



UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E
SISTEMAS

Vinícius Moreno Sanches

**Embedded Nonlinear Optimization Tools for Real-Time Model Predictive Control:
Application for renewable microgrids**

Florianópolis
2026

Vinícius Moreno Sanches

**Embedded Nonlinear Optimization Tools for Real-Time Model Predictive Control:
Application for renewable microgrids**

Dissertação submetida ao Programa de Pós-Graduação
em Engenharia de Automação e Sistemas da Universi-
dade Federal de Santa Catarina para a obtenção do tí-
tulo de mestre em Engenharia de Automação e Sistemas.
Orientador: Prof. Marcelo Menezes Morato, PhD
Co-orientador: Prof. Julio Elias Normey Rico, PhD

Florianópolis
2026

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Sanches, Vinícius Moreno

Embedded Nonlinear Optimization Tools for Real-Time
Model Predictive Control: Application for renewable
microgrids / Vinícius Moreno Sanches ; orientador, Marcelo
Menezes Morato, coorientador, Julio Elias Normey-Rico,
2026.

107 p.

Dissertação (mestrado) - Universidade Federal de Santa
Catarina, Centro Tecnológico, Programa de Pós-Graduação em
Engenharia de Automação e Sistemas, Florianópolis, 2026.

Inclui referências.

1. Engenharia de Automação e Sistemas. 2. Microrrede. 3.
Energia renovável. 4. Controle preditivo baseado em
modelo. 5. Linear a Parâmetros Variantes. I. Morato,
Marcelo Menezes. II. Normey-Rico, Julio Elias. III.
Universidade Federal de Santa Catarina. Programa de Pós
Graduação em Engenharia de Automação e Sistemas. IV. Título.

Vinícius Moreno Sanches

**Embedded Nonlinear Optimization Tools for Real-Time Model Predictive Control:
Application for renewable microgrids**

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Eduardo Camponogara, PhD.
Universidade Federal de Santa Catarina

Prof. Guilherme Vianna Raffo, PhD.
Universidade Federal de Minas Gerais

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia de Automação e Sistemas.



Documento assinado digitalmente
Julio Elias Normey Rico
Data: 03/03/2026 11:11:50-0300
CPF: ***,840.859-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof. Julio Elias Normey-Rico, PhD.
Coordenador do Programa



Documento assinado digitalmente
Marcelo Menezes Morato
Data: 03/03/2026 10:18:13-0300
CPF: ***,098.789-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof. Marcelo Menezes Morato, PhD
Orientador

Este trabalho é dedicado ao Vinícius de 2022, que decidiu se arriscar na carreira acadêmica e que, apesar de tudo, não desistiu.

ACKNOWLEDGEMENTS

Neste breve texto, tento expressar de forma singela minha gratidão a todos que fizeram parte, de algum jeito, da minha formação neste último ano e meio de universidade.

Primeiramente, agradeço aos meus colegas pelo trabalho em conjunto, especialmente à Cecília, à Samira e ao Johannes. Também agradeço a todos que colaboraram comigo na publicação de trabalhos, em especial ao Prof. Marcus Americano, ao Prof. Ariel Medero e ao Prof. Olivier Sename.

Agradeço enormemente às peças mais importantes dessa trajetória: meu orientador Prof. Marcelo, fonte de inspiração e de ajuda em tudo o que eu preciso (e até mais), e ao meu coorientador Prof. Julio, que é referência não apenas para mim, mas para todos que seguem na área de controle.

Também agradeço a todos os amigos com quem tive (ou não) contato durante esses 18 meses. Em especial, agradeço ao Pedro, à Laura, ao Yuri, ao Victor e ao Matheus.

À minha namorada, colega, parceira de projetos e melhor amiga Juliana e sua (também minha) família, em especial à Luísa e ao Daniel, que dividiram moradia conosco por 6 ótimos meses. Também deixo uma menção ao meu afilhado Pietro que, mesmo pequeno, já é muito amado.

Estendo a gratidão a toda a minha família e, em especial, aos meus pais Islaine e Sergio, e ao meu irmão Gustavo, que me apoiam em todos os momentos de minha vida e, em alguns, inclusive fazendo parte (obrigado, pai, por participar do projeto do barco).

Faço questão de registrar, também, meu agradecimento à todas as pessoas "invisíveis" da universidade (funcionários e terceirizados) que discretamente fazem a engrenagem continuar rodando, mesmo sem receberem o devido reconhecimento. Em particular, agradeço a todos os trabalhadores por trás do Restaurante Universitário (RU), que me alimentam diariamente desde 2018.

Por último, este trabalho foi realizado com o apoio da Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina (FAPESC), por meio de uma bolsa de Mestrado concedida em agosto de 2024. A pesquisa foi desenvolvida no âmbito do laboratório INCT CAPE, que também conta com o suporte do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). A minha ida a congressos também foi possível graças ao auxílio do GIPSA-lab do instituto Grenoble-INP na França.

*“Todos esses que aí estão
Atravancando meu caminho,
Eles passarão...
Eu passarinho!”*
(Mario Quintana)

RESUMO

Microrredes desempenham um papel fundamental no contexto da transição energética. A literatura específica indica o Controle Preditivo Baseado em Modelo como algoritmo chave para a operação adequada de gestão de demandas e cargas, operando como Controladores Centrais de Microrredes. No entanto, estes algoritmos encontram uma barreira significativa em relação à sua implementação para sistemas não lineares com requisitos de tempo real (período de amostragem curto e dinâmica rápida): a elevada carga computacional da otimização inerente impede a aplicação em microcontroladores embarcados com menor capacidade numérica. Para abordar esse tópico, exploramos a estrutura Linear com Parâmetros Variantes para desenvolver um *solver* dedicado, que converte o problema não linear em uma formulação quadrática. O algoritmo em questão combina elementos de representação exata, através de inclusões lineares, com a simplicidade de resolução de otimização por meio de estruturas quadradas. O *solver* proposto, denominado LPV-PILLOT, tem como base o método de ponto interior, além de outras técnicas consagradas (barreira logarítmica, warm-start, contingência, ingredientes terminais, restrições apertadas, etc.). Dessa forma, possibilitamos a rápida convergência do problema de otimização e a determinação da lei de controle preditivo não linear, além de discutirmos sobre garantias teórica de estabilidade e robustez correspondentes. O *solver* é avaliado através de exemplos numéricos e experimentais, cujos resultados destacam sua adequação para sistemas não lineares operando sob tempos de amostragem estritos (na ordem dos milissegundos), embarcados em *hardware* de baixo custo. Não obstante, o algoritmo proposto é validado em uma microrrede renovável física de escala reduzida, demonstrando sua viabilidade prática.

