient version was designed and tested, Figure 1.2. Originally, the first version of the SlothBot was developed by Gennaro Notomista, Yousef Emam and Dr. Magnus Egerstedt [1] in 2019.

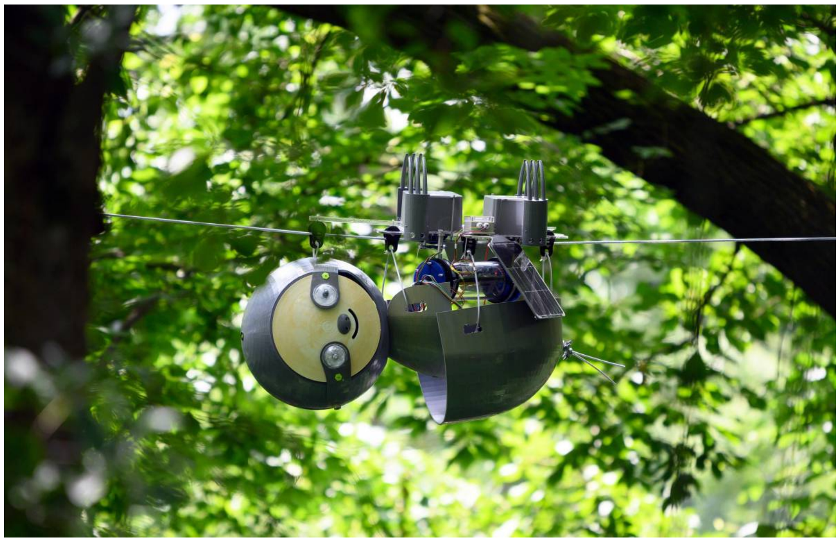


Figure 1.1: SlothBot at the Atlanta Botanical Garden

More information about the SlothBot and its mission at the Atlanta Botanical Garden can be found in Georgia Tech School of Electrical Enginering webpage:

- https://coe.gatech.edu/news/2021/07/year-alone-trees
- https://www.ece.gatech.edu/news/636291/slothbot-garden-demonstrates-hyper-efficient-conservation-robot
- https://www.ece.gatech.edu/news/622103/slothbot-takes-leisurely-approach-environmental-monitoring

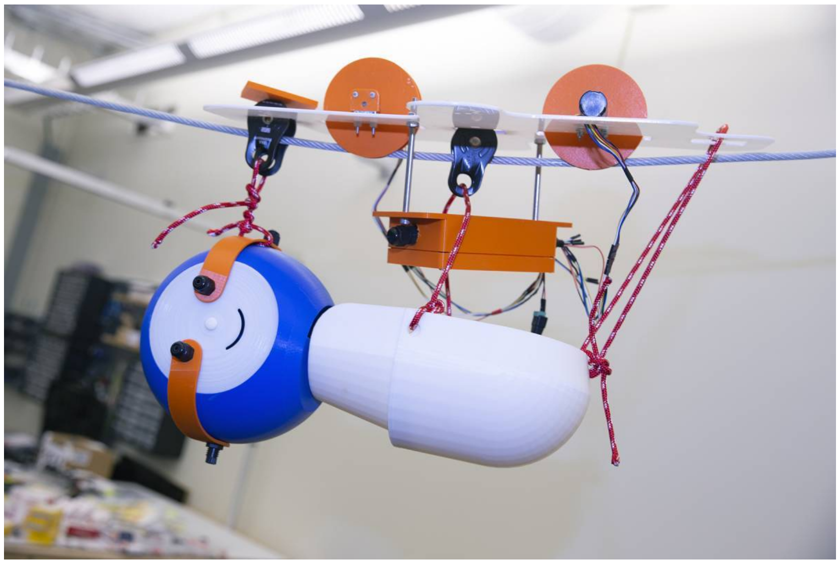


Figure 1.2: SlothBot 2.0 new prototype at GRITS Lab

Having a smaller and easier to build robotic platform allows to build numerous Slothbots and deploy them in different areas of an ecosystem, so they can complete their mission in a collaborative way. Using multi-robot systems would solve the collecting-data problem optimally and broadens the current applications of swarm robotics.

1.2 Start of Art

Robotics have started to play fundamental roles, both in industry and our daily lives, but the environmental application of robotics is still unexplored, arising new problems for both ecologists and the controls and robotics community.

When used in outdoors environment, robotics have been mostly used to solve the data coverage and collecting problem optimally. One approach consists on using Networked Infomechanical Systems (NIMS) to explore the environment [6]. In combination to adaptive sampling techniques, the error between the real environment and the reconstructed from the data points can be minimized. Specifically NIMS were developed and used for spatiotemporal variation of atmospheric phenomena monitorization.

One of the advantages of using robotics is their versatility. Using the adequate hardware they can be deployed in all kind of different ecosystems. For example, we can find robotics applications in oceanographic and marine applications. In [7] the problem of studying non-stationary and multiple spatial phenomena is addressed. In order to obtain an estimate of the distribution of the data, space-time points samples can be collected, and assuming that the observed process is stationary during the time intervals between the samples, the actual behavior can be reconstructed from them. Alternatively, robotics can be used [8] as a solution to the large amounts of data that needs to be gathered for marine monitoring. This application consists on the use of swarm robotics, that can adapt to unknown and dynamic environments thanks to the use of decentralized controllers, onboard sensing and local communications.

In order to ensure the longevity of the SlothBot, it needs to use its solar charger and panels to recharge its battery. Due to this fact, we need to characterize the light intensity on the wire where it is deployed. This light intensity function consists of an unknown scalar valued and time invariant function, which shape and parameters need to be identified. Recently, Machine Learning and Artificial Intelligence techniques have become the most popular approach for model identification. Specifically, in robotics some of the most used methods are Split-and-merge, the Hough transform or the Random Sampling Consensus.

Split-and-merge [9] consists on building maps or models using line segments. The advantage of using lines is that it is the simplest geometric construction, it is compact and can represent most of the indoors environments.

A possible alternative is to use the Hough transform [10], that consists in finding the parameter set which represents the most data-points. It is based on an accumulator over the parameter space, and it can fit not only lines, but also circles or any other curve that we initialize and parameterize. This advantage constitutes as well its major drawback, as we are presetting a particular shape for our data beforehand. It is a very robust model identifying technique, as it has almost no sensitivity to outliers, but it is only computationally efficient for a very low number of parameters, as its memory requirements grows exponentially with both data and the dimensions of the parameter space, that indeed needs to be bounded.

A more efficient algorithm to deal with higher numbers of parameters or unbounded parameters sets is the Random Sampling Consensus, or more commonly known as RANSAC [11]. Its main limitation is that it only provides a probabilistic guarantee of the fit. RANSAC's objective is to estimate a model with p parameters using n data points, by using a subset of p consisting of q data points $(q << n)$ it estimates a closed form of the p parameter.

From a controls approach, we can use an adaptive control [12] to identify the system parameters if we are using a data driven or *black-box* model, where the error between the real system output and the simulated with the estimated model is feedback to the system and the parameters modified until this error is equal to zero.

The main drawback of all the model identification methods introduced is that they are too expensive in terms of computation requirements and thus energy consumption. As we want to ensure the longevity of the system, we need to use the available energy in a conservative way, which is incompatible with the computation complexity needed for all these methods. Secondly, as the data size grows, the computation time increases exponentially, again making all these algorithms unsuitable for the SlothBot application.

1.3 Previous work

Any of the presented papers in section 1.2 addresses the increased power consumption due to the use of non standard static sensors, so obliquely these missions will be executed over short time periods by the robots. The aim of this project is to design a control law that allows our robot to succeed on its surveillance and exploration missions, and also continue to perform them over a time period that exceeds its battery capacity.

Previous students from the Georgia Robotics and InTelligent Systems laboratory (GRITS lab) already started studying the possibility of using Control Barrier Functions as a solution to the environmental monitorization and exploration problems [13]. In this project, we aim to continue developing the already started theory proposition and implement it on both simulation and the designed robotic platform.

The use of Control Barrier Functions is becoming popular in safety applications [14] [15] [16]. The survival of a robot in an unknown and dynamic environment can be also expressed in terms of ensuring that the state variables of the robot remain within the safe set, i.e. they belong to a forward invariant set. Due to this fact, the application of CBFs seems to be the perfect fit for the persistification problem.

To overcome the model fitting data-size challenge, we propose to use smoothing splines based on Egerstedt and Martin work in [2], and we further expand its application to introduce the concept of recursiveness, so that with any new set of data points the current solution is modified without increasing the number of measurements or waypoints used to fit the output curve.

CHAPTER 2
HARDWARE AND SOFTWARE RESOURCES

Chapter two covers all components of the SlothBot and their functionality in the robot. First, section 2.1 includes all the hardware components of the SlothBot: sensors, actuators and microcontrollers, as well as a small description of each of them. Then, section 2.2 summarizes the software tools that have been used to program the robot. Finally, section 2.3 includes some captions from the original SlothBot CADs and the new version that has been designed.

2.1 Sensors, actuators and microcontrollers

2.1.1 Sensors

The SlothBot mission is environmental exploration and surveillance, due to this it is equipped with numerous sensors that enable him to gather all helpful data for ecologists.

Humidity, temperature and altitude: BME 680

This system provides the majority of the environmental information of interest: temperature, humidity, barometric pressure and Volatile Organic Compound (VOC) gases.

Figure 2.1: Humidity, temperature and altitude sensor BME 680

Current, and Power: INA219

The INA219 plays a fundamental role for monitoring the power consumption of the robot, as it allows us to measure the current that it is being drawn by the robot at all time.

This is one of the most important values to track, as it will determine the expected battery life of the robot.

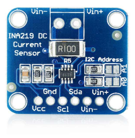


Figure 2.2: Current, and Power sensor INA219

<u>Air Quality: SGP30</u>

In combination with the BME680 environmental sensor, the SGP30 air quality sensor is used to obtain more precise air quality data and also carbon dioxide (CO2) readings.

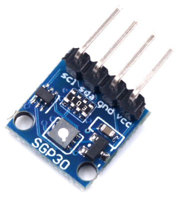


Figure 2.3: Air Quality sensor SGP30

<u>Light luminosity Lux: TSL2591</u>

As we will see in chapter 3 , light intensity is one of the main factors that influences our system dynamics. It determines whether the battery charges or discharges at each location on the wire. Thanks to the TSL2591 an accurate measure that ranges from 188 μLux to 88,000 Lux can be obtained.

Figure 2.4: Light luminosity Lux sensor TSL2591

<u>Onboard Voltage Regulators: S18V20F6</u>

One of the emerging problems of having such a wide variety of sensors and actuators is that they operate with different input voltages. To solve this issue, a 6V Step-Up/Step-Down voltage regulator is used. This voltage regulator S18V20F6 provides a constant 6V output with any voltage input between 3V and 30V.

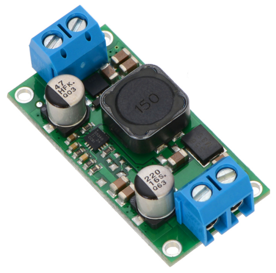


Figure 2.5: Onboard Voltage Regulators S18V20F6

Lidar: Stemadu TFMini Plus

Finally, the last sensor is a Lidar that allows to measure the distance from the robot to the end of the wire, i.e. the distance to the next tree. With this measure and in combination with the encoder count it is possible to have an accurate localization of the robot at all time.

Figure 2.6: Lidar sensor Stemadu TFMini Plus

2.1.2 Actuators

Motors: 499:1 25Dx73L mm LP 6V with 48 CPR Encoder

In the current version of the SlothBot the motors used are a pair of 6V brushed DC motors that also include an integrated quadrature encoder. These encoders are used to control both the speed and the position of the robot during its exploration mission.

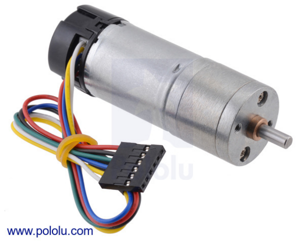


Figure 2.7: Motors 499:1 25Dx73L mm LP 6V with 48 CPR Encoder

<u>Motor Drivers: TB9051FTG</u>

In order to control the motors, we use the Toshiba's TB9051FTG motor drivers.

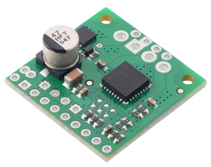


Figure 2.8: Motor Drivers TB9051FTG

2.1.3 Microcontrollers

The microcontrollers used have been a Raspberry Pi Zero W, and a Teensy 3.2.

A Raspberry Pi Figure 2.9 is commonly known and used as a microcontroller but it is indeed a computer, as it possesses a CPU, GPU, DSP, SDRAM and USB ports. Specifically the Zero W model includes 1GHz, single-core CPU, and 512 MB RAM. Thanks to all its features, it is used for the high level tasks, such as the control algorithms and the data processing.

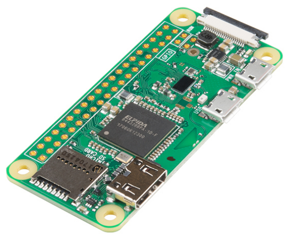


Figure 2.9: Raspberry Pi Zero W

On the other hand, the Teensy 3.2. Figure 2.10 acts as a low level microcontroller, executing simple tasks as writing the actuators and reading the sensor measurements, following the commands provided by the Raspberry Pi.

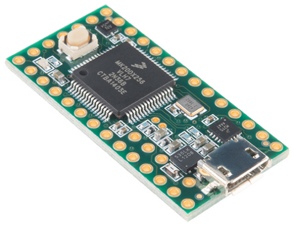


Figure 2.10: Teensy 3.2.

2.2 Software tools

Due to the broad extension of the scope of this project, several software tools were needed, being the most representative:

- Solid Edge: For hardware modeling and structure analysis
- Eagle: Needed to design the Printed Control Board (PCB), main electronic circuit of the robot
- Matlab: For designing the Control and light intensity model fitting algorithms, in addition to their respective tests in simulation
- Arduino IDE: With the Teensyduino Add-on to program our low level microcontroller
- C++: Used to initialize and program all drivers for the sensors, communications and high level tasks for the Raspberry Pi.

2.3 Mechanical Design

One of the most urgent goals to complete during the Spring semester for the project was to come up with a new design for the robot. We needed a robotic platform that was smaller, easy to reproduce and most importantly, easier to open for maintenance routines checks.

The original SlothBot had a volume of $6552364.46mm^3$ including the robot and the waterproof tube, and a head to tail length of $956.54mm$. This measurements were reduced to a total of $3205735.79mm^3$ for the volume of the robot and the waterproof sensors cage, and a head to tail length of $372.11mm$. This is more than a 50% reduction, which clearly shows than this objective of the project was successfully achieved.

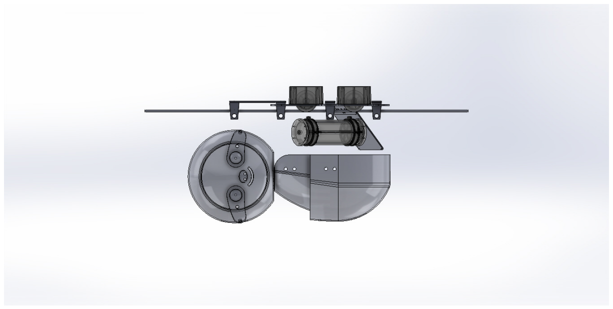


Figure 2.11: CAD model of the original SlothBot

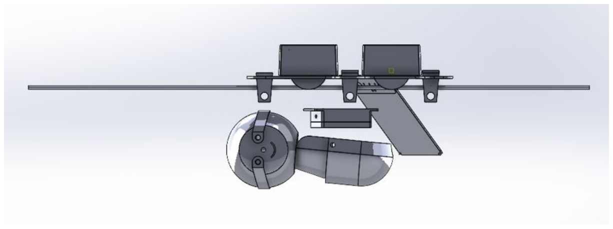


Figure 2.12: CAD model of the new SlothBot design

CHAPTER 3
CONTROL FRAMEWORK

Chapter three describes the system dynamics model, introduces the background theory of Control Barrier Functions (CBFs), and concludes with how a control strategy based on CBFs was implemented to solve the persistification problem. The proposed model includes both the robot dynamics and the environment, as the robot needs to be able to complete its mission while surviving in an unknown environment. A control algorithm inspired in Control Barrier Functions was used. This technique ensures that our system state variables remain within the boundaries of their respective forward invariant safe sets. The minimization problem that results has a quadratic cost and two nested scalar constraints: one of which is quadratic and the other is linear. We denote this new class of problems as Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problems.

Finally, in this chapter an analytical solution to the general QCSLQC problem is provided as well as a successful implementation of the solution for a SlothBot simulator.

This chapter is organized as follows: section 3.1 presents the system dynamics that have been used to model the behavior of our robot and the effect of the environment. Then, section 3.2 covers the control strategy used in the project, inspired by Control Barrier Functions theory. This section includes an overview of the theoretical background as well as the explicit application to the SlothBot. When building the optimization problem to solve in section 3.3, the reader can observe that this problem does not belong to any of the known convex optimization programs, and that is why in section 3.4 we include a in-depth analysis of the existence of solutions for the problem, and thus conclude with a closed form solution. Finally, section 3.5 presents the simulation results of the analytical solution for the QCSLQC problem in the SlothBot application.