Palavras-chave: Microrrede. Energia Renovável. Hidrogênio Verde. Controle Preditivo Baseado em Modelo. Linear a Parâmetros Variantes. Método de Pontos Interiores. Otimização em Tempo Real.

RESUMO EXPANDIDO

INTRODUÇÃO

O nosso planeta enfrenta hoje um problema urgente sem precedentes: a mudança climática acelerada. Dentre diversos fatores causadores, a queima de combustíveis fósseis gera poluentes que desequilibram o ecossistema global e ameaçam a vida na Terra.

Face a esse contexto, urge-se a importância de um processo de transição energética, a fim de substituir a utilização majoritária de fontes poluentes, por fontes de energia renovável. Desta forma, as microrredes surgem como uma tecnologia chave, que une essas fontes de energia renovável (DERs), aos sistemas de armazenamento (ESS) e à rede em uma solução integrada e única, que tem como objetivo o gerenciamento seguro e ótimo desses sistemas.

Tais microrredes carregam consigo diferentes desafios de controle associados. Dentre eles, a intermitência e imprevisibilidade das energias renováveis, que são perturbações complexas que afetam o sistema. Além disso, seus componentes possuem forte dinâmica não-linear (como as baterias e células de combustível, por exemplo). Sendo assim, técnicas de controle avançado são requeridas a fim de lidar com as dinâmicas complexas do sistema, restrições operacionais, lógicas e o objetivo econômico das microrredes.

Portanto, a literatura sugere que o Controle Preditivo Baseado em Modelo (*Model Predictive Control* - MPC) desponta como uma das abordagens mais adequadas para gerenciamento ótimo de energia das microrredes (Bordons; Garcia-Torres; Ridao, 2020). O algoritmo MPC utiliza o modelo do sistema para prever seu comportamento e determinar a ação de controle ótima a cada instante de tempo (amostra de controle). Tal sistema de gerenciamento é denominado, em inglês, de *Microgrid Central Controller* (MGCC).

Apesar da ampla difusão do uso de modelos linearizados, a maioria dos sistemas práticos da vida real possuem dinâmica não-linear. Nesse contexto, o algoritmo de MPC Não-Linear (NMPC) é a abordagem ideal, levando em consideração essas características em seu modelo de predição. No entanto, apesar de sua eficácia e optimalidade, tais algoritmos exigem a solução de um problema de Programação Não-Linear (NLP) a cada amostra de tempo. Tal problema é computacionalmente custoso e normalmente acarreta em alto tempo de cômputo, que acaba se tornando uma barreira para a sua implementação em sistemas embarcados com microcontroladores de baixo custo (com pouco poder de processamento e memória), principalmente com requisitos de tempo real estrito.

OBJETIVOS

Essa pesquisa, portanto, tem como objetivo **diminuir a lacuna entre a teoria de algoritmos NMPC e sua aplicação em sistemas embarcados**, com foco em microcontroladores de baixo custo (programados em C/C++) e em aplicações de gerenciamento ótimo de energia para microrredes (MGCC).

Para isso, duas hipóteses são traçadas. Em primeiro lugar, consideramos que é possível transformar o problema de otimização original em um problema mais simples de resolver, sem perda de acurácia do modelo (isto é, sem aproximações). Para isso, utiliza-se a **abordagem de sistemas**

lineares a parâmetros variáveis (LPV) por meio da técnica de incorporações quase-LPV (qLPV) para transformar o problema NLP em uma sequência de problemas convexos de Programação Quadrática (QP).

Ademais, os *solvers* mais utilizados na indústria e na academia (como o CasADi, o IPOPT, o OSQP, o TinyMPC e o qpOASES, por exemplo) trazem um gargalo computacional intrínseco, necessitam muitas vezes de alocação de memória dinâmica, rotinas de diferenciação automática e possuem dependência de outras bibliotecas de álgebra linear. Tais aspectos acabam criando uma barreira que torna impraticável a utilização de tais *solvers* em muitas aplicações no setor industrial.

Face a isso, e de acordo com a literatura específica (Boyd, 2004), essa pesquisa traça a hipótese de que o **Método de Ponto Interior Primal-Dual com Barreiras Logarítmicas é a opção mais adequada para o desenvolvimento de um algoritmo de otimização embarcada eficiente.**

Enfim, a contribuição principal dessa pesquisa está no desenvolvimento de um *solver* "plug-and-play" para NMPC baseado na estrutura LPV, e no método de ponto interior com barreira logarítmica, especificamente para sistemas embarcados em tempo real. Como contribuições específicas, esse trabalho embarca o *solver* em um microcontrolador de baixo custo (ESP32) utilizando a linguagem C/C++ e testa sua eficácia em uma aplicação de microrrede renovável.

METODOLOGIA

Considerando o contexto geral apresentado, a contribuição principal desse trabalho a fim de solucionar o problema estabelecido é o desenvolvimento do LPV-PILOT, um novo *solver open-source* dedicado que se baseia no ferramental LPV para representação de modelos não-lineares e no método do ponto interior com barreira logarítmica para otimização embarcada eficiente. Tal *solver* constitui uma solução completa, desde a modelagem não-linear do sistema até a solução de controle ótimo integrada e rápida.

Primeiramente, esse *solver* possui a formulação de um esquema de NMPC LPV estabilizante, por meio de uma função custo que penaliza o erro de seguimento de referência, esforço de controle e custo terminal, sujeito às restrições do modelo LPV, e às restrições operacionais e terminais do sistema. Nesse arcabouço, utilizam-se variáveis de agendamento variantes no tempo a fim de representar a dinâmica não-linear do sistema durante o horizonte de predição. A fim de diminuir o custo computacional, as matrizes de igualdade do problema são pre-computadas *offline*. Além disso, a fim de restringir as incertezas do modelo e aumentar a robustez da solução, uma técnica de estreitamento de restrições é aplicada na predição do MPC.

Para lidar com as restrições do problema, adotamos a estratégia da barreira logarítmica. Essa barreira é uma função que penaliza a violação de restrição e, ao ser adicionada à função custo irrestrita, permite a reformulação do problema original, criando um problema auxiliar matematicamente irrestrito, mas cuja minimização resulta em uma solução forçadamente restrita (garantindo admissibilidade).

O problema (quadrático) é, então, resolvido iterativamente via direção de descenso, por meio de dois laços:

- Laço interno: utiliza o método de Newton para resolver o problema auxiliar.
- Laço externo: aumenta o peso da barreira, a fim de ajustar o problema auxiliar.

Após um número suficiente de iterações, a solução do problema auxiliar irrestrito converge para a solução do problema original restrito. No entanto, este trabalho se baseia no esquema de *Fast MPC* (Wang; Boyd, 2009) e fixa o parâmetro de barreira de forma a acelerar a convergência da solução, eliminando a dependência do laço externo e obtendo uma resposta subótima. A literatura específica mostra que tal resposta é muito próxima da ótima global e, na maioria dos casos, é suficientemente precisa para aplicações de controle, resultando em um desempenho dinâmico praticamente idêntico.

Outrossim, o *solver* usufrui de diferentes técnicas para garantir velocidade de convergência e robustez:

- Vetor de decisão estendido: a variável de decisão é formulada para conter os estados e entradas do sistema, simplificando as operações *online* do *solver*,
- Ganho agendado: para a estimativa da trajetória de agendamento da predição do MPC, utiliza-se a técnica do ganho congelado, que é mais leve computacionalmente,
- *Warm-start*: O *solver* é inicializado com a solução deslocada do passo de tempo anterior, diminuindo drasticamente a quantidade de iterações para convergência.
- Decomposição LDL^T customizada: utilizada para o cômputo rápido da direção de descenso, evitando o cálculo de raízes quadradas e explorando a esparsidade das matrizes.
- Estratégia de controle *fallback*: alternativa de controle segura para garantir resposta em tempo real, sem violação do tempo de amostragem.

RESULTADOS E DISCUSSÃO

O *solver* desenvolvido foi, então, testado em diferentes experimentos. Primeiramente, o algoritmo foi validado em uma simulação numérica avaliando o controle de um sistema MIMO de rotor duplo (TRMS), obtendo uma precisão de controle para seguimento de referências senoidais, com tempo de cômputo médio de 0.51 ms. Tal resultado demonstra a capacidade do *solver* em lidar de maneira precisa com a dinâmica não-linear de um sistema complexo, mantendo rápida convergência.

Em seguida, o algoritmo foi embarcado em um ESP32 utilizando linguagem C/C++, e um experimento *hardware-in-the-loop* foi realizado, avaliando o controle de vôo de um helicóptero automático. Os resultados apresentaram alta precisão de seguimento de referência e tempo de execução médio de 12.23 ms.

No terceiro experimento, comparamos o LPV-PILOT com o CasADi a fim de avaliar a performance do *solver* face ao estado da arte em otimização não-linear. Os resultados mostraram que o LPV-PILOT obteve uma resposta dinâmica praticamente idêntica à do CasADi, mas com tempo de cômputo 14.4% menor.

Posteriormente, o *solver* foi testado em um veleiro de 32 pés movido a motor, com o propósito de seguimento de trajetória. Os resultados apontam uma precisão de seguimento de referência de menos de 20 metros de erro e tempo de cômputo médio de 39.86 ms. Este resultado evidencia

a capacidade do *solver* de operar em sistemas físicos do mundo real, sujeitos a requisitos estritos de tempo real e perturbações severas (correnteza marítima, ondas e vento).

Por fim, o algoritmo foi testado em uma microrrede renovável em escala, desenvolvida no laboratório INCT-CAPE da UFSC (Universidade Federal de Santa Catarina) e os resultados demonstraram um RMS (Raiz da média dos quadrados - *Root Mean Square*) de 15.36% para o erro de seguimento de referência de nível da bateria, 0.68 mm para o nível de hidrogênio verde e de 59.91 W para o erro no cumprimento da demanda. O tempo de cômputo médio ficou entre 9 ms e 10 ms. Tal resultado evidencia o caráter prático, agilidade e eficácia da solução proposta.

CONSIDERAÇÕES FINAIS

Essa dissertação traz como contribuição principal o desenvolvimento do LPV-PILOT, um *solver open-source* baseado no método do ponto interior com barreira logarítmica e na abordagem LPV. O algoritmo usufrui de diversas técnicas para assegurar rápida convergência e robustez, que o torna adequado para sistemas microcontrolados com requisitos estritos de tempo real. O *solver* foi validado em simulações numéricas, *hardware-in-the-loop*, comparações com o estado da arte e validações experimentais em sistemas físicos do mundo real, onde sua eficácia, precisão e agilidade foi comprovada.

Como contribuição científica, esse trabalho diminui a distância entre a teoria avançada de controle preditivo não-linear e aplicações do mundo real, especialmente em sistemas de energia renovável. Por fim, de maneira prática, contribuimos ao proporcionar uma ferramenta *ready-to-use* para engenheiros(as) implementarem controle ótimo em *hardware* de baixo custo.

Palavras-chave: Microrrede. Energia Renovável. Hidrogênio Verde. Controle Preditivo Baseado em Modelo. Linear a Parâmetros Variantes. Método de Pontos Interiores. Otimização em Tempo Real.

ABSTRACT

Microgrids play a fundamental role in the context of the energy transition. Literature widely identifies Model Predictive Control as a key algorithm for the adequate management of demands and loads, operating as Microgrid Central Controllers. However, these algorithms encounter a significant barrier regarding their deployment in real-time nonlinear systems (short sampling period and fast dynamics): the high computational burden of the inherent optimization hinders their application in embedded microcontrollers with limited numerical resources. To address this issue, we explore the Linear Parameter-Varying (LPV) framework to develop a dedicated solver that converts the nonlinear problem into a quadratic formulation. This algorithm combines elements of exact representation, through linear inclusions, with the simplicity of solving optimization problems via quadratic structures. The proposed solver, named `LPV-PILOT`, employs an interior-point method alongside other optimization techniques (log-barrier, warm-start, fallback, terminal ingredients, constraint tightening, etc.). Thus, we enable fast convergence of the optimization problem and the determination of a nonlinear predictive control law, while discussing theoretical stability and robustness guarantees. The solver is evaluated under both numerical and experimental scenarios, and the results highlight its suitability for nonlinear systems operating under strict sampling times on low-cost embedded hardware. Furthermore, the proposed algorithm is validated on a physical small-scale renewable microgrid, demonstrating its practical viability.