3.1 System model

As it has been expressed in chapter 1, one of the main objectives of this Master Thesis is to find a control law that ensures the survival of the robot in a harsh and unknown outdoor environment. We understand surviving as both the position of the robot and its battery voltage level to remain within safe boundaries. In order to achieve this goal, we need to include the effect of the environment in our system model.

As a first approach to solve this problem, we used the model proposed in previous work [13]:

$$\begin{pmatrix} \dot{p} \\ \dot{E} \end{pmatrix} = \begin{pmatrix} u \\ K\left(\frac{1}{1+\frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E\right) \end{pmatrix} \tag{3.1}$$

As it will be seen in subsection 3.2.3, using this model requires us to program a second order relative degree Control Barrier Function, and furthermore, this second relative degree is ill-defined, leading to an infeasible problem whenever the derivative of $I(p)$ becomes equal to 0, i.e. at a maximum, minimum or if $I(p)$ is constant. Also, it does not include the direct dependency between the battery voltage and the control input.

In order to ensure the longevity of the system, a conservative use of the available energy is needed. To do so, a direct relation between the robot's battery voltage and the control input has to be included. Doing so the new system model results to be:

$$\begin{pmatrix} \dot{p} \\ \dot{E} \end{pmatrix} = \begin{pmatrix} u \\ K\left(\frac{1}{1+\frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E\right) - \rho u^2 \end{pmatrix} \tag{3.2}$$

The main elements that constitute the state equations of Equation 3.2 are:

- $p \in [0, 10]$: State variable for robot's position on the cable.
- $E \in [0, 1]$: State variable for robot's unitary battery voltage. This variable has been

normalized so it is independent of the batteries used in the actual robot (6 V or 12 V).

- $I : \varepsilon \, \mathrm{x} \, \mathbb{R}_+ \rightarrow \boldsymbol{I} \subset \mathbb{R}_+[0,1]$: Unknown time invariant scalar field representing the solar light intensity at robot's position p.

- $I_c \in (0,1)$: Solar light intensity threshold. It expresses the value for the solar light intensity that makes the battery charge or discharge, as it will be explained in sub-subsection 3.1. Its value depends on the robot's electronics power consumption.

- $K, \lambda, \rho \in \mathbb{R}_+$: Are model parameters to be fitted by experiments on the actual robot.

Battery Voltage Dynamics

As it is expressed on Equation 3.2, the rate of change of the robot's unitary battery voltage due to the environment, and without any control input, is expressed as:

$$\dot{E} = K\left(\frac{1}{1 + \frac{1-E}{E} e^{-\lambda(I(p)-I_c)}} - E\right) \tag{3.3}$$

This equation yields to three possible scenarios:

1. $I(p) > I_c \rightarrow \dot{E} > 0$: If the light intensity at the robot's current position is greater than the threshold, the battery charges.

2. $I(p) < I_c \rightarrow \dot{E} < 0$: If the light intensity at the robot's current position is less than the threshold, the battery discharges.

3. $I(p) = I_c \rightarrow \dot{E} = 0$: If the light intensity at the robot's current position is equal to the threshold, the battery voltage remains constant.

For this analysis, the control input is assumed to be equal to 0. In order to guarantee that the robot is able to perform its mission over a long time period, the control energy will be used in a restrictive way. In fact, the control input will only be different to zero if the robot needs to move to charge its battery or if it needs to gather new data samples. Based on these statements, setting the control input equal to 0 is not a big assumption.

3.2 Control Barrier Functions

3.2.1 Control Barrier Functions fundamentals

Control Barrier Functions have become in the recent years one of the most promising control strategies, specially for safety applications [14] [15] [16], as they guarantee that the resulting controller will make the system remain within the specified safe region.

Mathematically speaking, if we consider a control-affine system:

$$\dot{x} = f(x) + g(x)u \tag{3.4}$$

and a set: $C = x : h(x) \geq 0$. Implementing the control barrier function $h(x)$ will output a controller $u(x)$ that makes C positively invariant. A set $C \subset \mathbb{R}^n$ is positively or forward invariant if for any initial condition $x_0 \in C$, $\phi(t, x_0) \in C$ for all $t \geq 0$, [17]. $\phi(t, x_0)$ is a trajectory of $\dot{x} = f(x)$ with $x(0) = x_0$ as an initial condition.

As it can be seen in Figure 3.1, the vector field needs to be pointing inwards on the boundary for C being positively invariant:

$$f(x) \cdot n(x) \leq 0 \tag{3.5}$$

In order words, being positively invariant means that if the system starts within the set, it will stay inside the set.

A function $h(x)$ with $C = \{x|\, h(x) \geq 0\}$ is a *control barrier function* for our system Equation 3.4, if there exists a *Locally Lipschitz* function:

$$\alpha : \mathbb{R} \to \mathbb{R}, \quad \alpha(0) = 0 \tag{3.6}$$

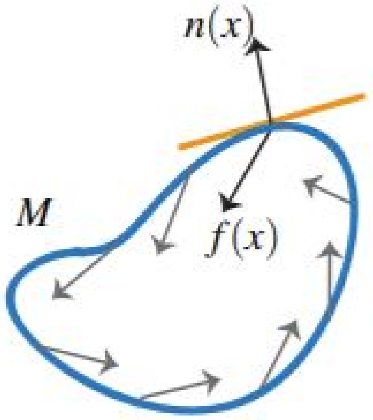


Figure 3.1: Representation of a positively invariant set from [17]

Satisfying:

$$\underset{u\in\mathbb{R}^m}{sup}\, L_f h(x) + L_g h(x)u \geq -\alpha(h(x)) \tag{3.7}$$

$L_f h(x)$ and $L_g h(x)$ express the *Lie Derivatives* of $h(x)$. We refer the reader to Appendix A for background theory about their computation.

The set of control inputs that satisfy the above statement $U(x)$ can be defined as follows:

$$U(x) = \{u \in \mathbb{R}^m | \bigtriangledown h(x)^T(f(x) + g(x)u) \geq -\alpha(h(x))\} \tag{3.8}$$

It holds that:

1. $U(x) \neq 0 \quad \forall x$

2. Any Lipschitz feedback control $u : \mathbb{R}^n \to \mathbb{R}^m$ satisfying $u(x) \in U(x)$ renders C invariant

For a more in depth explanation of Lipschitz continuity, we refer the reader to Appendix B.

3.2.2 Higher Degree Control Barrier Functions

In this section we will present the fundamental concepts of higher order CBFs, as the first approach to solve the persistification problem consisted in using a second order CBF. The derivation of this control strategy is covered in subsection 3.2.3.

If the chosen CBF $h(x)$ results in the term $L_g h(x) \equiv 0$, i.e. the control input vanishes from Equation 3.7, then $h(x)$ cannot be a CBF for the system. There are two alternatives:

- Try to find a new CBF
- Use a higher order degree CBF

Finding valid Barrier functions is closer to an art work than a mathematical procedure, and depending on the system it can be a very challenging task. Due to that fact, it is normally preferred to use a higher order CBF, even though this shrinks the area for the existence of solutions to the optimization problem. In order to build our higher order CBF, we start defining:

$$\Psi_1(x) = \bigtriangledown h(x)^T f(x) + \alpha_1(h(x)) \tag{3.9}$$

For some Lipschitz α_1, satisfying $\alpha(0) = 0$, and let:

$$C_1 = \{x| \, \Psi_1(x) \geq 0\} \tag{3.10}$$

If $u(x)$ is a feedback control so that C_1 is invariant, then $C \cap C_1$ is also invariant, where $C = \{x : h(x) \geq 0\}$. To ensure that C_1 is invariant we just need to use $\Psi_1(x)$ as our Control Barrier Function. If $\bigtriangledown \Psi_1(x)^T g(x) \equiv 0$, then we repeat the process, defining:

$$\Psi_2(x) = \bigtriangledown \Psi_1(x)^T f(x) + \alpha_2(\Psi_1(x)) \tag{3.11}$$

Eventually, we will obtain an $\Psi_i(x)$ such that $\bigtriangledown \Psi_i(x)^T g(x) \neq 0$, and we will have an $i + 1$ degree barrier function.

3.2.3 First approach - Second order relative degree CBFs

Originally, for the system in Equation 3.1, the candidate CBF was:

$$h_1(X) = (E_{chg} - E)(E - E_{min}) = -E^2 + E(E_{chg} + E_{min}) - E_{chg}E_{min} \tag{3.12}$$

Taking the derivative of $h_1(X)$ we get:

$$\begin{aligned} \dot{h}_1(X) &= L_f h_1(X) + L_g h_1(X)u \geq -\alpha_1(h_1(X)) \\ \dot{h}_1(X) &= (E_{chg} + E_{min} - 2E)\dot{E} \geq -\alpha_1(h_1(X)) \Rightarrow \frac{\partial h_1^T}{\partial X} g(X)u \equiv 0 \end{aligned} \tag{3.13}$$

As the term $L_g h_1(X)u$ does not intervene in the equation, a second order CBF needs to be built:

$$\Psi_1(X) = \dot{h}_1(X) + \gamma_1 h_1(X) = (E_{chg} + E_{min} - 2E)\dot{E} + \gamma_1 h_1(X) \tag{3.14}$$

Note that in Equation 3.14 $\alpha_1(h_1(X))$ has been replaced by $\gamma_1 h_1(X)$. This is because the chosen class-K function is simply a scalar function named γ_1.

Taking now the derivative of the higher order CBF $\Psi_1(X)$:

$$\dot{\Psi}_1(X) = L_f^2 h_1(X) + L_g L_f h_1(X)u + \gamma_1 L_f h_1(X) \geq -\alpha_2(\Psi_1(X)) \tag{3.15}$$

Where the different terms that intervene in Equation 3.15 are:

$$\begin{aligned} L_f^2 h_1(X) &= \Bigg((E_{chg} + E_{min})K\bigg(\frac{e^{-\lambda(I(p)-I_c)}}{(E + (1-E)e^{-\lambda(I(p)-I_c)})^2} - 1\bigg) \\ &- 2K\bigg(\frac{E^2 + 2Ee^{-\lambda(I(p)-I_c)} - E^2 e^{-\lambda(I(p)-I_c)}}{(E + (1-E)e^{-\lambda(I(p)-I_c)})^2} - 2E\bigg)\Bigg)K\bigg(\frac{1}{1 + \frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E\bigg) \\ L_g L_f h_1(X) &= (E_{chg} + E_{min} - 2E)K\frac{E(1-E)e^{-\lambda(I(p)-I_c)}\lambda}{(E + (1-E)e^{-\lambda(I(p)-I_c)})^2}\frac{\partial I(p)}{\partial p} \\ L_f h_1(X) &= (E_{chg} + E_{min} - 2E)K\bigg(\frac{1}{1 + \frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E\bigg) \end{aligned}$$

One can see that $L_g L_f h_1(X) \equiv 0$ whenever the derivative of the light intensity $\frac{\partial I(p)}{\partial p} = 0$, i.e. at any singular point or if the function $I(p)$ is constant. As the light intensity function is unknown, it is not possible to assume that any of this situations happen, so the only alternative is to reformulate the problem.

3.2.4 Second approach - Candidate CBFs for energy dependent dynamics

As it has been presented in chapter 1, one of our main goals in this project is to ensure the survival of the robot. We understand surviving as both state variables, i.e. position and battery voltage, remaining within the boundaries of their respective safe sets.

We defined the safe sets for our problem to be:

- $p \in [0, 10]$
- $E \in [0.1, 0.9]$

In order to guarantee that these two variables remain inside their safe sets, we implemented two Control Barrier Functions (CBFs). The first one, noted as h_1, is used to maintain the battery voltage E within its maximum and minimum safe levels. The second one, noted as h_2, is for the position p, to ensure that the robot stays within the boundaries of its cable.

The proposed candidate CBFs are:

CBF for the battery voltage E:

$$h_1(X) = (E_{chg} - E)(E - E_{min}) = -E^2 + E(E_{chg} + E_{min}) - E_{chg}E_{min}$$

CBF for the position p:

$$h_2(X) = (P_{max} - p)(p - P_{min}) = -p^2 + p(P_{max} + P_{min}) - P_{max}P_{min}$$

These two proposed barriers are indeed valid CBFs of relative degree 1, as when computing their first Lie derivatives, neither $L_g h_i$ term equals 0:

$$L_f h_1(X) = (E_{chg} + E_{min} - 2E)K(\frac{1}{1 + \frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E)$$

$$L_g h_1(X) = -(E_{chg} + E_{min} - 2E)\rho$$

$$L_f h_2(X) = 0$$

$$L_g h_2(X) = (P_{max} + P_{min} - 2p)$$

This candidate CBFs will be the constraints of the optimization problem in section 3.3.

3.3 Optimization problem statement

The proposed robotics contribution to environmental research consists in fulfilling two tasks: Exploration and Surveillance of the robot's environment. These tasks are encoded through a nominal control input noted as $\hat{u}$ in Equation 3.16. The goal is to minimally modify this nominal control input while ensuring the survival of the robot. This is obtained by minimizing the squared norm of the difference between the nominal control $\hat{u}$ and the safe control u. The safe control u satisfies the two constraints that guarantee the system remains forward invariant.

When expressing these objectives in mathematical terms, the optimization problem shown in Equation 3.16 is obtained:

$$\begin{aligned} &\underset{u}{minimize} \quad ||u - \hat{u}||^2 \\ &s.t. \\ &\quad - L_g h_1(X)u^2 \leq \gamma_1 h_1(X) + L_f h_1(X) \\ &\quad - L_g h_2(X)u \leq \gamma_2 h_2(X) + L_f h_2(X) \end{aligned} \tag{3.16}$$

Where:

- $X = [E \quad p]$ is the state vector
- $L_f h_i, L_g h_i$ are the Lie Derivatives of the barrier function h_i.

The problem in Equation 3.16 is not an standard minimization problem, as it has a quadratic cost and two nested constraints, one of which is linear and the other quadratic. One could try to solve this problem as a Quadratically Constrained Quadratic Program (QCQP).

QCQP belong to the class of problems denoted as NP hard problems, which do not have an analytical solution and due to that fact, they can't be implemented in a microcontroller.

The general formulation of an QCQP [18] can be found in Equation 3.17. Where $A_0 \succeq 0$, i.e. positive definite, and A_m can be negative semidefinite or indefinite matrices.

$$\begin{aligned} &\underset{x \in \mathbb{C}^n}{minimize} \quad x^H A_0 x \\ &s.t. \\ &\quad x^H A_m x \leq u_m, \forall m \in \boldsymbol{M} \end{aligned} \tag{3.17}$$

In the SlothBot application, due to the fact that the kinematics are one dimensional, there is only one control input available, making both the cost function and the constraints scalar. This leads to a new class of functions that has been denoted as Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problems. In order to use the available energy in a conservative way, convex optimization algorithms such as Second Order Cone Programming (SOCP) [19], or relaxations as Semidefinite Programming [20], or Reformulation-Linearization techniques [21] are not an optimal approach. Instead, we will proof in section 3.4 that an analytic solution to the general QCSLQC can be found and successfully implemented in simulation.

3.4 Analytical solution to the general Quadratic Cost Scalar Linear and Quadratically Constrained (QCSLQC) problem

3.4.1 Parameterized minimization problem

In order to proof the existence of solutions to the described problem in section 3.3, a generalization in terms of parameters $a, b, \alpha,$ and β, and a free variable x is formulated as follows:

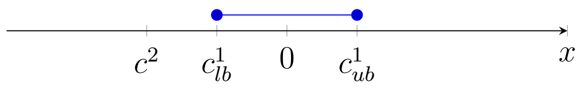
$$\begin{aligned} &\underset{x}{minimize} \quad ||x - \hat{x}||^2 \\ &s.t. \\ &\quad ax^2 \leq b \\ &\quad \alpha x \leq \beta \end{aligned} \tag{3.18}$$

The quadratic constraint defines two bounds of an interval in $\mathbb{R}$, whereas the linear constraint only defines one bound in $\mathbb{R}$. These nested constraints will lead to two possible scenarios:

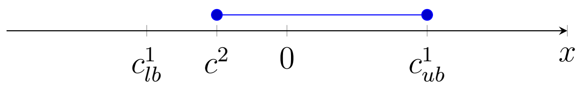
<u>Feasible Problem</u>

Where the region in $\mathbb{R}$ defined by the constraints is non-empty, and thus an optimal solution can exist:

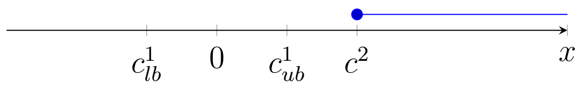
1. A closed feasible region is defined by the two bounds of the quadratic constraint

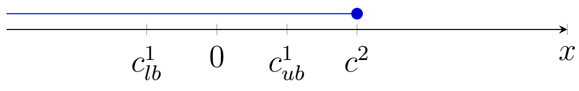


2. A closed feasible region is defined by the bound of the linear constraint and one of the bounds of the quadratic constraint



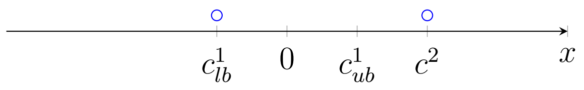
3. An open feasible region is defined by one of the constraints' bounds





Infeasible Problem

Where the region in $\mathbb{R}$ defined by the constraints is empty, for example when the lower bound is greater than the upper one, and thus an optimal solution cannot exist:



3.4.2 Analysis of parameterized nested quadratic and linear constraints

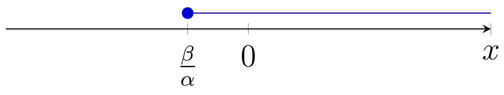
Once the two possible scenarios have been identified, the next step is to analyze all possible combinations of $a, b, \alpha,$ and β and verify which of them lead to an infeasible problem.