Keywords: Microgrid. Renewable Energy. Green Hydrogen. Model Predictive Control. Linear Parameter Varying. Interior Point Method. Real-Time Optimization.

RÉSUMÉ

Les micro-réseaux jouent un rôle fondamental dans le contexte de la transition énergétique. La littérature identifie largement la Commande Prédictive par Modèle comme un algorithme clé pour la gestion adéquate des demandes et des charges, opérant en tant que Contrôleurs Centraux de Micro-réseaux. Cependant, ces algorithmes rencontrent un obstacle majeur quant à leur déploiement dans des systèmes non linéaires en temps réel (période d'échantillonnage courte et dynamique rapide) : la charge de calcul élevée de l'optimisation inhérente entrave leur application dans des microcontrôleurs embarqués aux ressources numériques limitées. Pour y remédier, nous explorons le cadre Linéaire à Paramètres Variants pour développer un solveur dédié qui convertit le problème non linéaire en une formulation quadratique. Cet algorithme combine des éléments de représentation exacte, via des inclusions linéaires, avec la simplicité de résolution de l'optimisation par des structures quadratiques. Le solveur proposé, nommé *LPV-PILOT*, emploie une méthode de points intérieurs ainsi que d'autres techniques d'optimisation (barrière logarithmique, *warm-start*, stratégie de repli, ingrédients terminaux, resserrement des contraintes, etc.). Ainsi, nous permettons la convergence rapide du problème d'optimisation et la détermination d'une loi de commande prédictive non linéaire, tout en discutant des garanties théoriques de stabilité et de robustesse. Le solveur est évalué à travers des scénarios numériques et expérimentaux, et les résultats soulignent sa pertinence pour des systèmes non linéaires opérant sous des temps d'échantillonnage stricts sur du matériel embarqué à faible coût. En outre, l'algorithme proposé est validé sur un micro-réseau renouvelable physique à petite échelle, démontrant sa viabilité pratique.

Mots-clés: Micro-réseau. Énergie Renouvelable. Hydrogène Vert. Commande Prédictive par Modèle. Linear Parameter Varying. Méthode de Points Intérieurs. Optimisation en Temps Réel.

RESUMEN

Las microrredes desempeñan un papel fundamental en el contexto de la transición energética. La literatura identifica ampliamente el Control Predictivo basado en Modelos (MPC) como un algoritmo clave para la gestión adecuada de la demanda y las cargas, operando como Controlador Central de Microrredes. Sin embargo, estos algoritmos enfrentan un obstáculo mayor para su implementación en sistemas no lineales en tiempo real (periodo de muestreo corto y dinámica rápida): la elevada carga computacional de la optimización inherente dificulta su aplicación en microcontroladores embebidos con recursos digitales limitados. Para abordar este problema, exploramos el marco de Sistemas Lineales de Parámetros Variantes (LPV) con el fin de desarrollar un resolutor (*solver*) dedicado que convierta el problema no lineal en una formulación cuadrática. Este algoritmo combina elementos de representación exacta, mediante inclusiones lineales, con la simplicidad de resolución de la optimización a través de estructuras cuadráticas. El resolutor propuesto, denominado `LPV-PILOT`, emplea un método de puntos interiores junto con otras técnicas de optimización (barrera logarítmica, *warm-start*, estrategia de repliegue, ingredientes terminales, endurecimiento de restricciones, etc.). De este modo, permitimos la convergencia rápida del problema de optimización y la determinación de una ley de control predictivo no lineal, al mismo tiempo que se discuten las garantías teóricas de estabilidad y robustez. El resolutor se evalúa a través de escenarios numéricos y experimentales, y los resultados destacan su relevancia para sistemas no lineales que operan bajo tiempos de muestreo estrictos en *hardware* embebido de bajo costo. Además, el algoritmo propuesto se valida en una microrred renovable física a pequeña escala, demostrando su viabilidad práctica.

Palabras clave: Microrred. Energía Renovable. Hidrógeno Verde. Control Predictivo basado en Modelos. Parámetros Lineales Variantes. Método de Puntos Interiores. Optimización en Tiempo Real.

LIST OF FIGURES

Figure 1 – Example of a generic microgrid structure.	26
Figure 2 – Comparison between state-of-the-art NMPC and the proposed LPV MPC approach.	28
Figure 3 – Hierarchical control levels in microgrids.	33
Figure 4 – TRMS structure (Rotondo; Nejjari; Puig, 2013).	53
Figure 5 – Twin Rotor MIMO System control using LPV-PILOT.	54
Figure 6 – Processor-in-the-loop scheme.	56
Figure 7 – PIL scheme: ESP32 microcontroller and SimuLink.	56
Figure 8 – Helicopter landing between buildings using LPV-PILOT.	57
Figure 9 – Helicopter motion along the X and Y coordinates.	58
Figure 10 – Computational time for helicopter landing control.	58
Figure 11 – Inverted pendulum on a cart scheme.	59
Figure 12 – Pendulum dynamics comparison.	62
Figure 13 – Cost function evolution for the inverted pendulum benchmark.	62
Figure 14 – Computational time comparison ($\text{CasADi} \times \text{LPV-PILOT}$ for Inverted Pendulum).	63
Figure 15 – Matina III: a 32-foot sailboat (<i>Brasilia 32</i>).	63
Figure 16 – Motor-helm wheel coupling illustrative scheme.	63
Figure 17 – Rudder angle measurement using potentiometer.	64
Figure 18 – Electrical circuit and ESP32 board.	64
Figure 19 – Control system architecture.	65
Figure 20 – Satellite view of the path-following.	65
Figure 21 – Computational time (Experiment 2).	66
Figure 22 – Small-scale didactic microgrid schematic.	69
Figure 23 – Physical structure of the small-scale HAWT.	69
Figure 24 – Drive and measurement circuit for the wind turbine.	70
Figure 25 – Wind turbine: Identification and validation.	71
Figure 26 – Physical structure of the scaled PV system.	72
Figure 27 – Schematic of PV panel connections.	72
Figure 28 – Solar panel: Identification and validation.	73
Figure 29 – Setup of the green hydrogen branch.	74
Figure 30 – Green H_2 branch: identification and validation.	76
Figure 31 – Communication architecture.	77
Figure 32 – Physical structure of the small-scale microgrid.	78
Figure 33 – Experiment 1: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.	82