First constraint: $ax^2 \leq b$

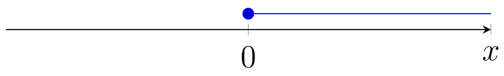
1. $a < 0, b > 0$

   Second constraint: $\alpha x \leq \beta$

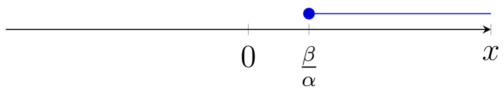
   (a) $\alpha < 0, \beta > 0 \Rightarrow x \geq \frac{\beta}{\alpha}$

   (b) $\alpha < 0, \beta = 0 \Rightarrow x \geq 0$

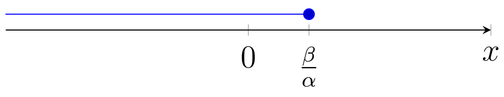
   

   (c) $\alpha < 0, \beta < 0 \Rightarrow x \geq \frac{\beta}{\alpha}$

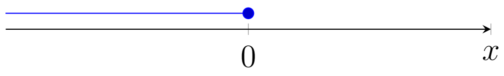
   

   (d) $\alpha = 0 :$ x would have no effect.

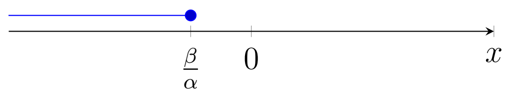
   (e) $\alpha > 0, \beta > 0 \Rightarrow x \leq \frac{\beta}{\alpha}$

   (f) $\alpha > 0, \beta = 0 \Rightarrow x \leq 0$

(g) $\alpha > 0$, $\beta < 0 \Rightarrow x \leq \frac{\beta}{\alpha}$



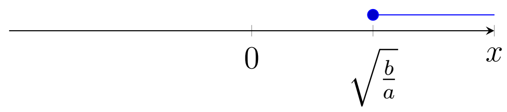
2. $a < 0$, $b = 0$. The feasible values for x after considering the second constraint are the same as in the previous case.

3. $a < 0, b < 0$

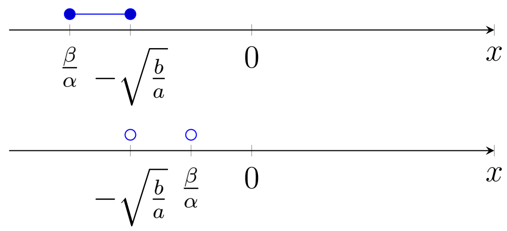
   Second constraint: $\alpha x \leq \beta$

   (a) $\alpha < 0$, $\beta > 0$

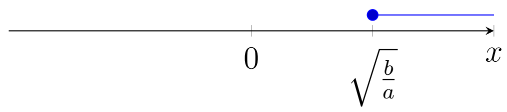
   i. $x \geq +\sqrt{\frac{b}{a}}$



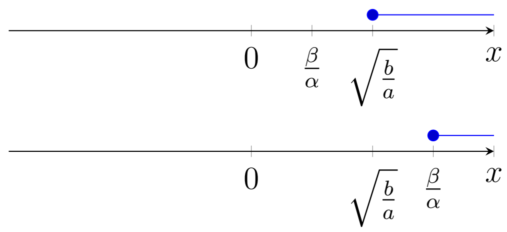
   ii. $\frac{\beta}{\alpha} \leq x \leq -\sqrt{\frac{b}{a}}$ If $\frac{\beta}{\alpha} > -\sqrt{\frac{b}{a}} \Rightarrow$ infeasible



   (b) $\alpha < 0$, $\beta = 0 \Rightarrow x \geq +\sqrt{\frac{b}{a}}$



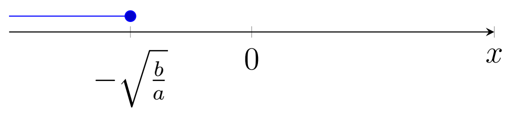
   (c) $\alpha < 0$, $\beta < 0 \Rightarrow x \geq max(+\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha})$



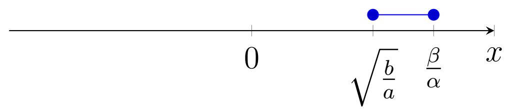
   (d) $\alpha = 0:\ x$ would have no effect.

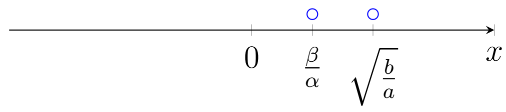
(e) $\alpha > 0$, $\beta > 0$

i. $x \leq -\sqrt{\frac{b}{a}}$

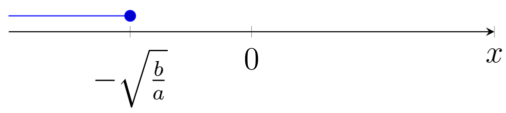


ii. $\sqrt{\frac{b}{a}} \leq x \leq \frac{\beta}{\alpha}$ If $\frac{\beta}{\alpha} < \sqrt{\frac{b}{a}} \Rightarrow$ infeasible

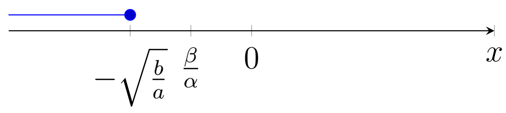


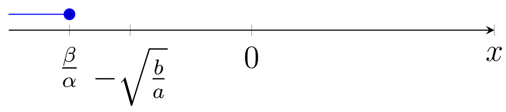


(f) $\alpha > 0$, $\beta = 0 \Rightarrow x \leq -\sqrt{\frac{b}{a}}$



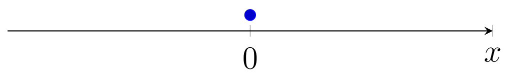
(g) $\alpha > 0$, $\beta < 0 \Rightarrow x \leq min(-\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha})$





4. $a > 0, b < 0 \Rightarrow$ Infeasible

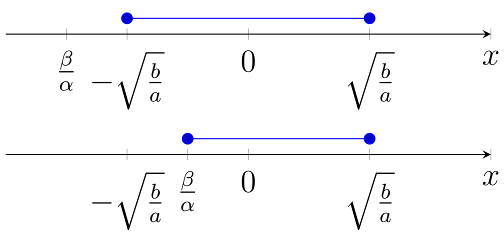
5. $a > 0, b = 0 \Rightarrow x = 0$



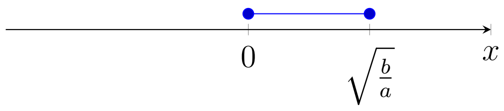
6. $a > 0, b > 0$

Second constraint: $\alpha x \leq \beta$

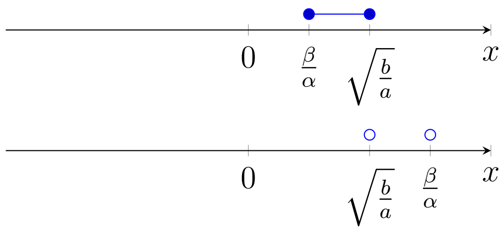
(a) $\alpha < 0$, $\beta > 0 \Rightarrow max(-\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha}) \leq x \leq +\sqrt{\frac{b}{a}}$



(b) $\alpha < 0, \beta = 0 \Rightarrow 0 \leq x \leq +\sqrt{\frac{b}{a}}$

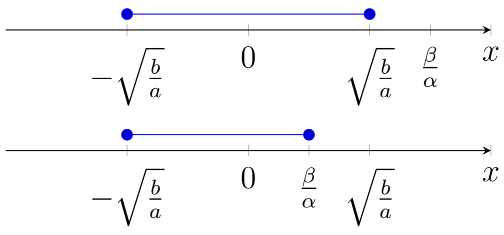


(c) $\alpha < 0, \beta < 0 \Rightarrow \frac{\beta}{\alpha} < x \leq +\sqrt{\frac{b}{a}}$ If $\frac{\beta}{\alpha} < \sqrt{\frac{b}{a}} \Rightarrow$ infeasible

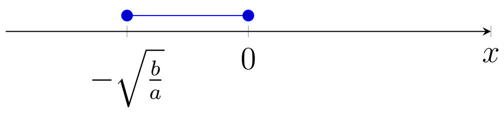


(d) $\alpha = 0:$ x would have no effect.

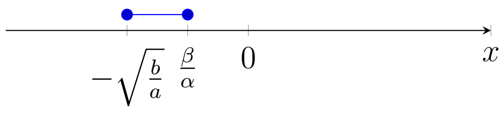
(e) $\alpha > 0, \beta > 0 \Rightarrow -\sqrt{\frac{b}{a}} \leq x \leq min(\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha})$

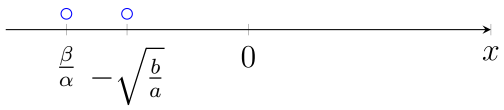


(f) $\alpha > 0, \beta = 0 \Rightarrow -\sqrt{\frac{b}{a}} \leq x \leq 0$



(g) $\alpha > 0, \beta < 0 \Rightarrow -\sqrt{\frac{b}{a}} \leq x \leq \frac{\beta}{\alpha}$ If $\frac{\beta}{\alpha} < -\sqrt{\frac{b}{a}} \Rightarrow$ infeasible





Based on the analysis above, it is possible to conclude that no feasible region exists, and thus no optimal solution, if any of the following parameters combinations occurs:

- $a < 0, b < 0, \alpha < 0, \beta > 0$:
  $x^* \in [\sqrt{\frac{b}{a}}, \infty) \cup [\frac{\beta}{\alpha}, -\sqrt{\frac{b}{a}}]$ Infeasible if $\frac{\beta}{\alpha} > -\sqrt{\frac{b}{a}}$
- $a < 0, b < 0, \alpha > 0, \beta > 0$:
  $x^* \in (-\infty, -\sqrt{\frac{b}{a}}] \cup [\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha}]$ Infeasible if $\frac{\beta}{\alpha} < \sqrt{\frac{b}{a}}$
- $a > 0, b < 0$
- $a > 0, b > 0, \alpha > 0, \beta < 0$:
  $x^* \in [-\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha}]$ Infeasible if $\frac{\beta}{\alpha} < -\sqrt{\frac{b}{a}}$

3.4.3 Constraints values for the SlothBot dynamics

To conclude the use of an analytic solution for the QCSLQC for robotic's tasks persistification, we verified if any of the identified infeasible cases could occur in the SlothBot application.

Recalling the dynamical equations for the environment and the SlothBot model, Equation 3.2, the general parameters a, b, α, β are equal to:

- $a = -L_g h_1(X) = E_{chg} + E_{min} - 2E$
- $b = \alpha_1(h_1) + L_f h_1(X) = \gamma_1(-E^2 + E(E_{chg} + E_{min}) - E_{chg}E_{min}) + (E_{chg} + E_{min} - 2E)K(\frac{1}{1+\frac{1-E}{E}e^{-\lambda(I(p)-I_c)}} - E)$
- $\alpha = -L_g h_2(X) = P_{max} + P_{min} - 2p$
- $\beta = \alpha_2(h_2) + L_f h_2(X) = \gamma_2(-p^2 + p(P_{max} + P_{min}) - P_{max}P_{min}) + 0$

Note that the used class-K functions were scalar: $\alpha_1(s) = \gamma_1 s$ and $\alpha_2(s) = \gamma_2 s$.

The strategy followed for this study consisted of plotting all four parameters, a, b, α, β, as functions of the state variables within their safe sets. The possibility for these state variables of taking values outside their safety boundaries is not considered, because of the action of the proposed barriers.

From Equation 3.2 it is known that the unitary battery voltage E dynamic equation is a function of both the position p and the voltage E. As the light intensity function $I(p)$ belongs to a scalar field bounded between $[0, 1]$ it is possible to fix the difference between the light intensity at the current position, and the light intensity threshold to its maximum, minimum and zero value. By doing so, any other value of such difference will lead to a curve bounded between the ones here represented. These graphs can be found in figures Figure 3.2 and Figure 3.3.

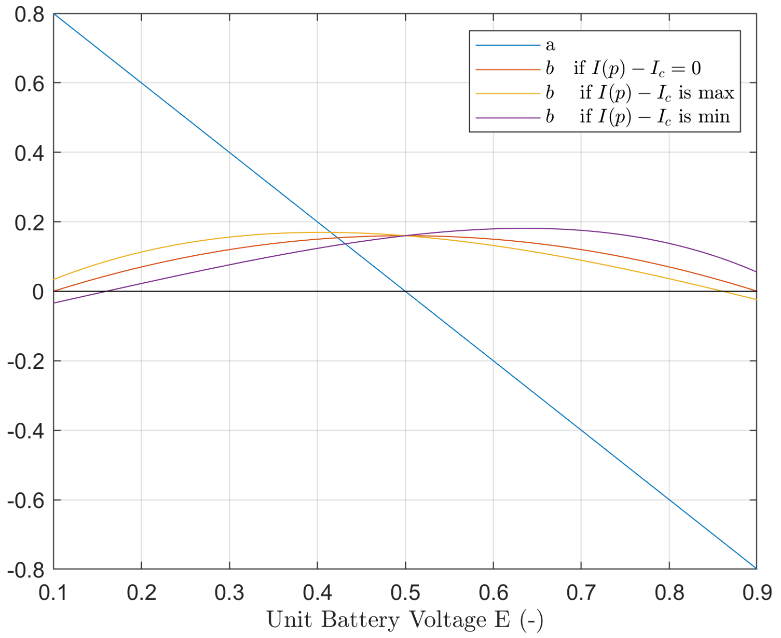


Figure 3.2: Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 1$

From the identified infeasible cases noted in subsection 3.4.2, three out of four correspond to negative values of b. In Figure 3.2 one can see that b is greater than zero for the

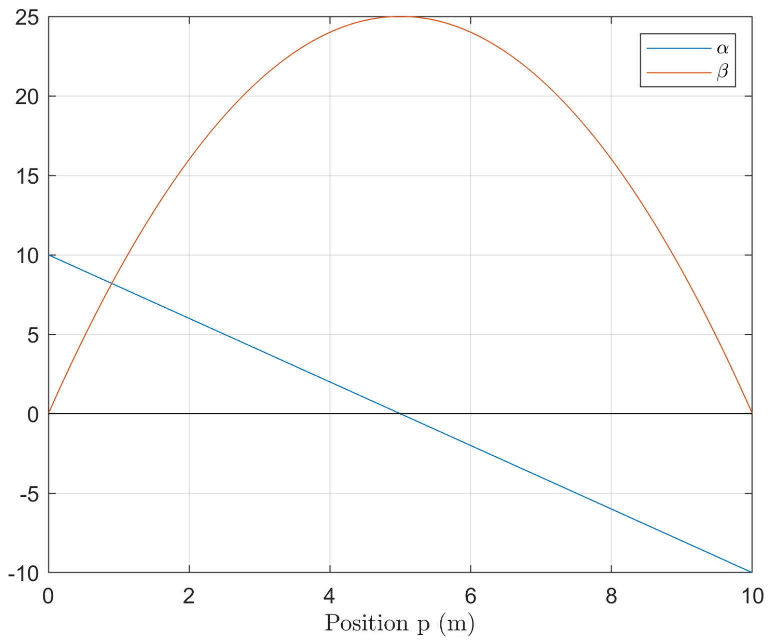


Figure 3.3: Terms α (Blue) and β (Orange) of the linear constraint

majority of the battery voltage E safe set:

For $\gamma_1 = 1$, the problem is infeasible if:

- $I(p) - I_c$ is minimum, and $E < 0.16$: This describes a situation when the SlothBot has its battery almost at its minimum value and thus the light intensity is zero.
- $I(p) - I_c$ is maximum, and $E > 0.86$: This is the opposite situation, when the Slothbot battery is almost fully charged at its maximum, and the light intensity is maximal.

From CBF's theory, it is possible to choose the class-K functions arbitrarily large or small, as it will result in a more aggressive or conservative control effort, when approaching the barrier boundary.

When increasing the value of γ_1 the regions where b is negative decreases, as shown in Figure 3.4, and Figure 3.5.