Figure 34 – Experiment 1: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.	83
Figure 35 – Experiment 2: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.	84
Figure 36 – Experiment 2: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.	85
Figure 37 – Experiment 3: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.	86
Figure 38 – Experiment 3: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.	87

LIST OF TABLES

Table 1	– State-of-the-art snippet: MPC strategies for microgrids.	34
Table 2	– MPC parameters: TRMS experiment.	54
Table 3	– Computational time statistics for the TRMS benchmark.	55
Table 4	– MPC parameters: helicopter experiment.	57
Table 5	– Simulation parameters used in both control strategies.	61
Table 6	– Performance metrics comparison between LPV-PILOT and CasADi controllers.	61
Table 7	– MPC parameters: small-scale microgrid.	81
Table 8	– Performance metrics for the LPV-NMPC controller applied to the MGCC. . .	82

LIST OF ABBREVIATIONS AND ACRONYMS

ADC	Analog-to-Digital Converter
ADMM	Alternating Direction Method of Multipliers
DC	Direct Current
DER	Distributed Energy Resource
EMS	Energy Management Systems
ESC	Electronic Speed Control
ESS	Energy Storage System
EVs	Electric Vehicles
GPS	Global Positioning System
HAWT	Horizontal Axis Wind Turbine
HIL	Hardware-in-the-Loop
I2C	Inter-Integrated Circuit
LBIP	Log-Barrier Interior-Point
LDI	Linear Differential Inclusion
LMI	Linear Matrix Inequality
LPV	Linear Parameter-Varying
LQR	Linear Quadratic Regulator
LTI	Linear Time-Invariant
MGCC	Microgrid Central Controller
MILP	Mixed-Integer Linear Programming
MIMO	Multiple-Input Multiple-Output
MINLP	Mixed-Integer Nonlinear Programming
MPC	Model Predictive Control
NLP	Nonlinear Program
NMPC	Nonlinear Model Predictive Control
NRMSE	Normalised Root Mean Square Error
PIL	Processor-in-the-loop
PV	Photovoltaic
PWM	Pulse Width Modulation
qLPV	quasi-Linear Parameter-Varying
QP	Quadratic Program
RMS	Root Mean Square
RTI	Real-Time Iteration
SOCP	Second-Order Cone Problem
SQP	Sequential Quadratic Programming
STEM	Science, Technology, Engineering, and Mathematics
ToF	Time-of-Flight

TRMS	Twin Rotor MIMO System
UART	Universal Asynchronous Receiver-Transmitter
WCET	Worst-Case Execution Time

LIST OF SYMBOLS

J	Cost function
ρ	Scheduling parameter (LPV)
x	System state
u	Control input
k	Discrete time instant
N_p	Prediction horizon
n_x	Number of states
n_u	Number of control inputs
$\mathcal{P}$	Polytopic set of scheduling parameters
n_ρ	Number of scheduling variables
$\mathcal{X}$	Admissible state set
$\mathcal{U}$	Admissible input set
J_k	Rolling performance cost
$\mathcal{S}_x(j)$	Uncertainty propagation set
Q	State weighting matrix
R	Input weighting matrix
ξ_i	Linear tightening factor
$\hat{P}_k$	Scheduling trajectory estimate
$\mathbf{X}_t$	Terminal set
$V(x)$	Terminal cost function
P	Terminal weighting matrix
Z_k	Stacked decision vector
$\mathbf{A}_{eq}$	Equality constraint matrix
$\mathbf{B}_{eq}$	Equality constraint matrix
n_c	Number of constraints in the optimization
ϕ	Barrier function
m	Number of linear inequality constraints
n_l	Number of additional specific constraints
t	Barrier trade-off parameter
Δ_x	Descent direction
ϵ	Sub-optimality criterion / tolerance
λ^2	Newton decrement
β	Line search scaling factor
L	Lower triangular matrix from LDL decomposition
D	Diagonal matrix from LDL decomposition
T_s	Sampling time
M	Mass of the cart

m	Mass of the pendulum
l	Pendulum length
g	Gravity acceleration
P_{wt}	Power generated by wind source
P_{pv}	Power generated by photovoltaic panels
P_{H_2}	Power generated by the fuel cell
SoC	Battery State of Charge
α	Battery self-discharge factor
P_{charge}	Battery charge power
$P_{\text{discharge}}$	Battery discharge power
η	Energy conversion efficiency
P_{total}	Total generated power
P_{demand}	Power demand

NOTATION

- $\mathbb{N}$ denotes the set of non-negative integers, and define $\mathbb{N}_{[a,b]} = \{i \in \mathbb{N} \mid a \leq i \leq b\}$, $\mathbb{N}_b = \mathbb{N}_{[0,b]}$ and $\mathbb{N}_b^+ = \mathbb{N}_{[1,b]}$.
- The identity matrix of dimension j is I_j .
- Define $\text{col}(v_1, \dots, v_n) = (v_1^\top, \dots, v_n^\top)^\top$ and $\text{diag}(V_1, \dots, V_n)$ denotes the block diagonal matrix with $V_1, \dots, V_n$ on its diagonal.
- $M^\top$ and $M^{-\top}$ denote, respectively, the corresponding transpose and inverse-transpose of M .
- Given $P = P^\top \succeq 0$, denoting a positive semi-definite matrix, define $\|x\|_P = \sqrt{x^\top P x}$ and $\bar{\sigma}(P)$ the maximal singular value of P .
- The predicted value of $v(k)$ at time instant $k+i$, computed based on the information available at instant k , is denoted $v(k+i|k)$; in particular, $v(k|k) = v(k)$.
- $\ominus$ is the Pontryagin set difference, defined as $\mathcal{A} \ominus \mathcal{B} = \{a : a + b \in \mathcal{A}, \forall b \in \mathcal{B}\}$.
- $\ell(x, u) := \|x\|_Q^2 + \|u\|_R^2$, being known Q and R positive-definite matrices.