Taking a closer look of the values of E where b changes its sign:

For $\gamma_1 = 10$, the problem is infeasible if:

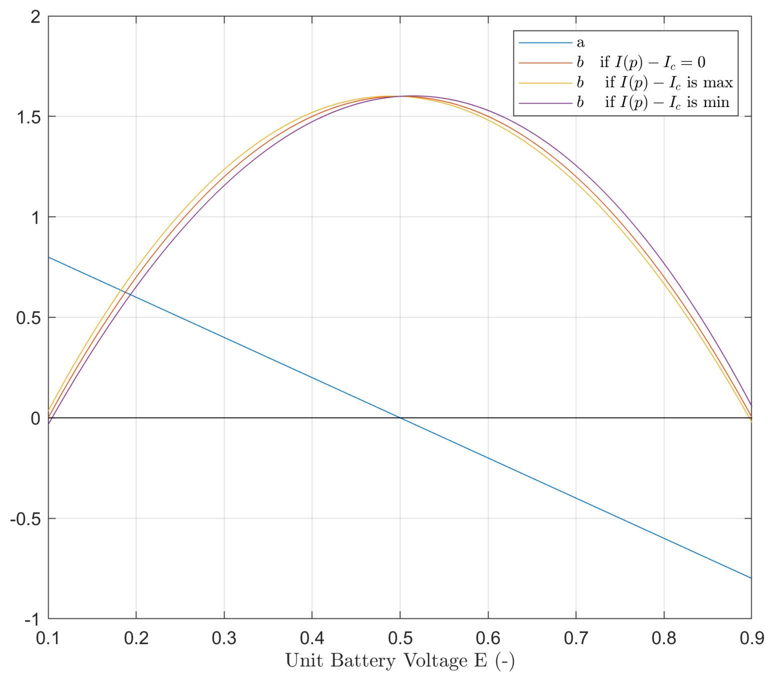


Figure 3.4: Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 10$

- $I(p) - I_c$ is minimum, and $E < 0.104$
- $I(p) - I_c$ is maximum, and $E > 0.8607$

For $\gamma_1 = 100$, the problem is infeasible if:

- $I(p) - I_c$ is minimum, and $E < 0.1004$
- $I(p) - I_c$ is maximum, and $E > 0.8997$

The results proved by Figure 3.2, Figure 3.4, and Figure 3.5 are summarized in Remark 1:

Remark 1:

Feasibility of the quadratic constraint can be ensured by choosing sufficiently large values of γ_1, as in standard Control Barrier Functions theory. As γ_1 becomes larger, the intervals where the quadratic constraint is infeasible approach the boundaries of the safe set for the battery voltage E.

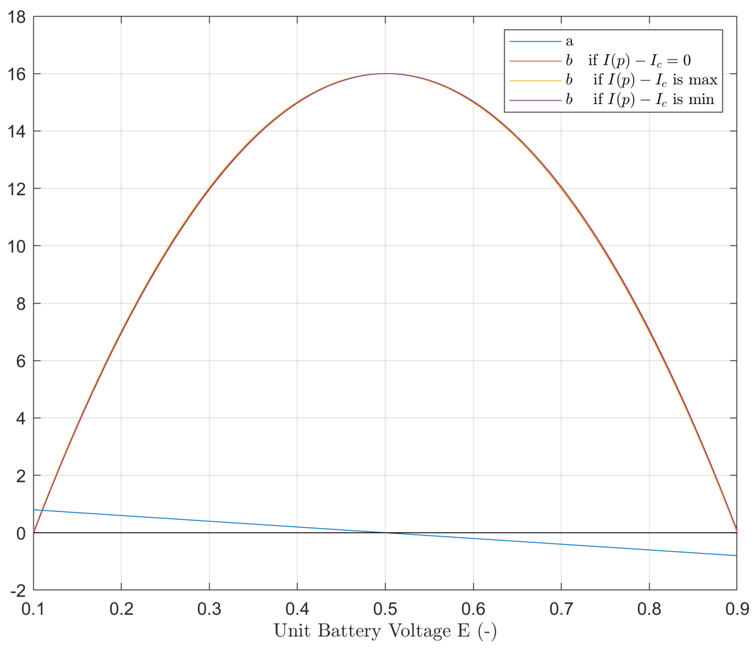


Figure 3.5: Terms a (Blue) and b (Orange, Yellow, Purple) of the quadratic constraint when $\gamma_1 = 100$

Now that it has been proven that three of the possible infeasible situations can be prevented, we need to evaluate the last one, that corresponds to values of the parameters $a, b, \alpha > 0$ and $\beta < 0$.

Analyzing Figure 3.3 the reader can see that β is always greater than zero, independently of the value of our class-K function γ_2, leading to a result presented in Remark 2:

Remark 2:

The linear constraint is always feasible in the SlothBot application, independently of the value of γ_2. Furthermore, it does not interfere with the existence of solutions for the Quadratically Constrained Quadratic Program.

3.5 Simulation results of the analytic solution for QCSLQC problem in the SlothBot application

After studying the analytic solution to the general QCSLQC problem in subsection 3.4.2, and the particular infeasible scenarios that can occur in the SlothBot application, the final step to conclude this study is to test the solution in simulation.

In order to test the solution to the QCSLQC, the light intensity model was assumed to be known. The simulation environment, shown in Figure 3.6, consists of an arbitrary light intensity function $I(p)$ (represented in blue), the light intensity threshold I_c (displayed as a orange straight line), and an schematic representation of the SlothBot on its wire. Depending on the value of the difference $I(p) - I_c$ at each particular location, the robot's battery will charge or discharge as it moves along the simulated wire.

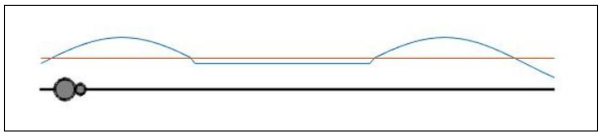


Figure 3.6: Light intensity function $I(p)$ (blue), light intensity threshold I_c (orange), and simulated SlothBot

During the simulation, the robot starts at position $p = 0$ and with a 20% of its battery, i.e. $E = 0.2$. During its mission it starts moving towards the end of the wire, and returns backwards to the starting position. The simulation results can be seen in Figure 3.7, where the orange horizontal line represents the upper bound of the safe sets. The graph at the top of Figure 3.7 shows how the position (blue curve), and thus the speed, of the robot are modified by the CBFs to ensure that the system state variables remain within their safe sets. The orange horizontal line represents the upper bound of the safe set of p. The graph at the bottom of Figure 3.7 displays the variations of the battery voltage (blue curve) due to the effect of both the light intensity and the control input applied. The orange horizontal line represents the upper bound of the safe set of E. It can be seen how, when the robot is moving forward and the difference $I(p) - I_c$ is not sufficiently large, its battery discharges, and also how it stops to recharge its battery when the voltage level is not enough to continue with its mission.

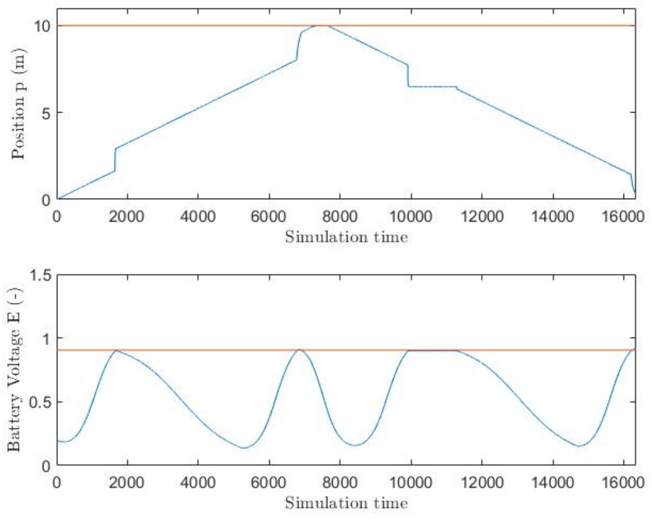


Figure 3.7: State variables when traversing the wire forward and backwards. Upper plot: Position p. Lower plot: Battery Voltage E

These simulation results, in addition to the analysis conducted in subsection 3.4.2, prove that under regular operation conditions the robot won't get stuck in a position where it is not possible to recharge its battery and continue its mission. The only scenarios in which a solution does not exist correspond to situations that in a real implementation, there is no action that the robot can take to charge its battery. These cases correspond to the robot having its battery almost depleted and being under minimum light intensity conditions, or vice versa, having its battery fully charged with maximum irradiance.

CHAPTER 4
SMOOTHING SPLINES FOR LIGHT INTENSITY MODEL ESTIMATION

Chapter 4 addresses the problem of finding a model for the solar light intensity, an unknown scalar valued function, while the robot is completing its mission. The idea is that, thanks to the light intensity sensor equipped on the robot, a set of N measurements will be gathered at different locations and times on the wire. Then, using these data and smoothing splines, a light intensity model will be built and periodically modified. Knowing this model will allow the robot to perform longer exploration missions, as the regions where its battery is drained or recharged will be fully characterized.

This chapter is organized as follows: section 4.1 presents the theory of Smoothing Splines, a technique based on a control systems approach to optimal trajectory planning. This section includes the theoretical derivation, a couple generic examples and concludes with an implementation for the light intensity. At the end of the section, the problem of large data sets arises, and to solve it we develop a recursive algorithm in section 4.2 to help keep the dimensions of the problem fixed. This theoretical derivation has been done for both continuous and discrete systems.

4.1 Smoothing Splines

As in any model fitting problem, the goal is to drive a variable close to certain waypoints at particular time instants. To do so, we propose a solution in terms of control theory, so the problem can be formulated as finding the control input that drives the output of an arbitrary system to a set of waypoints at certain interpolation time instants.

In order to choose the control input, we will solve a minimization problem on the control energy. The resulting output when using this controllers is named Smoothing Spline.

The proposed method in this section is a direct application of the published work by Dr.

Magnus B. Egerstedt and Dr. Clyde F. Martin in [2].

4.1.1 Theoretical derivation

Starting with a linear, single input multiple output (SIMO) time invariant system:

$$\begin{aligned} \dot{x} &= Ax + bu \\ y &= C^T x \end{aligned} \tag{4.1}$$

Where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^p$, $u \in \mathbb{R}$.

The assumption that system (A, b, C) is both controllable and observable is made. Without loss of generality it can also be assumed that the initial state corresponds to the zero state, i.e. $x(0) = 0$.

Given a set of light intensity measurements at different time instants (ξ_i, ti), the goal is to drive $y(t_i)$ close to ξ_i but without aiming for a perfect fit, but an overall output that represents the whole set of measurements. The justification to this approach is that measures are not perfect, as they have noise, so it is not worth to do the computational time investment to obtain a model based on imperfect samples of reality.

The cost functional that will be minimized while driving the output close to the given points and time instants is:

$$\frac{1}{2}\int_0^T \rho u^2(t)dt \tag{4.2}$$

With $u \in L_2[0, T]$, i.e. u is a continuously differentiable function.

Summarizing these two objectives into a single cost function, the minimization problem in Equation 4.3 is obtained:

$$\underset{u \in L_2[0,T]}{minimize} \quad \frac{1}{2}\int_0^T \rho u^2(t)dt + \frac{1}{2}\sum_{i=1}^{m}(y(t_i) - \xi_i)^T \tau_i (y(t_i) - \xi_i) \tag{4.3}$$

Where ρ controls the amount of smoothing in the output, and $\tau_i = diag(\tau_{i1}, ..., \tau_{ip}) \succeq$

$0 \in \mathbb{R}^{pxp}$ expresses the importance for the jth output to interpolate close to ξ_i's jth component at time t_i.

One can easily note that Equation 4.3 depends both on u, and y. Using the fact that the system is linear, and thus has an analytical solution, it is possible to define a set of linearly independent basis functions:

$$g_i(t) \begin{cases} C^T e^{A(t_i - t)} b & t_i - t \geq 0 \\ 0 & otherwise \end{cases} \tag{4.4}$$

Substituting Equation 4.4 in Equation 4.3, a cost function only in terms of the free variable u is obtained:

$$\begin{aligned} J(u) &= \frac{1}{2}\int_0^T \rho u^2(t)dt + \frac{1}{2}\sum_{i=1}^{m}(\int_0^T g_i(t)u(t)dt - \xi_i)^T \tau_i (\int_0^T g_i(t)u(t)dt - \xi_i) \\ &= \frac{1}{2}\int_0^T \rho u^2(t)dt + \frac{1}{2}(\int_0^T g(t)u(t)dt - \xi)^T \tau (\int_0^T g(t)u(t)dt - \xi) \end{aligned} \tag{4.5}$$

$J(u)$ is closed and quadratic in u, and as all parameters are positive or positive semidefinite if they are matrices, it is thus convex, and from existence and uniqueness of solutions an optimum exists [22].

To find the optimum, Egerstedt and Clide [2] proposed using the Fréchet differential, as it will we make the cost function vanish for all increments of $h \in L_2[0, T]$ when using the unique minimizer u_0.

Fréchet differential

The Fréchet derivative or directional derivative $D_u f$ defines the rate a which a function, for example $f(x, y)$, changes at each point $\mathbf{x} = a$ in the direction of u. u is an unit vector that specifies the points in the direction in which the slope wants to be computed.

The directional derivative of $f(x, y)$ in the direction of u at the point a is computed as:

$$D_u f(a) = \lim_{h \to 0} \frac{f(a+hu) - f(a)}{h} \tag{4.6}$$

It is important to note that $D_u f(a)$ is not a matrix, it is a number.

Applying the definition in Equation 4.6 to the cost function $J(u)$ from Equation 4.5:

$$D_\epsilon J(u) = \lim_{\epsilon \to 0} \frac{J(u+\epsilon h) - J(u)}{\epsilon} = \int_0^T \left(\rho u(t) + g(t)^T \tau \left(\int_0^T g(s)u(s)ds - \xi \right)\right) h(t)dt \tag{4.7}$$

And for the differential to vanish for all $h \in L_2[0,T]$:

$$\rho u(t) + g(t)^T \tau \left(\int_0^T g(s)u(s)ds - \xi \right) = 0 \tag{4.8}$$

Using a finite dimensional parameterization of the problem $u(t) = \eta^T g(t)$, we obtain an optimal η and minimizer u_0:

$$\begin{aligned} \eta &= (\rho I + \tau G)^{-1} \tau \epsilon \\ u_0(t) &= g(t)^T (\rho I + \tau G)^{-1} \tau \epsilon \end{aligned} \tag{4.9}$$

Where $G \in \mathbb{R}^{mpxnp}$ is the Gramian of our basis functions g:

$$G = \int_0^T g(s)g(s)^T ds \tag{4.10}$$

4.1.2 Implementation of the algorithm

We now evaluate the performance of the proposed smoothing splines in subsection 4.1.1 in two different systems:

System 1

We will first consider the third order control system given in canonical form:

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u \tag{4.11}$$

$$y = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} x$$

It is important to note that we named this formulation as canonical but it is indeed a combination of both controllable and observable canonical forms. These system dynamics lead to a third order spline output, Figure 4.1 shows the different outputs obtained when the value of the smoothing parameter ρ is modified. A smaller value of ρ will make the output get closer to the given waypoints.

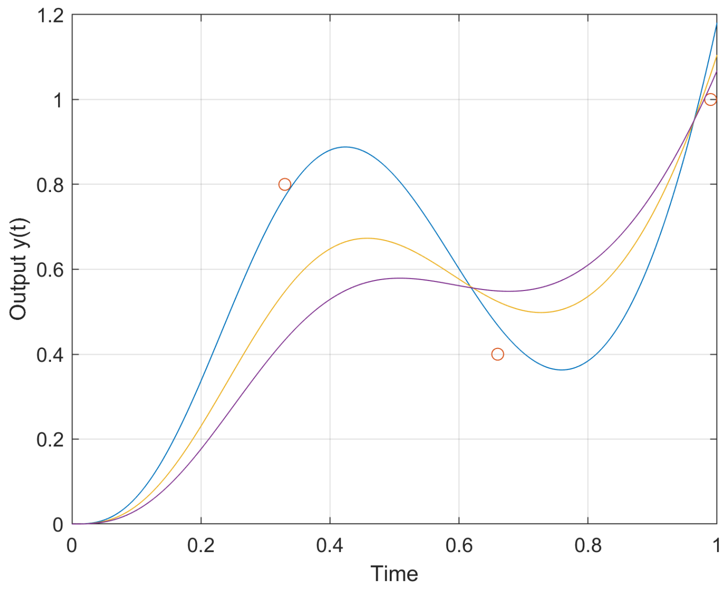


Figure 4.1: Waypoints and smoothing spline with different smoothing parameter ρ: 1/1000000 (Blue), 1/100000 (Yellow), and 2/100000 (Purple)

System 2

For our second system we decided to introduce poles different from zero and compare the performance with the previous canonical system:

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -10 & -5 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u \tag{4.12}$$

$$y = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} x$$

Figure 4.2 shows the obtained outputs for this new system and the same values of the smoothing parameter ρ.

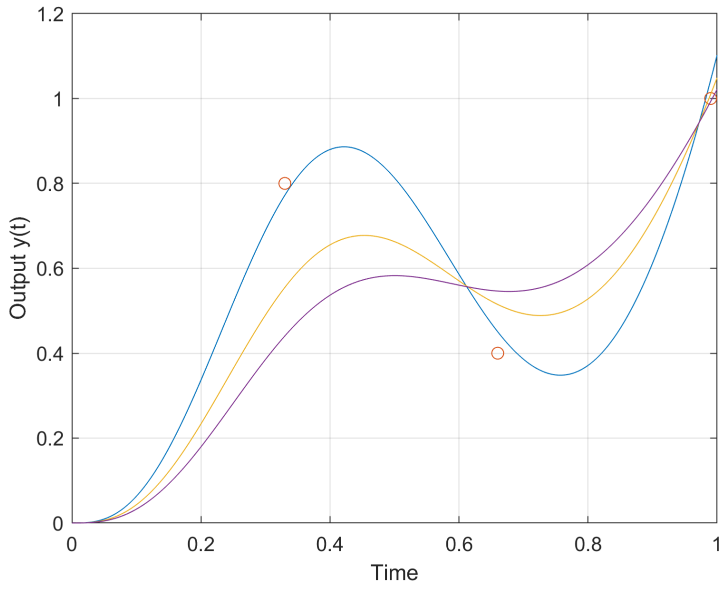


Figure 4.2: Waypoints and smoothing spline with different smoothing parameter ρ: 1/1000000 (Blue), 1/100000 (Yellow), and 2/100000 (Purple)

Even tough these results may look very similar to the ones obtained for the first system, Figure 4.1, if we compare the two outputs for the same value of ρ, we can see that they indeed have minor differences, Figure 4.3.

After taking a closer look to the performance of these two systems we can conclude this section with Remark 3.

Remark 3

The poles of the linear SIMO system used for the computation of the smoothing splines

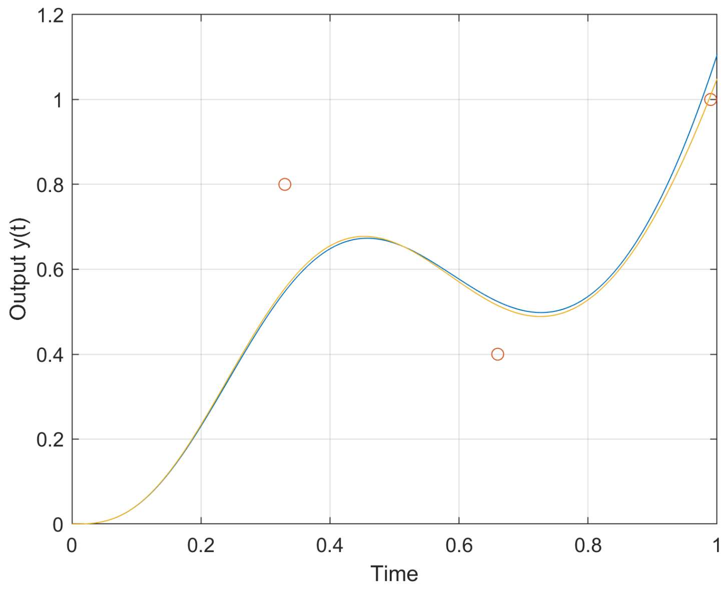


Figure 4.3: Waypoints and smoothing spline with $\rho = 1/1000000$ for system 1 (Blue), and system 2 (Yellow)

do not have an impact on the performance of this technique.

4.1.3 Application to Light Intensity Model Fitting

When using the technique presented in subsection 4.1.1 for finding the light intensity model, we would like to introduce new measurements. i.e. waypoints, when the robot moves to a new location, so a model for the light intensity over the whole wire can be obtained. Figure 4.4, Figure 4.5, and Figure 4.6 show how, as new data samples are included, the resulting smoothing spline modifies its shape to fit the whole set of waypoints.

The problem that arises with this approach is that Equation 4.9 requires to calculate a Gramian and a matrix inverse. These two operations' computation times grow exponentially with the size of the problem, so after including a few measurement sets, it becomes inefficient.

The solution proposed to this problem consists of the use of recursive splines, section 4.2.

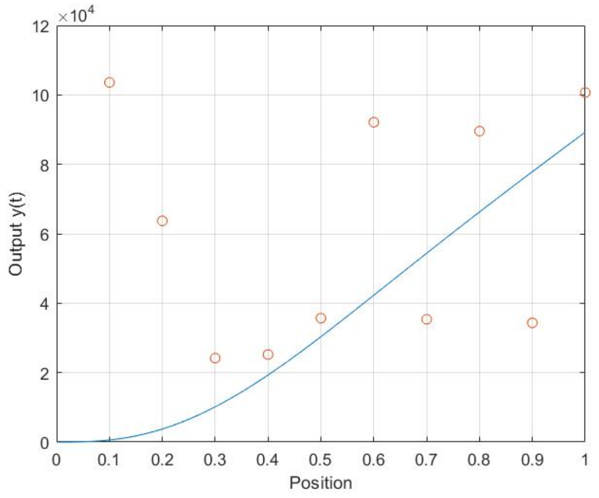


Figure 4.4: Waypoints and smoothing spline with 10 samples

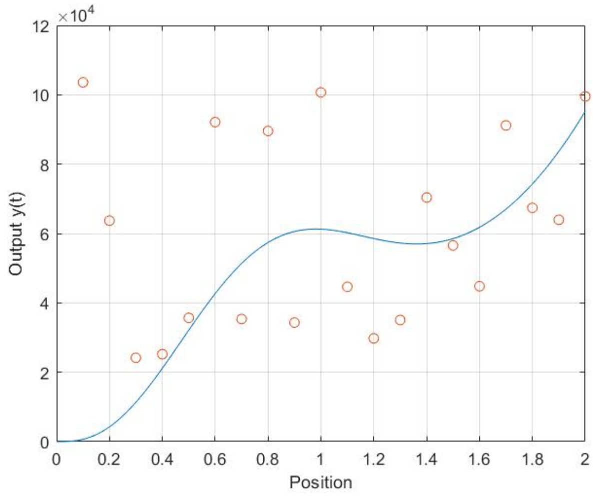


Figure 4.5: Waypoints and smoothing spline with 20 samples

4.1.4 Simulation results for the SlothBot application

In order to test the splines algorithm with the robot itself, we built a simulated experiment where the SlothBot starts at $p = 0$, with a 60% of its battery, i.e. $E = 0.6$ and doesn't know the light intensity model.

The light intensity model is randomly generated using the *spline* function from Matlab, and a set of 10 random samples between 0 and 1.

To generate the samples we basically used the average values, in lux, of the iluminance

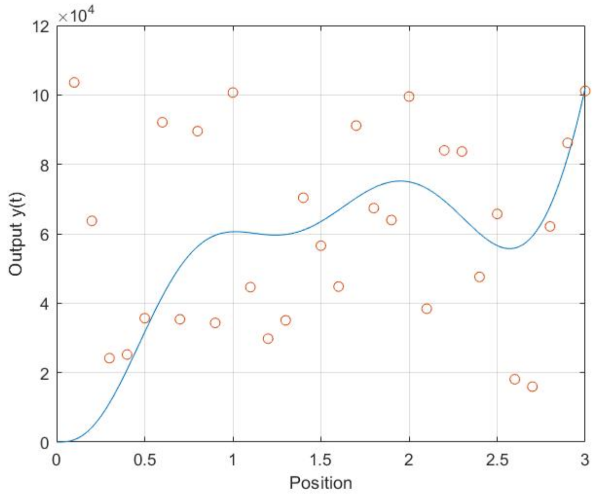


Figure 4.6: Waypoints and smoothing spline with 30 samples

for outdoors environments, Table 4.1, for a full daylight to sunlight condition and normalized it. These data has been obtained from *The Engineering ToolBox*: (https://www.engineeringtoolbox.com.)

Condition	Foot candles	Lux
Sunlight	10000	107527
Full daylight	1000	10752
Overcast day	100	1075
Very Dark day	10	107
Twilight	1	10.8
Deep twilight	0.1	1.08
Full moon	0.01	0.108
Quarter moon	0.001	0.0108
Starlight	0.0001	0.0011
Overcast night	0.00001	0.0001

Table 4.1: Illuminance values for different outdoor conditions

The simulation works as follows:

1. After executing the program, an Slothbot object is created and initialized with $p = 0$ and $E = 0.6$ initial conditions.

2. Then, the arbitrary light intensity model is generated.

3. The Slothbot starts traversing the wire, up to $p = 1$, with the CBFs ensuring that he

is not running out of battery.

4. Once it reaches this position, it gathers 10 samples and fits its own light intensity model.

5. Finally, using the information of the model, we can implement a feedback action to modify the mission in case the current objective would lead the SlothBot to an unsolvable situation, such as running out of battery and going to an area without the sufficient light intensity.

Some captions of both the simulator and the accuracy of the model obtained can be seen in Figure 4.7 and Figure 4.8.

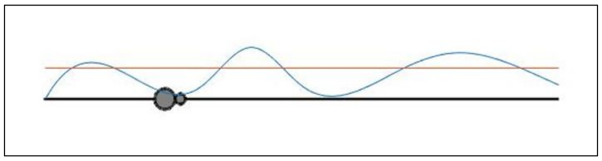


Figure 4.7: SlothBot simulator interface. The blue curve represents the real light intensity and the red line the value for I_c

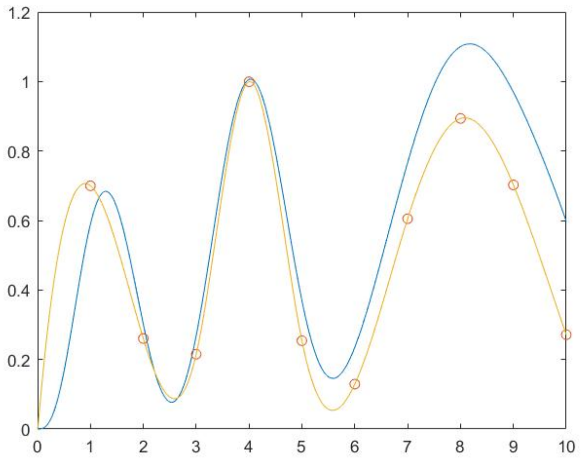


Figure 4.8: Real light intensity model (Yellow), samples gathered by the SlothBot (Orange circles), and model obtained with smoothing splines (Blue)

4.2 Recursive Splines

As it has been mentioned numerous times in this thesis, one of the robot's objective is environmental exploration. Identifying the light intensity model would allow the robot to go for longer displacements on the wire during its missions, as it would know which areas of the wire have sunlight and help him to get its battery recharged, and which ones are in the shadow and it should avoid for long time periods.

Thanks to its light intensity sensors, the robot will take N measurements at different locations, and time schemes. Our goal now is to use each of these new sets to modify the previous model, i.e. spline output, so the dimensions of the problem remain fixed.

4.2.1 Problem Statement

We define the sequence of data sets of equal size as:

$$D_n = \{\alpha_{in} \ : i = 1, ..., N\} \tag{4.13}$$

Each data point α_{in} is of the form:

$$\alpha_{in} = f(t_{in}) + \epsilon_{in} \tag{4.14}$$

Note that we use the index $i = 1, ..., N$ to denote each measurement of a data set, and $n = 1, 2, ...$ for the working set of samples.

$f(t_{in})$ is assumed to be at least piecewise smooth, and ϵ_{in} are values of a random variable symmetrically distributed about 0.

The set $\{t_{in} : \ i = 1, ..., N, n = 1, 2, ...\}$ is dense in $[0, T]$. This set denotes the different time instants for which we have a sample in the current working data set n.

For each working set n, the cost function is defined as:

$$J_n(u) = \int_0^T u(t)^2\,dt + \lambda_n \int_0^T (y(t) - y_{n-1}(t))^2\,dt + \sum_{i=1}^{N} (y(t_{in}) - \alpha_{in})^2 \tag{4.15}$$

The different elements that can be identified in Equation 4.15 are:

- $\int_0^T u(t)^2\,dt \rightarrow$ Control energy
- $\lambda_n \int_0^T (y(t) - y_{n-1}(t))^2\,dt \rightarrow$ Predictor: Ensures that our new solution does not differ too much from the previous one
- $\sum_{i=1}^{N} (y(t_{in}) - \alpha_{in})^2 \rightarrow$ Corrector: Minimizes the difference between the current output and the given set of data points

The solution $u_n(t)$ and $y_n(t)$ are the optimal control and output when minimizing the cost function, equation Equation 4.15, for the sample set n.

λ_n expresses a sequence of coefficients that approach to infinity. We will choose it in such a way that our sequence of smoothing splines $\{y_n(t)\}_{n=1}^{\infty}$ converges.

Instead of using classical optimal control techniques, building the Hamiltonian and evaluating the necessary conditions for optimallity, we propose to find optimum as a projection in a Hilbert Space $H_n = \{(u; g; \alpha) | \, (u; g; \alpha) \in L_2[0, T] \,\text{x}\, L_2[0, T] \,\text{x}\, \mathbb{R}^N\}$.

A brief revision of the procedure to compute projections with any subspace or variety can be found in subsection 4.2.2.

4.2.2 Projections

Given the minimization problem:

$$\begin{aligned} &\underset{x}{minimize} \quad ||x - p||^2 \\ &s.t. \quad Ax = b \Rightarrow x \in \{z | Az = b\} = V_b \end{aligned} \tag{4.16}$$

Where V_b is an affine hyperplane. We know that the optimum x^* is the projection of p on V_b, as it is shown in figure Figure 4.9.

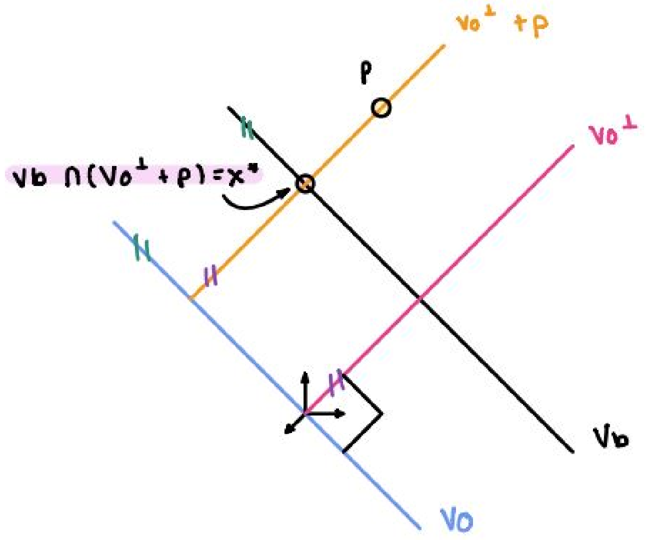


Figure 4.9: Projection: Geometric construction

In order to find the projection of p on the hyperplane V_b we will follow the steps pictured in figure Figure 4.9 and described below:

1. Find the subspace V_0, which is parallel to V_b and goes through the origin
2. Find its orthogonal compliment subspace $V_0^\perp$
3. Shift $V_0^\perp$ so it contains our point of interest p, this is done by simply adding a constant vector
4. x^* is the intersection: $V_b \cap (V_0^\perp + p)$

Following the derivation above, we see that we can compute projections as long as we can talk about orthogonality. This property also holds in Hilbert spaces, at it is indeed what we will use to solve our minimization problem.

4.2.3 Two-point boundary problem

In order to illustrate how to compute projections in Hilbert Spaces, let us start with a well-known optimal control example and minimize the control energy for a linear system with fixed initial and terminal states:

$$\begin{aligned}
&\underset{u}{minimize} \quad \int_0^T ||u||^2 \, dt \\
&s.t. \\
&x(0) = x_0 \quad x(T) = x_T \\
&\dot{x} = Ax + Bu
\end{aligned} \tag{4.17}$$

Our solution will consist on finding a projection in the Hilbert space $H = \{u | \, u \in L_2[0, T]\}$.

We will start noting that we can use the following alternative notation for the objective function:

$$\int_0^T ||u||^2 \, dt = ||u||^2_{L_2} = ||u - 0||^2_{L_2} \tag{4.18}$$

Where $p = 0$.

As the system is linear, it has solution:

$$x(T) = e^{A^T} x_0 + \int_0^T e^{A(T-t)} Bu(t) \, dt \tag{4.19}$$

The integral term is indeed a linear operator. It takes $u(t)$, a function in L_2, as an input and outputs a number in $\mathbb{R}^n$. We can thus define the operator L as:

$$\int_0^T e^{A(T-t)} Bu(t) \, dt = Lu, \quad L : L_2 \to \mathbb{R}^n \tag{4.20}$$

We now define $\xi = x_T - e^{AT}x_0$, and the variety:

$$V_\xi = \{u|\, Lu = \xi\} \tag{4.21}$$

Furthermore, the subspace of interest can be defined as:

$$V_0 = \{u|\, Lu = 0\} \tag{4.22}$$

And its orthogonal complement:

$$V_0^\perp = \{w|\ < w, u >_{L_2} = 0 \quad \forall u \in V_0\} \tag{4.23}$$

The graphical representation of the above varieties can be found in figure Figure 4.10.

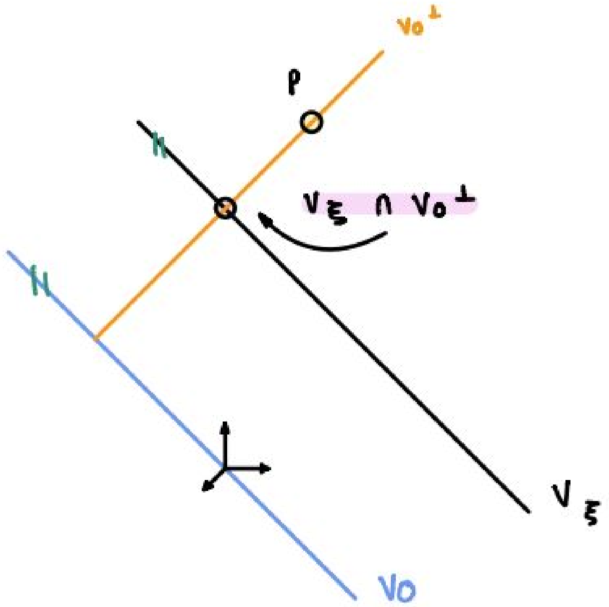


Figure 4.10: V_ξ, V_0, and $V_0^\perp$ representation

As $u \in V_0 \rightarrow Lu = 0$ by definition. We have defined $Lu \in \mathbb{R}^n$, so we can transform an inner product in L_2 to one in $\mathbb{R}^n$ as follows:

$$< w, u >_{L_2} = 0 = < \eta, \underset{0}{L} u >_{\mathbb{R}^n} \tag{4.24}$$

Before we can continue with our derivation, we need to recall the definition of adjoint operators.