CONTENTS

1	INTRODUCTION	25
1.1	THE ROLE OF MICROGRIDS IN THE ENERGY TRANSITION	25
1.2	PROBLEM STATEMENT: THE CONTROL CHALLENGE	26
1.3	MAIN HYPOTHESIS: LPV APPROACH	27
1.4	SECONDARY HYPOTHESIS: EFFICIENT EMBEDDED OPTIMIZATION	29
1.5	OBJECTIVES AND CONTRIBUTIONS	31
1.5.1	Scope of validation	31
1.5.2	Organization of this document	32
2	LITERATURE REVIEW	33
2.1	MPC AS ENERGY MANAGEMENT SYSTEMS FOR MICROGRIDS	33
2.2	REAL-TIME NMPC SOLVERS	35
2.3	LINEAR PARAMETER VARYING FRAMEWORK	36
2.4	LINEAR PARAMETER VARYING FRAMEWORK FOR MODEL PREDICTIVE CONTROL ALGORITHM (LPV-MPC)	37
2.5	OPEN INVESTIGATION THREADS	39
3	LPV-PILOT, A DEDICATED SOLVER	40
3.1	STABILIZING LPV NMPC DESIGN	40
3.1.1	Constraints tightening	41
3.1.2	Quadratic program formulation	42
3.1.3	Iterative (SQP) approach	44
3.1.4	Recursive extrapolation approach	44
3.2	LPV-PILOT SOLVER	44
3.2.1	Stacked decision vector	45
3.2.2	Logarithmic-barrier	46
3.2.3	Interior-point algorithmic structure	47
3.2.4	Proposed Log-Barrier Interior-Point (LBIP) implementation	48
3.2.5	Linear decomposition	49
3.2.6	Warm-start	50
3.2.7	Fallback control strategy	51
3.2.8	Overview of the proposed solver	51
3.3	RESULTS	52
3.3.1	Twin Rotor MIMO System (TRMS)	52
3.3.2	Automatic helicopter flight	55
3.3.3	Comparison with CasADi	58
3.3.4	Path-following in a real-world propeller driven sailboat: experimental validation	61
3.4	A SUMMARY OF FINDINGS	66

4	RENEWABLE MICROGRID APPLICATION	67
4.1	LOW-COST SMALL-SCALE RENEWABLE MICROGRID	67
4.1.1	Wind turbine	68
4.1.2	Solar panel	71
4.1.3	Green hydrogen branch	72
4.1.4	MGCC and emulated battery	75
4.1.5	Involved communication protocols	76
4.2	MICROGRID MODELLING	78
4.3	MICROGRID CENTRAL CONTROLLER (MGCC)	80
4.3.1	Experiment 1: Results	81
4.3.2	Experiment 2: Results	83
4.3.3	Experiment 3: Results	83
4.3.4	Results summary	86
5	CONCLUSION	88
5.1	SUMMARY OF THE DISSERTATION	88
5.2	CONCLUDING REMARKS	89
5.3	SCIENTIFIC CONTRIBUTIONS	90
	References	92
	APPENDIX A EXAMPLES OF QUASI-LPV EMBEDDING	101
A.1	LINEAR DIFFERENTIAL INCLUSION (LDI) PROPERTY	101
A.2	GENERIC EXAMPLE	101
A.3	GENERALIZATION OF THE LDI PROPERTY FOR COMMON NONLIN- EARITIES	102
	APPENDIX B ALGORITHMS	104
	APPENDIX C SCHEDULING PARAMETERS FOR THE TRMS MODEL	105

1 INTRODUCTION

This Chapter introduces the research topic, outlining the purpose and scope of the study regarding the role of microgrid control within the energy transition. The main contributions of this work are related to the control challenges imposed by such systems' nonlinear characteristics and hard real-time¹ constraints.

1.1 THE ROLE OF MICROGRIDS IN THE ENERGY TRANSITION

The global community is currently facing a climate emergency of unprecedented proportions, primarily driven by the excessive emission of greenhouse gases from fossil fuel combustion. This phenomenon has catalysed an urgent need for decarbonization to mitigate the severe impacts of global warming and climate change. Consequently, there is a worldwide imperative to transition from carbon-intensive energy matrices to clean, renewable energy sources. This ethical goal has been formalized within the scope of several global initiatives, such as the United Nations' 2030 Agenda for sustainable development, that establishes clear goals for the generation of renewable, affordable, and reliable energy (Olivares *et al.*, 2014; Sen; Kumar, 2018a; Morato *et al.*, 2021; Lup *et al.*, 2023). Achieving these targets requires not only the adoption of renewable generation but also a paradigm shift in how power systems are structured and operated.

In this context, the integration of intermittent renewable sources poses significant challenges to the traditional, centralized power grid. To address these complexities, Microgrids—also known as *smart grids*—arise as a fundamental technological paradigm (Hu *et al.*, 2023). These schemes represent a decentralized approach to energy generation and management, integrating multiple distributed energy resources (DERs), energy storage systems (ESS), and controllable loads into a single, manageable entity that can operate both connected to the main grid and in an isolated mode (Olivares *et al.*, 2014; Sen; Kumar, 2018a; Lasseter, 2002; Lasseter; Paigi, 2004; Sen; Kumar, 2018b).

Figure 1 presents an example of a general microgrid structure with centralized control architecture. As shown, the whole structure is centralized in a single main controller, so-called Microgrid Central Controller (MGCC), which takes decisions based on the information received by each system unit. It is a bidirectional communication, where the MGCC receives and send signals to each module in the microgrid. There are also two other structures: the decentralized, which is based on a local controller for each system unit, and the distributed control, that has a bidirectional communication between local controllers with the objective of optimizing the microgrid operation.

¹ In this work, the term 'real-time' is employed in the sense of Real-Time Optimization, where the objective is for the algorithm to converge within time windows useful for control purposes. It is important to note that this work does not focus on providing formal hard real-time guarantees, such as deterministic worst-case execution time (WCET) analysis, but rather on developing control algorithms that are shown to operate within certain fixed time constraints (in practice).