Adjoint operator:

Given an operator M, with Domain D_L such that D_L is dense in H, the adjoint M^* of M is defined as:

$$< z, My >_{D_L} = < M^* z, y >_{D_L^*} \quad \forall y \in D_L^* \tag{4.25}$$

It is important to note that the domain D_L^* of M^* is not necessarily the same as D_L.

If $M \in \mathbb{R}^{mxm}$, and $z,\ y \in \mathbb{R}^n$. Then the adjoint of $< z, My >_{\mathbb{R}^m}$ can be performed by simply using transposition:

$$< z, My >_{\mathbb{R}^m} = z^T My = (M^T z)^T y = < M^T z, y >_{\mathbb{R}^n} \tag{4.26}$$

The transpose term that appears when shifting M to the left hand side term shows that the transpose is an adjoin operator.

Continuing with the proposed method, we now want to find w, elements of the orthogonal complement $V_0^{\perp}$. Applying the same reasoning as in Equation 4.25:

$$< \eta, Lu >_{\mathbb{R}^n} = < L^* \eta, u >_{L_2} \rightarrow w = L^* \eta \tag{4.27}$$

Substituting in the definition of $V_0^{\perp}$:

$$V_0^{\perp} = \{ w |\, w = L^* \eta, \quad \eta \in \mathbb{R}^n \} \tag{4.28}$$

If we now substitute in the inner product expression of Equation 4.27 the definition of L from Equation 4.20:

$$\begin{aligned}&< \eta, Lu >_{\mathbb{R}^n} = \eta^T \int_0^T e^{A(T-t)} Bu(t)\, dt = \int_0^T (B^T e^{A^T(T-t)}\eta)^T u(t)\, dt \\ &=< B^T e^{A^T(T-t)}\eta, u(t) >_{L_2} =< L^*\eta, u(t) >_{L_2}\end{aligned} \tag{4.29}$$

Recalling the definition of V_ξ:

$$V_\xi = \{u |\, Lu = \xi\} \tag{4.30}$$

And expressing $V_0^\perp$ in terms of u:

$$V_0^\perp = \{u |\, u = L^*\eta, \quad \eta \in \mathbb{R}^n\} \tag{4.31}$$

Substituting the definition of u from Equation 4.31 in Equation 4.30:

$$LL^*\eta = \xi = \int_0^T e^{A(T-t)} BB^T e^{A^T(T-t)} \eta\eta\, dt \rightarrow \Gamma\eta = \xi \tag{4.32}$$

Where Γ denotes the *Controllability Gramian*. Assuming that the system is controllable, so we can compute Γ^{-1}:

$$\eta = \Gamma^{-1}\xi \tag{4.33}$$

Then the optimal control, solution of the minimization problem stated in Equation 4.17 results:

$$u = L^*\eta = B^T e^{A^T(T-t)} \Gamma^{-1}\xi = B^T e^{A^T(T-t)} \Gamma^{-1}(x_T e^{A^T} - x_0) \tag{4.34}$$

4.2.4 Application to a dynamical system

We will now extend the idea introduced in subsection 4.2.3 to solve the recursive splines minimization problem. This is again a minimum norm problem in the following Hilbert

Space:

$$H = \{(u; g; \alpha) | \, (u; g; \alpha) \in L_2[0, T] \, \mathrm{x} \, L_2[0, T] \, \mathrm{x} \, \mathbb{R}^N\} \tag{4.35}$$

Where, as we will see later in this section, u corresponds to the control input, g the output of the system, and α a sampled value of the output. This Hilbert Space has norm:

$$||(u; g; \alpha)||^2_{H_n} = \int_0^T u(t)^2 \, dt + \lambda_n \int_0^T g(t)^2 \, dt + \alpha^T \alpha \tag{4.36}$$

As we are using a linear system, as we did with the smoothing splines in section 4.1, we define a set of basis functions:

$$l_{in}(s) \begin{cases} ce^{A(t_{in}-s)}b & t_{in} - s \geq 0 \\ 0 & otherwise \end{cases} \tag{4.37}$$

And our linear variety:

$$\begin{aligned} &V_{x_0} = \{(u; g; \alpha) | \, y(t) = ce^{At}x_0 + \int_0^T ce^{A(t-s)}bu(s) \, ds, \\ &z_i = ce^{At_{in}}x_0 + \int_0^T l_{in}u(s) \, ds\} \end{aligned} \tag{4.38}$$

Where $y(t)$ expresses the output as an $L_2[0, T]$ function and z_i is a particular value of the output at time instant i, so it is a scalar in $\mathbb{R}^k$.

We define our data point of interest in H as:

$$p_n = (0, y_{n-1}(t), \alpha^n), \quad \text{where} \quad \alpha^n = (\alpha_{1n}, ..., \alpha_{Nn}) \tag{4.39}$$

Note that p_n is defined with the information of the previous spline y_{n-1} and the new set of measurements that we want to include in our interpolation α^n.

The solution to our minimization problem will be the unique point in the variety V_{x_0} that is closest to p_n. Solving this problem for continuous data is of high complexity, so we will approximate it to one with only discrete data. To do so, we approximate the second integral term of the cost function defined in Equation 4.15 by an finite sum.

It is possible to use the Riemman sum approximation, but we will use a quadrature scheme instead:

$$T_j = (r_{j1}, ..., r_{jj}, 0, 0, ...) \tag{4.40}$$

$$S_j = (\beta_{j1}, ..., \beta_{jj}, 0, 0, ...) \tag{4.41}$$

Our quadrature scheme:

$$E_j(f) = \left| \int_0^T f(t)dt - \sum_{q=1}^{j} (\beta_{qj} f(qj)) \right| \tag{4.42}$$

Converges if for any continuous function $f \in C[0, T]$:

$$\lim_{j \to \infty} E_j(f) = 0 \tag{4.43}$$

Finally, our cost function is redefined as:

$$J_n^j(u) = \int_0^T u(t)^2 \, dt + \lambda_n \sum_{q=1}^{j} \beta_{qj} (y(r_{qj}) - y_{n-1}(r_{qj}))^2 + \sum_{i=1}^{N} (y(t_{in}) - \alpha_{in})^2 \tag{4.44}$$

Note that the subscripts refer to:

- $n \to$ working set of measurements and output spline
- $i \to$ particular data point of the set
- $q \to$ element index in the quadrature scheme
- $j \to$ total elements in the approximation of the integral as a sum ($j \to \infty \Rightarrow$ perfect approximation)

The optimal solution consists on finding the unique point of V_{x_0} that is closest to the given point p_n.

To solve this problem, we have two possible approaches, one continuous and another one discrete. These two approaches are covered in subsection 4.2.5 and subsection 4.2.6 respectively.

4.2.5 Recursive Splines - Continuous Case

Let our variety be:

$$\begin{aligned} V_{x_0}^n = \{(u; y; y(t_{in}))|\, y(t) = ce^{At}x_0 + \int_0^t ce^{A(t-s)}bu(s)\,ds, \\ y(t_{in}) = ce^{At_i}x_0 + \int_0^T l_{in}u(s)\,ds\} \end{aligned} \tag{4.45}$$

Where n expresses the working data set.

Then the subspace that goes through the origin is defined as:

$$\begin{aligned} V_0^n = \{(u; y; y(t_{in}))|\, y(t) = \int_0^t ce^{A(t-s)}bu(s)\,ds, \\ y(t_{in}) = \int_0^T l_{in}u(s)\,ds\} \end{aligned} \tag{4.46}$$

And its orthogonal compliment:

$$\begin{aligned} V_0^\perp &= \{(w; z; \beta)|\, < w; u >_{L_2} + < z; y >_{L_2} + < \beta; \alpha >_{\mathbb{R}^n} = 0\} \\ &= \{(w; z; \beta)|\, \int_0^T w(s)u(s)\,ds + \lambda_n \int_0^T z(t)y(t)\,dt + \sum_{i=1}^N y(t_{in})\beta_i = 0\} \end{aligned} \tag{4.47}$$

Substituting the expressions for $y(t)$ and $y(t_{in})$:

$$\int_0^T w(s)u(s)\,ds + \lambda_n \int_0^T z(t) \int_0^t ce^{A(t-s)}bu(s)\,ds\,dt + \sum_{i=1}^N \int_0^T l_{in}u(s)\,ds\beta_i = 0 \tag{4.48}$$

We will now manipulate the above expression to get an equation with only integrals between 0 and T of functions times $u(s)$ and equal to zero.

As $z(t)$ is not a function of s, we can introduce it into the inner integral, and also by linearity we can take the integral of the third term outside the summation:

$$\int_0^T w(s)u(s)\,ds + \lambda_n \int_0^T \int_0^t z(t)ce^{A(t-s)}bu(s)\,ds\,dt + \int_0^T \sum_{i=1}^N l_{in}u(s)\,ds\beta_i = 0 \tag{4.49}$$

Finally, we make a change in the order of integration in the double integral of the second term of our expression, so we get:

$$\int_0^T w(s)u(s)\,ds + \lambda_n \int_0^T \int_s^T z(t)ce^{A(t-s)}bu(s)\,ds\,dt + \int_0^T \sum_{i=1}^N l_{in}u(s)\,ds\beta_i = 0 \tag{4.50}$$

And as $u(\cdot) \neq 0$ we can conclude:

$$V_0^\perp = \{(w;z;\beta)|\, w(s) + \lambda_n \int_s^T z(t)ce^{A(t-s)}b\,dt + \sum_{i=1}^N l_{in}(s)\beta_i = 0\} \tag{4.51}$$

We want to find the projection of the point p on our variety V_{x_0}. We denote this projection as p_p. In order to obtain p_p, we first need to shift $V_0^\perp$ so it contains p as follows:

$$\begin{aligned} &p_p \in V_{x_0} = \{(u;y;y(t_{in}))\} \\ &(u;y;y(t_{in})) \in V_0^\perp + p\,; \quad (u;y;y(t_{in})) - p \in V_0^\perp \\ &(u;y;y(t_{in})) - (0, y_{n-1}(t), \alpha^n) \in V_0^\perp \end{aligned} \tag{4.52}$$

Giving the final expression:

$$(u;y;y(t_{in})) - (0, y_{n-1}(t), \alpha^n) \in V_0^\perp \Rightarrow u(s) + \lambda_n \int_s^T (y(t) - y_{n-1}(t))ce^{A(t-s)}b\,dt + \sum_{i=1}^N l_{in}(s)(y(t_{in}) - \alpha_{in}) = 0 \tag{4.53}$$

In order to obtain the intersection of V_{x_0} and $V^\perp + p$, i.e. p_p, the closest point to p, we must solve the system of equations:

$$u(s) = -\lambda_n \int_s^T (y(t) - y_{n-1}(t)) c e^{A(t-s)} b \, dt - \sum_{i=1}^{N} l_{in}(s)(y(t_{in}) - \alpha_{in}) = 0 \tag{4.54}$$

$$y(t) = c e^{At} x_0 + \int_0^t c e^{A(t-s)} b u(s) \, ds \tag{4.55}$$

$$y(t_{jn}) = c e^{At_i} x_0 + \int_0^T l_{jn}(s) u(s) \, ds \tag{4.56}$$

Substituting $u(s)$ into $y(t)$ and $y(t_{in})$ expressions, we have:

$$\begin{aligned}
&y(t) = c e^{At} x_0 + \\
&\int_0^t \left[c e^{A(t-s)} b \Big(-\lambda_n \int_s^T (y(\tau) - y_{n-1}(\tau)) c e^{A(\tau-s)} b \, d\tau - \sum_{i=1}^{N} l_{in}(s)(y(t_{in}) - \alpha_{in}) \Big) \right] ds = \\
&= c e^{At} x_0 - \lambda_n \int_0^t c e^{A(t-s)} b \int_s^T (y(\tau) - y_{n-1}(\tau)) c e^{A(\tau-s)} b \, d\tau \, ds - \\
&\int_0^t c e^{A(t-s)} b \sum_{i=1}^{N} l_{in}(s)(y(t_{in}) - \alpha_{in}) \, ds = \\
&= c e^{At} x_0 - \lambda_n \int_0^t \int_s^T y(t) c e^{A(t-s)} b c e^{A(\tau-s)} b \, d\tau \, ds + \lambda_n \int_0^t \int_s^T y_{n-1}(t) c e^{A(t-s)} b c e^{A(\tau-s)} b \, d\tau \, ds \\
&- \sum_{i=1}^{N} \int_0^t c e^{A(t-s)} b l_{in}(s) \, ds \, y(t_{in}) + \sum_{i=1}^{N} \int_0^t c e^{A(t-s)} b l_{in}(s) \, ds \, \alpha_{in}
\end{aligned} \tag{4.57}$$

$$\begin{aligned}
&y(t_{jn}) = c e^{At_{in}} x_0 + \int_0^T l_{jn}(s) (-\lambda_n \int_s^T (y(\tau) - y_{n-1}(\tau)) c e^{A(\tau-s)} b \, d\tau - \sum_{i=1}^{N} l_{in}(s)(y(t_{in}) - \alpha_{in})) \, ds = \\
&= c e^{At_{in}} x_0 - \lambda_n \int_0^T l_{jn}(s) \int_s^T y(\tau) c e^{A(\tau-s)} b \, d\tau ds + \lambda_n \int_0^T l_{jn}(s) \int_s^T y_{n-1}(\tau) c e^{A(\tau-s)} b \, d\tau ds \\
&- \sum_{i=1}^{N} \int_0^T l_{jn}(s) l_{in}(s) \, ds \, y(t_{jn}) + \sum_{i=1}^{N} \int_0^T l_{jn}(s) l_{in}(s) \, ds \, \alpha_{in}
\end{aligned} \tag{4.58}$$

Let the operator $L(t)(f)$ be defined as:

$$L(t)(f) = \int_0^t \int_s^T f(t) c e^{A(t-s)} b c e^{A(\tau-s)} b \, d\tau \, ds \tag{4.59}$$

And the Grammian of the linearly independent base functions $l_i(s)$:

$$G_n = \left[\int_0^t l_{in}(s) l_{jn}(s) \, ds \right]_{i,j=1}^{N} \tag{4.60}$$

In order to simplify notation, we can define a vector H_n:

$$H_n = \left(\int_0^t c e^{A(t-s)} b l_{1n}, ..., \int_0^t c e^{A(t-s)} b l_{Nn} \right) \tag{4.61}$$

And moreover:

$$\hat{y} = (y_{1n}, ..., y_{Nn})^T \tag{4.62}$$

$$C_n = [c e^{A t_{1n}} x_0, ..., c e^{A t_{Nn}} x_0]^T \tag{4.63}$$

$$\hat{L}_n(y) = [L(t_{1n})(y), ..., L(t_{Nn})(y)]^T \tag{4.64}$$

If we now rewrite the previous expressions with the new notation, we get:

$$\begin{aligned} y(t) &= c e^{At} x_0 - \lambda_n L(t)(y) + \lambda_n L(t)(y_{n-1}) - H_n \hat{y} + H_n \hat{\alpha} \\ (I + \lambda_n L(t))(y) + H_n \hat{y} &= c e^{At} x_0 + \lambda_n L(t)(y_{n-1}) + H_n \hat{\alpha} \end{aligned} \tag{4.65}$$

$$
\begin{aligned}
&y(t_{jn}) = ce^{At_{in}}x_0 - \lambda_n \int_0^T l_{jn}(s) \int_s^T y(\tau) ce^{A(\tau-s)} b \, d\tau ds + \\
&\lambda_n \int_0^T l_{jn}(s) \int_s^T y_{n-1}(\tau) ce^{A(\tau-s)} b \, d\tau ds - G_n \hat{y} + G_n \hat{\alpha} \\
&(I + G_n)\hat{y} = [ce^{At_1}x_0, ..., ce^{At_N}x_0]^T + G_n \hat{\alpha} \\
&- \lambda_n \int_0^T l_{jn}(s) \int_s^T y(\tau) ce^{A(\tau-s)} b \, d\tau ds + \lambda_n \int_0^T l_{jn}(s) \int_s^T y_{n-1}(\tau) ce^{A(\tau-s)} b \, d\tau ds
\end{aligned}
\tag{4.66}
$$

That leads to the following system of equations, Equation 4.67 and Equation 4.68:

$$
(I + \lambda_n L(t))(y) + H_n \hat{y} = \lambda_n L(t)(y_{n-1}) + H_n \hat{\alpha} + ce^{At}x_0 \tag{4.67}
$$

$$
\lambda_n \hat{L}_n(y) + (I + G_n)\hat{y} = \lambda_n \hat{L}_n(y_{n-1}) + G_n \hat{\alpha} + [ce^{At_1}x_0, ..., ce^{At_N}x_0]^T \tag{4.68}
$$

Note that as our basis functions l_{in} in equation Equation 4.37 are zero for any $t_{in} \geq s$, we can use the defined operator $L(t)(f)$ from Equation 4.59 also in equation Equation 4.68.