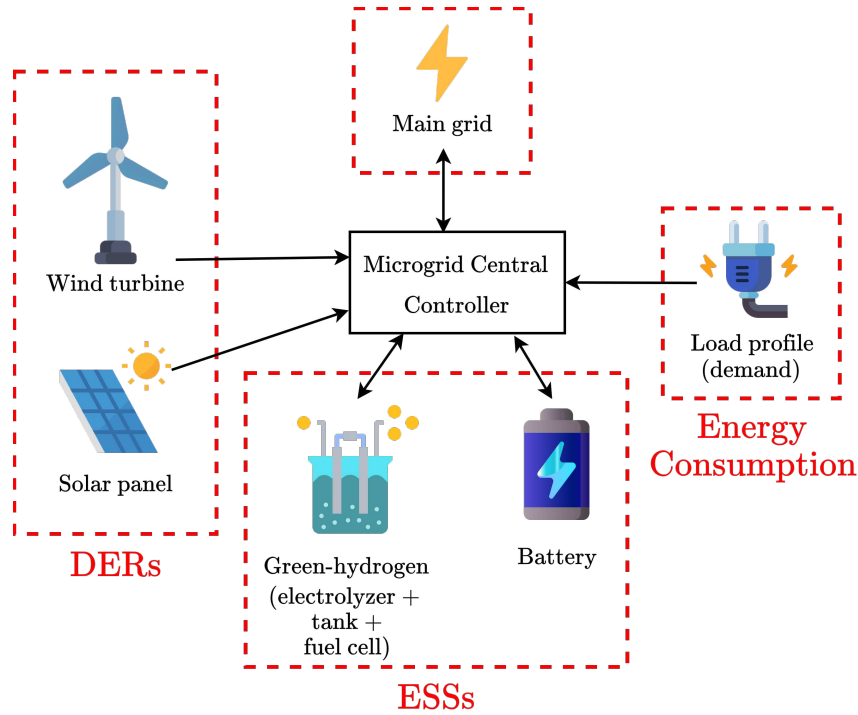


Figure 1 – Example of a generic microgrid structure.

The transition of the current energy generation setting to microgrids offers robust solutions to the challenges posed by renewable sources. By bringing generation closer to the end-users (energy consumers), these schemes are able to mitigate transmission losses, increase system resilience against faults, and enable greater penetration of intermittent sources, such as solar photovoltaic and wind. However, the efficient and safe operation of these complex systems introduces significant control challenges. The stochastic and intermittent nature of renewable generation, combined with the variability of consumer demand, requires sophisticated, real-time coordination of all microgrid assets (Velarde *et al.*, 2017; Morato *et al.*, 2020b, 2018, 2019).

Additionally, modern microgrids exhibit great complexity due to the strong nonlinear characteristics originating from associated processes such as batteries and green-hydrogen-based technologies (e.g., electrolyzers and fuel cells), as well as the power converters that interface with the grid. Such dynamics cannot be accurately described by simple linear models. The optimal management of such heterogeneous systems, therefore, requires advanced control techniques that can simultaneously handle nonlinear dynamics, operational constraints, and inherent model uncertainties (Pereira *et al.*, 2013).

1.2 PROBLEM STATEMENT: THE CONTROL CHALLENGE

To address the diverse challenges of nonlinear regulation and energy management in microgrid systems, Model Predictive Control (MPC) is well-established in the literature as a highly effective method for the tertiary control level, as thoroughly discussed in Bordons, Garcia-Torres, and Ridao (2020). The key aspect of the MPC algorithms lies in its intrinsic ability to

optimize the system's performance over a prediction horizon, taking into account the physical and operational constraints of the involved components (Camacho; Bordons, 2007).

However, model representation accurateness (i.e. description fidelity and reliability) is crucial for MPC performance. Hence, since microgrid components are nonlinear, linear time-invariant (LTI) models are typically unable to accurately describe the involved dynamics. For this reason, Nonlinear MPC (NMPC) emerges as the necessary approach for systems seeking to capture the nonlinearity of these processes and provide optimal performance (Allgower; Findeisen; Nagy, 2004). Despite their high performance, their formulation requires, at each sampling time, the solution of a Nonlinear Optimization Problem (NLP) which is generally non-convex and computationally intensive (Sanches *et al.*, 2025b).

When NMPC is applied to a broader class of systems (not only to microgrids), the computational cost of NMPC becomes particularly a significant challenge, since these systems usually operate under very strict sampling times, on the order of a few milliseconds - or even microseconds (Sanches *et al.*, 2024). Herein lies the main problem investigated in the context of this Master's dissertation, which is posed as follows: **How to enable NMPC applications for complex nonlinear systems, operating under strict sampling?**

This research problem imposes a question inherently related to the significant computational burden required by NMPC algorithms. The implementation of NMPC algorithms in low-cost hardware, with limited numerical resources, such as microcontrollers commonly used for embedded systems, is a subject of scholastic relevance (Raffo *et al.*, 2009; Carlos *et al.*, 2020).

Nowadays, solving the Nonlinear Programming (NLP) problem required by NMPC inherently imposes a high numerical burden, often hindering its application in resource-constrained embedded hardware due to feasibility and stability concerns. While powerful solvers like CasADi and acados (Andersson *et al.*, 2019; Verschueren *et al.*, 2022) can achieve real-time performance based on specific approximations, they often rely on non-trivial software stacks and third-party libraries. This complexity poses a significant barrier for industrial validation and certification processes—such as in the automotive sector—where code reproducibility, transparency, and independence from complex external software environments are mandatory.

This research gap between the theoretical relevance of NMPC and its practical limitations when operating in embedded hardware constitutes the main challenge for implementing optimal control techniques in real-world microgrids.

1.3 MAIN HYPOTHESIS: LPV APPROACH

The dedicated available literature regarding real-time optimization has consistently focused its efforts on developing solvers that efficiently approximate the original NLP solution such that a corresponding near-optima can be generated online, during the control implementation. Indeed, the state-of-the-art exhibits several tools for this task, such as the acados (Verschueren *et al.*, 2022) and GRAMPC (Englert *et al.*, 2019). These solvers are based on Real-

Time Iteration (RTI) schemes, multiple-shooting discretization and gradient-based formulations to solve a sequence of Quadratic Programs (QPs) to converge rapidly to the neighbourhood of the true nonlinear solution.