Since G_n is positive definite, so it is $I + G_n$, and thus its inverse exist. We can then solve $\hat{y}$ from equation Equation 4.68:

$$
\begin{aligned}
&(I + G_n)\hat{y} = \lambda_n \hat{L}_n(y_{n-1} - y) + G_n \hat{\alpha} + [ce^{At_1}x_0, ..., ce^{At_N}x_0]^T; \\
&\hat{y} = \lambda_n (I + G_n)^{-1} \hat{L}_n(y_{n-1} - y) + (I + G_n)^{-1} G_n \hat{\alpha} + (I + G_n)^{-1} C_n^T
\end{aligned}
\tag{4.69}
$$

And now we substitute it in Equation 4.67:

$$
\begin{aligned}
&(I + \lambda_n L(t))(y) + H_n (I + G_n)^{-1} \Big(\lambda_n \hat{L}_n(y_{n-1} - y) + G_n \hat{\alpha} + C_n \Big) = \lambda_n L(t)(y_{n-1}) + H_n \hat{\alpha} + ce^{At}x_0 \\
&y + \lambda_n L(t)(y - y_{n-1}) + \lambda_n H_n (I + G_n)^{-1} \hat{L}_n(y_{n-1} - y) = \\
&- H_n (I + G_n)^{-1} G_n \hat{\alpha} - H_n (I + G_n)^{-1} C_n + H_n \hat{\alpha} + ce^{At}x_0
\end{aligned}
\tag{4.70}
$$

Dividing by λ_n:

$$\begin{aligned} &\lambda_n^{-1} y + L(t)(y - y_{n-1}) + H_n (I + G_n)^{-1} \hat{L}_n (y_{n-1} - y) = \\ &- \lambda_n^{-1} \Big(H_n (I + G_n)^{-1} G_n \hat{\alpha} - H_n (I + G_n)^{-1} C_n + H_n \hat{\alpha} + c e^{At} x_0 \Big) \end{aligned} \tag{4.71}$$

The right hand side is bounded, and as λ_n goes to infinity, the right hand side goes to zero. It remains to be proved that the sequence of y_n is bounded.

Unfortunately, the result for $y_n(t)$ is only a convergence result and can't be implemented.

4.2.6 Recursive Splines - Discrete Case

We define our new working Hilbert Space as:

$$H = \{(u; x) |\, L_2[0, T] \text{x} \mathbb{R}^{N+J}\} \tag{4.72}$$

With norm

$$||(u; x)||_H^2 = \int_0^T u(t)^2 \, dt + x^T Q x \tag{4.73}$$

Where

$$\begin{aligned} Q &= \begin{pmatrix} Q_1 & 0 \\ 0 & I \end{pmatrix} \\ Q_1 &= diag(\lambda_n w_{1J}, ..., \lambda_n w_{JJ}) \end{aligned} \tag{4.74}$$

Now there are two different time sequences, one in terms of t_{in} in which n expresses the current set of data points and thus our current spline, and i a particular element of such set. The other one, $r(jJ)$, refers to the sample j of the output due to the discrete implementation.

Therefore we define:

$$\varrho_{jJ}(s) \begin{cases} ce^{A(r_{jJ}-s)}b & r_{jJ}-s \geq 0 \\ 0 & otherwise \end{cases} \tag{4.75}$$

The linear variety is:

$$\begin{aligned} &V_{x_0} = \{(u;(\rho,\gamma))|\, \rho_{jJ} = ce^{Ar_{jJ}}x_0 + \int_0^T \varrho_{jJ}u(s)\,ds, \\ &\gamma_{in} = ce^{At_{in}}x_0 + \int_0^T l_{in}u(s)\,ds\} \end{aligned} \tag{4.76}$$

And the subspace that goes through the origin, i.e. zero initial condition of the system:

$$\begin{aligned} &V_{x_0} = \{(u;(\rho,\gamma))|\, \rho_{jJ} = \int_0^T \varrho_{jJ}u(s)\,ds, \\ &\gamma_{in} = \int_0^T l_{in}u(s)\,ds\} \end{aligned} \tag{4.77}$$

To simplify notation, we define:

Vector of sampled outputs, with no initial condition, from $j = 1$ to $j = J$:

$$\hat{y} = (\int_0^T \varrho_{1j}(s)u(s)\,ds, ..., \int_0^T \varrho_{jJ}(s)u(s)\,ds)$$

Vector of sampled states, with no initial condition, from $i = 1$ to $i = N$, at output sample j:

$$\hat{x} = (\int_0^T l_{1j}(s)u(s)\,ds, ..., \int_0^T l_{Nj}(s)u(s)\,ds)$$

Vector of basis functions for all desired outputs $i = 1, ..., N$:

$$\hat{l} = (l_{1n(s)}, ..., l_{Nn}(s))^T$$

Vector of basis functions defined on the second time sequence $j = 1, ...J$

$$\hat{\varrho} = (\varrho_{1J(s)}, ..., \varrho_{jJ}(s))^T$$

Finally, the orthogonal compliment of the subspace:

$$\begin{aligned} V_0^{\perp} &= \{(v;(z,w))| < v; u >_{L_2} + < (z,w);(\rho,\gamma) >_{\mathbb{R}^{N+J}} = 0\} \\ &= \{(v;(z,w))| \int_0^T v(s)u(s)\,ds + z^T Q_1 \hat{\rho} + w^T \hat{\gamma} = 0\} \end{aligned} \tag{4.78}$$

Substituting the expressions for ρ and γ:

$$\int_0^T v(s)u(s)\,ds + z^T Q_1 \int_0^T \hat{\varrho}(s)u(s)\,ds + w^T \int_0^T \hat{l}(s)u(s)\,ds = 0 \tag{4.79}$$

We repeat the same derivation that we did on the continuous case, to obtain an expression with only integrals from 0 to T of functions times $u(s)$ and equal to zero.

As z, Q_1 and w are not functions of s, we can include them in the integral:

$$\int_0^T v(s)u(s)\,ds + \int_0^T z^T Q_1 \hat{\varrho}(s)u(s)\,ds + \int_0^T w^T \hat{l}(s)u(s)\,ds = 0 \tag{4.80}$$

As $u(s) \neq 0$:

$$\int_0^T v(s)\,ds + \int_0^T z^T Q_1 \hat{\varrho}(s)\,ds + \int_0^T w^T \hat{l}(s)\,ds = 0 \tag{4.81}$$

And thus

$$v(s) + z^T Q_1 \hat{\varrho}(s) + w^T \hat{l}(s) = 0 \tag{4.82}$$

So we can define the orthogonal compliment simply as:

$$V_0^{\perp} = \{(v;(z,w))|v(s) + z^T Q_1 \hat{\varrho}(s) + w^T \hat{l}(s) = 0\} \tag{4.83}$$

The next step, as in the continuous case, is to find the intersection of V_{x_0} and $V_0^{\perp} + p$, noted as p_p.

As we now have a discrete problem, our point p is defined in the working Hilbert space as:

$$p = (0, (\hat{y}_{n-1}, \hat{\alpha}^n)) \quad \text{where} \quad \hat{\alpha}^n = (\alpha_{1n}, ..., \alpha_{Nn})$$

$$\begin{aligned} &p_p \in V_{x_0} = \{(u; (\rho, \gamma))\} \\ &(u; (\rho, \gamma)) \in V_0^{\perp} + p\,; \quad (u; (\rho, \gamma)) - p \in V_0^{\perp} \\ &(u; (\rho, \gamma)) - (0, (\hat{y}_{n-1}, \hat{\alpha}^n)) \in V_0^{\perp} \end{aligned} \tag{4.84}$$

Giving the final expression:

$$(u; (\rho, \gamma)) - (0, (\hat{y}_{n-1}, \hat{\alpha}^n)) \in V_0^{\perp} \Rightarrow u(s) + (\rho - \hat{y}_{n-1})^T Q_1 \hat{\varrho} + (\gamma - \hat{\alpha}^n)^T \hat{l} = 0 \tag{4.85}$$

We need to solve the following system of equations:

$$u(s) = -(\rho^T - \hat{y}_{n-1}^T) Q_1 \hat{\varrho} - (\gamma^T - \hat{\alpha}^{nT}) \hat{l} \tag{4.86}$$

$$\rho_{jJ} = c e^{A r_{jJ}} x_0 + \int_0^T \varrho_{jJ} u(s)\, ds \tag{4.87}$$

$$\gamma_{in} = c e^{A t_{in}} x_0 + \int_0^T l_{in} u(s)\, ds \tag{4.88}$$

To simplify notation we can define

$$\begin{aligned} C_r &= c e^{A r_{jJ}} x_0 \\ C_t &= c e^{A t_{in}} x_0 \end{aligned} \tag{4.89}$$

So the system of equations becomes

$$u(s) = -(\rho^T - \hat{y}_{n-1}^T)Q_1\hat{\varrho} - (\gamma^T - \hat{\alpha}^{nT})\hat{l} \tag{4.90}$$

$$\rho = C_r + \int_0^T \hat{\varrho}u(s)\,ds \tag{4.91}$$

$$\gamma = ce^{At_{in}}x_0 + \int_0^T \hat{l}u(s)\,ds \tag{4.92}$$

Substituting $u(s)$ in the expressions for ρ and γ:

$$\begin{aligned}\rho &= C_r + \int_0^T \hat{\varrho}[-(\rho^T - \hat{y}_{n-1}^T)Q_1\hat{\varrho} - (\gamma^T - \hat{\alpha}^{nT})\hat{l}]\,ds \\ &= C_r - \int_0^T \hat{\varrho}\hat{\varrho}^T Q_1\rho\,ds + \int_0^T \hat{\varrho}\hat{\varrho}^T Q_1\hat{y}_{n-1}\,ds - \int_0^T \hat{\varrho}\hat{l}^T\gamma\,ds + \int_0^T \hat{\varrho}\hat{l}^T\hat{\alpha}^{nT}\,ds\end{aligned} \tag{4.93}$$

$$\gamma = C_t - \int_0^T \hat{l}\hat{\varrho}^T Q_1\rho\,ds + \int_0^T \hat{l}\hat{\varrho}^T Q_1\hat{y}_{n-1}\,ds - \int_0^T \hat{l}\hat{l}^T\gamma\,ds + \int_0^T \hat{l}\hat{l}^T\hat{\alpha}^{nT}\,ds \tag{4.94}$$

We can now take $Q_1\rho$, $Q_1\hat{y}$, γ, and $\hat{\alpha}^{nT}$ out of the integrals, as they are not functions of s.

$$\rho = C_r - \int_0^T \hat{\varrho}\hat{\varrho}^T\,dsQ_1\rho + \int_0^T \hat{\varrho}\hat{\varrho}^T\,dsQ_1\hat{y}_{n-1} - \int_0^T \hat{\varrho}\hat{l}^T\,ds\gamma + \int_0^T \hat{\varrho}\hat{l}^T\,ds\hat{\alpha}^{nT} \tag{4.95}$$

$$\gamma = C_t - \int_0^T \hat{l}\hat{\varrho}^T\,dsQ_1\rho + \int_0^T \hat{l}\hat{\varrho}^T\,dsQ_1\hat{y}_{n-1} - \int_0^T \hat{l}\hat{l}^T\,ds\gamma + \int_0^T \hat{l}\hat{l}^T\,ds\hat{\alpha}^{nT} \tag{4.96}$$

If we multiply on the left equation Equation 4.95 times $Q_1^{-1}Q_1$:

$$Q_1^{-1}Q_1\rho = C_r - \int_0^T \hat{\varrho}\hat{\varrho}^T\, ds Q_1\rho + \int_0^T \hat{\varrho}\hat{\varrho}^T\, ds Q_1\hat{y}_{n-1} - \int_0^T \hat{\varrho}\hat{l}^T\, ds\gamma + \int_0^T \hat{\varrho}\hat{l}^T\, ds\hat{\alpha}^{nT} \tag{4.97}$$

We further define new operators and express the above in matrix notation:

$$G = \int_0^T \hat{\varrho}\hat{\varrho}^T\, ds$$

$$S = \int_0^T \hat{l}\hat{l}^T\, ds$$

$$H = \int_0^T \hat{\varrho}\hat{l}^T\, ds$$

$$\begin{pmatrix} Q_1^{-1}+G & H \\ H^T & I+S \end{pmatrix} \begin{pmatrix} Q_1\rho \\ \gamma \end{pmatrix} = \begin{pmatrix} G & H \\ H^T & S \end{pmatrix} \begin{pmatrix} Q_1\hat{y}_{n-1} \\ \hat{\alpha}^n \end{pmatrix} + \begin{pmatrix} C_r \\ C_t \end{pmatrix} \tag{4.98}$$

Now we eliminate γ from the first equation by simply performing some row operations: First, we multiply the second row on the left: $(I+S)^{-1}R_2$

$$\begin{gathered} \begin{pmatrix} Q_1^{-1}+G & H \\ (I+S)^{-1}H^T & (I+S)^{-1}(I+S) \end{pmatrix} \begin{pmatrix} Q_1\rho \\ \gamma \end{pmatrix} = \\ \begin{pmatrix} G & H \\ (I+S)^{-1}H^T & (I+S)^{-1}S \end{pmatrix} \begin{pmatrix} Q_1\hat{y}_{n-1} \\ \hat{\alpha}^n \end{pmatrix} + \begin{pmatrix} C_r \\ (I+S)^{-1}C_t \end{pmatrix} \end{gathered} \tag{4.99}$$

Then, we subtract HR_2 from the first row: $R_1 - HR_2$:

$$\begin{pmatrix} Q_1^{-1} + G - H(I+S)^{-1}H^T & H - HI \\ (I+S)^{-1}H^T & I \end{pmatrix} \begin{pmatrix} Q_1\rho \\ \gamma \end{pmatrix} =$$
$$\begin{pmatrix} G - H(I+S)^{-1}H^T & H - H(I+S)^{-1}S \\ (I+S)^{-1}H^T & (I+S)^{-1}S \end{pmatrix} \begin{pmatrix} Q_1\hat{y}_{n-1} \\ \hat{\alpha}^n \end{pmatrix} + \begin{pmatrix} C_r - H(I+S)^{-1}C_t \\ (I+S)^{-1}C_t \end{pmatrix} \tag{4.100}$$

Our final expression is then:

$$\begin{pmatrix} Q_1^{-1} + G - H(I+S)^{-1}H^T & 0 \\ (I+S)^{-1}H^T & I \end{pmatrix} \begin{pmatrix} Q_1\rho \\ \gamma \end{pmatrix} =$$
$$\begin{pmatrix} G - H(I+S)^{-1}H^T & H - H(I+S)^{-1}S \\ (I+S)^{-1}H^T & (I+S)^{-1}S \end{pmatrix} \begin{pmatrix} Q_1\hat{y}_{n-1} \\ \hat{\alpha}^n \end{pmatrix} + \begin{pmatrix} C_r - H(I+S)^{-1}C_t \\ (I+S)^{-1}C_t \end{pmatrix} \tag{4.101}$$

The first row equation gives:

$$\begin{aligned} &(I + GQ_1 - H(I+S)^{-1}H^TQ_1)\rho = \\ &(GQ{-}1 - H(I+S)^{-1}H^TQ_1)\hat{y}_{n-1} + H(I+S)^{-1}\hat{\alpha}^n + C_r - H(I+S)^{-1}C_t \end{aligned} \tag{4.102}$$

Going back to the original notation:

$$\begin{aligned} &(I + \lambda_n(GQ - H_n(I+S_n)^{-1}H_n^TQ))\hat{y}_n = \\ &\lambda_n(GQ - H_n(I+S_n)^{-1}H_n^TQ)\hat{y}_{n-1} + H_n(I+S_n)^{-1}\hat{\alpha}^n + C_r - H_n(I+S_n)^{-1}C_t \end{aligned} \tag{4.103}$$

And dividing both sides by λ_n:

$$\begin{aligned} &(\lambda_n^{-1}I + (GQ - H_n(I+S_n)^{-1}H_n^TQ))\hat{y}_n = \\ &(GQ - H_n(I+S_n)^{-1}H_n^TQ)\hat{y}_{n-1} + \lambda_n^{-1}(H_n(I+S_n)^{-1}\hat{\alpha}^n + C_r - H_n(I+S_n)^{-1}C_t) \end{aligned} \tag{4.104}$$

If we define:

$$A_n = -(GQ - H_n(I+S_n)^{-1}H_n^T Q)$$
$$U_n = H_n(I+S_n)^{-1}\hat{\alpha}^n + C_r - H_n(I+S_n)^{-1}C_t$$

We get:

$$\begin{aligned}(\lambda_n^{-1} - A_n)\hat{y}_n &= -A_n\hat{y}_{n-1} + \lambda_n^{-1}U_n\,;\\ \hat{y}_n &= -(\lambda_n^{-1} - A_n)^{-1}A_n\hat{y}_{n-1} + (\lambda_n^{-1} - A_n)^{-1}\lambda_n^{-1}U_n\end{aligned} \tag{4.105}$$

And rewriting it in a less cumbersome form:

$$\hat{y}_n = -(\epsilon_n - A_n)^{-1}A_n\hat{y}_{n-1} + (\epsilon_n - A_n)^{-1}\epsilon_n U_n \tag{4.106}$$

CHAPTER 5
CONCLUSIONS, CONTRIBUTIONS AND FUTURE WORK

Chapter 5 concludes this Master Thesis with an overall view of the SlothBot project itself. First, section 5.1 summarizes the main conclusions of the project as a collection of the main results and remarks obtained. Then, section 5.2 enumerates all the contributions of this project and highlights why are they a valuable contribution for this new application of robotics. Finally, section 5.3 presents what are the next steps that need to be taken, and questions to be solved to continue the project.

5.1 Conclusions and Main Results

The main conclusions of this Master Thesis are presented as a collection of the main results obtained:

Optimization Problem to Persistify Robotic Tasks for Systems with Non Control-Affine Dynamics

Ensuring that the robot's mission will be executed during a time period longer than its battery capacity is obtained by solving the optimization problem:

$$\begin{aligned} &\underset{u}{minimize} \quad ||u-\hat{u}||^2 \\ &s.t. \\ &\quad -L_gh_1(X)u^2 \leq \gamma_1 h_1(X) + L_fh_1(X) \\ &\quad -L_gh_2(X)u \leq \gamma_2 h_1(X) + L_fh_2(X) \end{aligned} \tag{5.1}$$

Where:

- $X = [E \quad p]$ is the state vector
- L_fh_i, L_gh_i are the Lie Derivatives of the barrier function h_i

Feasibility of the General Quadratic Cost Scalar Linear and Quadratically Constrained Problem

The constraint set of the general QCSLQC problem is empty, and thus no optimal solution exists, if any of the following situations occurs:

- $a < 0, b < 0, \alpha < 0, \beta > 0 :$
  $x^* \in [\sqrt{\frac{b}{a}}, \infty) \cup [\frac{\beta}{\alpha}, -\sqrt{\frac{b}{a}}]$ Infeasible if $\frac{\beta}{\alpha} > -\sqrt{\frac{b}{a}}$
- $a < 0, b < 0, \alpha > 0, \beta > 0 :$
  $x^* \in (-\infty, -\sqrt{\frac{b}{a}}] \cup [\sqrt{\frac{b}{a}} \frac{\beta}{\alpha}]$ Infeasible if $\frac{\beta}{\alpha} < \sqrt{\frac{b}{a}}$
- $a > 0, b < 0$
- $a > 0, b > 0, \alpha > 0, \beta < 0 :$
  $x^* \in [-\sqrt{\frac{b}{a}}, \frac{\beta}{\alpha}]$ Infeasible if $\frac{\beta}{\alpha} < -\sqrt{\frac{b}{a}}$

Feasibility of the QCSLQC Problem for the SlothBot Application:

Feasibility of the quadratic constraint can be ensured by choosing sufficiently large values of γ_1, as in standard Control Barrier Functions theory. As γ_1 becomes larger, the intervals where the quadratic constraint is infeasible approach the boundaries of the safe set for the battery voltage E.

The linear constraint is always feasible in the SlothBot application, independently of the value of γ_2. Furthermore, it does not interfere with the existence of solutions for the Quadratically Constrained Quadratic Program.

Smoothing Splines for Light Intensity Model Estimation:

When using Smoothing Splines to estimate the Light Intensity function, the objective function to minimize is:

$$J(u) = \frac{1}{2}\int_0^T \rho u^2(t)dt + \frac{1}{2}(\int_0^T g(t)u(t)dt - \xi)^T \tau (\int_0^T g(t)u(t)dt - \xi) \tag{5.2}$$

Where:

- ξ are the different waypoints
- ρ controls the amount of smoothing in the output
- τ is a matrix in which each element expresses the importance for the jth output to interpolate close to ξ_i's jth component at time t_i
- $g(t)$ is a set of basis functions

$$g_i(t) \begin{cases} C^T e^{A(t_i - t)} b & t_i - t \geq 0 \\ 0 & otherwise \end{cases} \tag{5.3}$$

5.2 Contributions

The main results and contributions of this Master Thesis can be summarized as follows:

Persistification of Robotic Tasks for Non Control-affine Dynamical Systems

The term persistification expresses that the robot's mission and tasks require to be executed during long time periods, which exceed its battery total capacity. Persistification of robotic tasks has been previously studied in systems with control-affine dynamics. The longevity approach of the SlothBot project introduced a quadratic dependency between the robot's battery voltage and the control input, i.e. the control energy, what required for a revision and reformulation of the existing theory. The developed work in this thesis proves that it is possible to persistify robotic tasks even when the system dynamics possesses non-linear terms.

Analytic solution to Quadratic Cost Scalar Linear and Quadratically Constrained Problems

When formulating the optimization problem that minimizes the difference between the nominal control, which commands the robot's mission, and the safe control, that makes the system to remain forward invariant, it results in a new class of problems. These problems have been noted as Quadratic Cost Scalar Linear and Quadratically Constrained Problems

(QCSLQC). Convex optimization algorithms and relaxations do not constitute an appropriate solution to this problem, as their computation and energy demands make them incompatible with the conservative energy use required to maximize the longevity of the SlothBot.

Smoothing Splines for Light Intensity Model Estimation

Smoothing Splines were presented in the literature as a trajectory planning technique. Using an arbitrary system and modifying the control input applied, the output of the system can be driven close to different waypoints and specific interpolation times. In this thesis it has been shown how this technique can also be used for model estimation, in particular, to estimate the Light Intensity function using different measurements that are taken by the robot periodically.

Recursive Smoothing Splines Theory and Mathematical Fundamentals

When the Smoothing Splines are used to estimate the Light Intensity Model, as new samples are added, the computation time grows exponentially, due to the Grammian computations and matrices inversions required in the method. This is also a problem that all Machine Learning and Convex optimization algorithms have to face, limiting their efficiency and application. We aimed to use Smoothing Splines in a recursive way, using each new data set to modify the previous solution instead of solving a new problem. With this approach the size of the problem would remain constant and thus the computation complexity of the problem would always be the same. Finding an alternative approach to model estimation, able to keep the size of the problem constant when new samples are added, would open a new horizon in model identification techniques. Unfortunately, only convergence results have been obtained in this thesis.

Smaller and easier to assembly and maintain SlothBot robot

Finally, while all the theoretical work and research presented in this thesis was being done, a new, smaller and more maintenance-friendly robotic platform was also being developed. This new version of the SlothBot will be the one used to test all the developed algorithms and continue exploring the applicability of our proposed robotic platform for environmental research.

5.3 Future Work

After going over all the results and achievements of this project, the immediate following up problems and future developments that stand out are:

Test the Analytic Solution to the QCSLQC Problem in the Real SlothBot

Due to time constraints and the fact that the only available SlothBot robot was hanged at the Atlanta Botanical Garden until July 2021, it was not possible to test the proposed solution in the actual robot. This will be one of the first tasks to be completed when the new version of the robot is deployed at its new location.

Development of Multi-Robot Algorithms for a Collaborative Approach to Environmental Research

One of the reasons why a new, smaller, and easier to manufacture version of the robot was needed is because in a future stage of the project it would be of great interest to explore the application of swarm robotics to environmental research. Using robots in a collaborative way has proven to solve coverage problems in a more efficient way, but its application in an outdoors habitat for environmental exploration and surveillance needs still to be explored.

Find an Implementable Solution to the Recursive Smoothing Splines Problem

Unfortunately, as it has been previously mentioned, so far the only results that have been obtained for the recursive smoothing splines approach are convergence results. These results show that the problem is solvable but unfortunately they can't be programmed and tested in simulation or the actual robot. Finding an explicit solution is the only missing piece to solve and close this problem.

Appendices

APPENDIX A
LIE DERIVATIVES

Lie Derivatives are widely used in Systems and Controls theory, specially in nonlinear systems [23], for example for feedback linearization.

Particularly for dynamical systems, the Lie derivative of a map h along the vector field f at the point $x \in M$ is:

$$L_f h(x) = \frac{d}{dt} h(\phi_t^f(x))\Big|_{t=0} \tag{A.1}$$

Where $\phi_t^f(x)$ defines the flow of the vector field f at time t.

Simply applying the chain rule, we obtain the alternative notation:

$$L_f h(x) = h'(x) f(x) \tag{A.2}$$

Where h' is the Jacobian matrix of h. It is important to note that the Lie derivative $L_f h(x)$ has the same dimensions as the original map h.

As a consequence, if we repeat the process in a recursive way we obtain the general equation for solving higher order Lie derivatives:

$$L_f^k h(x) = \frac{\partial L_f^{k-1} h(x)}{\partial x} f(x) \tag{A.3}$$

With $L_f^0 h(x) = h(x)$.

If we have also a vector field g with flow $\phi_s^g f(x)$ at time s, the mixed Lie derivative is defined as:

$$L_g L_f h(x) = \frac{\partial L_f h(x)}{\partial x} g(x) \tag{A.4}$$

Before concluding this section, it is important to note how our map and vector fields are defined: $h : M \to \mathbb{R}^p$, $f : M \to \mathbb{R}^n$, and $g : M \to \mathbb{R}^n$. With $M \subseteq \mathbb{R}^n$.

APPENDIX B
LIPSCHITZ CONTINUITY

Lipschitz continuity is commonly used in Control theory to proof that a unique solution to our system exists. Given a system:

$$\begin{aligned} \dot{x} &= f(x) \\ x(0) &= x_0 \end{aligned} \tag{B.1}$$

If $f(\cdot)$ is continuous (C^0), a solution will exist, but C^0 is not enough to guarantee uniqueness. This can be easily proved by the reader using the system $\dot{x} = x^{\frac{1}{3}}$.

The sufficient condition for uniqueness of solutions is Lipschitz continuity, which is more restrictive than C^0. A function $f(\cdot)$ is *locally Lipschitz* if every point x^0 has a neighborhood where Equation B.2 holds for all x, y in such neighborhood for some L.

$$|f(x) - f(y)| \leq L|x - y| \tag{B.2}$$

Furthermore, if a function $f(\cdot)$ is continuously differentiable (C^1), then it is *locally Lipschitz*. The converse is not true, an example of this is the saturation function.

Regarding the uniqueness of solutions:

- If $f(\cdot)$ is locally Lipschitz $\rightarrow$ existence and uniqueness of solutions is guaranteed on $[0, t_f)$
- If $f(\cdot)$ is globally Lipschitz $\rightarrow$ existence and uniqueness of solutions is guaranteed on $[0, \infty)$

Finally, a function is *globally Lipschitz* if Equation B.2 holds $\forall x, y \in \mathbb{R}^n$, i.e. we can use the same L everywhere in $\mathbb{R}^n$. Once again, it is important to emphasize that *globally Lipschitz* does not imply either C^1.

APPENDIX C
ALIGNMENT WITH SUSTAINABLE DEVELOPMENT GOALS (SDGS)

As we presented in the introduction of this Master Thesis, the aim of the project was to develop a robotic platform, and its control algorithms, that would help environmental researchers in tree-canopies ecosystems characterization. Tree-tops are profoundly important as they are where solar energy becomes carbohydrates to fuel entire ecosystems, they are home for more than fifty percent of our biodiversity and they are where the most crucial data for environmental indicators are gathered. These indicators are commonly known as Footprints, such as the Carbon Footprint, the Ecological Footprint, and the Biodiversity Footprint, and they constitute the only qualitative measure that is currently used to analyze the environmental situation, and making decisions and taking actions that would help to improve it.

The reader can see how this motivation has a straightforward connection with the Sustainable Development Goals (SDGs) number 13 ”Climate action”, and number 15 ”Life on land”.

Indirectly, there is also a relation with the SDG number 3 ”Good health and well-being”. A significant amount of the actions that environmental researchers propose pursue the goal of improving the quality of life of all individuals. Science supports how a cleaner and better air quality has a positive impact on humans longevity and overall individuals health [24] [25] [26], and that is why the SlothBot’s mission contributes to the well-being of our society.

Finally, the alignment of the SlothBot project with the SDGs can also be extended to the SDG number 9 ”Industry, innovation, and infrastructure”. The use of robots for environmental research constitutes a completely new application for robotics, broadening the extension of how robots can positively impact humans lives. It also brings up new and challenging problems of interest for researchers of all technical areas, from computer scientists to control theorists. Solving new problems is the only mean to further extend the current boundaries of knowledge and continue developing an enlightened scientific community.

REFERENCES

[1] Y. E. Gennaro Notomista and M. Egerstedt, "The SlothBot: A Novel Design for a Wire-Traversing Robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, 2019.

[2] M. B. Egerstedt and C. F. Martin, "Optimal trajectory planning and smoothing splines," *Automatica*, vol. 37, no. 7, 2001.

[3] V. Lawley, M. Lewis, K. Clarke, and B. Ostendorf, "Site-based and remote sensing methods for monitoring indicators of vegetation condition: An australian review," *Ecological Indicators*, vol. 60, pp. 1273–1283, Mar. 2016.

[4] D. Seidel, S. Fleck, C. Leuschner, and T. Hammett, "Review of ground-based methods to measure the distribution of biomass in forest canopies," *Annals of Forest Science*, vol. 68, pp. 225–244, Mar. 2011.

[5] A. Adhikari, M. Kumar, S. Agrawal, and R. S, "An integrated object and machine learning approach for tree canopy extraction from uav datasets," *J Indian Soc Remote Sens*, vol. 49, pp. 471–478, Mar. 2021.

[6] M. Rahimi, R. Pon, W. J. Kaiser, G. S. Sukhatme, D. Estrin, and M. Srivastava, "Adaptive sampling for environmental robotics," *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA*, 2004.

[7] D. R. B. Vadiraj Hombal Arthur Sanderson, "Multiscale adaptive sampling in environmental robotics," *IEEE Conference on Multisensor Fusion and Integration*, 2010.

[8] M. Duarte, J. Gomes, V. Costa, T. Rodrigues, F. Silva, V. Lobo, M. M. Marques, S. M. Oliveira, and A. L. Christensen, "Application of swarm robotics systems to marine environmental monitoring," *IEE: OCEANS - Shanghai*, 2016.

[9] E. Lughofer and M. Sayed-Mouchaweh, "Autonomous data stream clustering implementing split-and-merge concepts – towards a plug-and-play approach," *Information Sciences*, vol. 304, no. 54-79, 2015.

[10] H. Kälviäinen, P. Hirvonen, L. Xu, and E. Oja, "Probabilistic and non-probabilistic hough transforms: Overview and comparisons," *Image and Vision Computing*, vol. 13, no. 4, 1995.

[11] K. G. Derpanis, "Overview of the ransac algorithm," 2010.

[12] I. D. Landau, R. Lozano, M. M'Saad, and A. Karimi, "Adaptive control: Algorithms, analysis and applications," *Springer*, 2011.

[13] G. Notomista and M. Egerstedt, "Persistification of robotic tasks," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 2, 2021.

[14] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," *IEEE 18th European Control Conference (ECC)*, 2019.

[15] M. Z. Romdlony and B. Jayawardhana, "Stabilization with guaranteed safety using control lyapunov–barrier function," *Automatica*, vol. 66, pp. 39–47, 2016.

[16] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 6271–6278.

[17] S. Coogan, "Ece 6552: Non-linear systems and controls lecture notes," *Georgia Institute of Technology, School of Electrical and Computer Engineering*, 2021.

[18] A. Konar and N. D. Sidiropoulos, "Hidden convexity in qcqp with toeplitz-hermitian quadratics," *IEEE Signal Processing Letters*, vol. 22, no. 10, 2015.

[19] S. Kim and M. Kojima, "Second order cone programming relaxation of nonconvex quadratic optimization problems," *Optimization Methods and Software*, vol. 15 (3-4), pp. 201–224, 2001.

[20] H. Wolkowicz, R. Saigal, and L. Vandenberghe, *Handbook of semidefinite programming: theory, algorithms, and applications*. Springer Science & Business Media, 2012.

[21] K. M. Anstreicher, "Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming," *Journal of Global Optimization*, vol. 43 (2-3), pp. 471–484, 2009.

[22] D. Liberzon, "Calculus of variations and optimal control theory: A concise introduction," *Princeton University Press*, 2012.

[23] K. Robenack, "Computation of multiple lie derivatives by algorithmic differentiation," *Journal of Computational and Applied Mathematics*, vol. 123, no. 2, 2008.

[24] R. M. Kaplan and A. Milstein, "Contributions of health care to longevity: A review of 4 estimation methods," *The Annals of Family Medicine*, vol. 17, no. 3, pp. 267–272, 2019.

[25] J. Lv, W. Wang, T. Krafft, Y. Li, F. Zhang, and F. Yuan, "Effects of several environmental factors on longevity and health of the human population of zhongxiang, hubei, china," *Biological trace element research*, vol. 143, no. 2, pp. 702–716, 2011.

[26] T. Takano, K. Nakamura, and M. Watanabe, “Urban residential environments and senior citizens’ longevity in megacity areas: The importance of walkable green spaces,” *Journal of Epidemiology & Community Health*, vol. 56, no. 12, pp. 913–918, 2002.