In this work, however, a distinct scholastic is explored. Instead of approximating the solution to a problem that is intrinsically difficult (the sampled NLP required to determine the NMPC policy), the main hypothesis of this work is that it is possible to transform the original problem into a simpler format that can be faster to solve, without compromising the nonlinear model's accuracy, by exploiting the structure of the original system representation. This hypothesis holds for any nonlinear system respecting the so-called linear differential inclusion (LDI) property. This property and some examples of linear inclusion using qLPV are extracted from Morato, Normey-Rico, and Sename (2020a) and are presented in Appendix A. Moreover, we consider that this can be done without compromising the model fidelity. The rationale is presented in Figure 2.

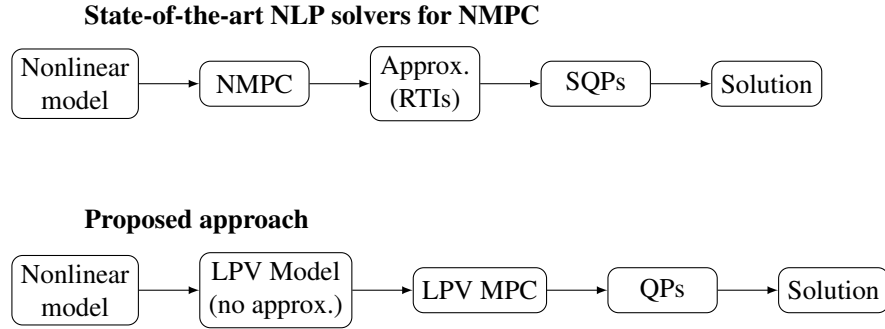


Figure 2 – Comparison between state-of-the-art NMPC and the proposed LPV MPC approach.

To this end, our proposal aims to enable NMPC algorithms to operate under hard real-time constrained environments, i.e. on embedded platforms running in the millisecond sampling range. Accordingly, the developments consider reformulation of the nonlinear prediction model by means of a Linear Parameter Varying (LPV) framework (Morato; Normey-Rico; Sename, 2020b).

The adoption of the LPV framework is motivated by its proven ability to bridge the gap between established linear control theory and the nonlinear nature of physical systems - cf. the survey (Morato; Normey-Rico; Sename, 2020b). The LPV paradigm is widely consolidated in practice, allowing for the exact representation of complex dynamics without the local approximation errors inherent to traditional linearization methods (Sanches *et al.*, 2025a; Mohammadpour; Scherer, 2012; Zanette *et al.*, 2021; Peixoto *et al.*, 2025; Souza *et al.*, 2022; Oliveira *et al.*, 2025). Furthermore, the theoretical foundations for LPV-based NMPC—including conditions for stability and recursive feasibility—are already well-documented in the literature (Morato; Holicki; Scherer, 2023; Morato *et al.*, 2024). However, despite this theoretical maturity, a significant technological aspect remains open: up to our best knowledge, there is currently no dedicated solver specifically engineered to exploit this LPV structure for embedded applications. Consequently, real-time implementations are often hindered by the lack of numerical tools capable of handling

these formulations on resource-constrained hardware.

Specifically, through a technique known as quasi-LPV (qLPV), it is possible to exactly represent the nonlinear dynamics of a system using a model with a linear structure (Shamma, 2012; Scherer, 2001; Sala, 2017). By exploiting such qLPV embeddings, the corresponding state-space matrices depend on scheduling variables, which in turn are known functions of the system's states and/or inputs.

The practical consequence of such a framework is transformative, since there is no approximation in the conversion of the nonlinear model to its LPV representation. By using this reformulation, the nonlinear problem (NLP) is transformed into a quadratic problem (QP). QPs can be solved much more efficiently than their NLP counterparts, using robust algorithms with guaranteed convergence, such as the Interior-Point method, which is discussed in the sequel. Therefore, the conversion from NLP to QP is the key to enabling NMPC applications for real-time embedded applications and constitutes the pillar of the solution proposed in this study.

1.4 SECONDARY HYPOTHESIS: EFFICIENT EMBEDDED OPTIMIZATION

The current landscape regarding optimization for control purposes is diverse, encompassing various paradigms to handle the computational demands of real-time applications. Beyond traditional online optimization, approaches such as Explicit MPC (Bemporad *et al.*, 2002) seek to pre-compute the control law offline, while emerging methods leverage Neural Networks and Machine Learning (Chen *et al.*, 2018; Hose *et al.*, 2025) to approximate the optimal control surface, significantly reducing online effort. However, these alternatives often face scalability issues with high-dimensional states or challenges in formal stability certification.

In this context, the strategies most commonly deployed for industrial and embedded Nonlinear MPC (NMPC) may be broadly categorized, from an implementation perspective, into two main groups, each presenting specific limitations for embedded applications.

At the pinnacle of complexity lie solvers designed for generic Nonlinear Programming (NLP) problems. Tools such as CasADi (Andersson *et al.*, 2019) and IPOPT (Wächter; Biegler, 2006) represent the state-of-the-art in terms of flexibility and mathematical capacity. IPOPT (Interior Point OPTimizer), for instance, employs an interior-point filter line-search method (Boyd, 2004) to ensure global convergence even in non-convex scenarios, while CasADi serves as a symbolic framework uniting various methods for numerical optimal control, including Nonlinear MPC (NMPC). Although highly capable, these solvers intrinsically carry a significant computational overhead. These algorithms typically rely on dense linear algebra libraries, dynamic memory allocation, and automatic differentiation routines (for Hessian and Jacobian computation) that are prohibitively heavy for microcontrollers with RAM in the order of kilobytes (e.g., ESP32 or STM32). Furthermore, certifying the worst-case execution time for generic NLPs is challenging, posing a major obstacle for hard real-time constraints. Beyond these numerical constraints, for industrial applications (such as the automotive sector), these algorithms face significant challenges regarding certification processes, since they rely on third-party libraries

that are complex (and often impossible) to embed.

In the other category, there is a class of solvers highly optimized for Quadratic Programming (QP), which assumes linear system dynamics and quadratic cost functions. Prominent examples include OSQP (Operator Splitting Quadratic Program) (Stellato *et al.*, 2020), qpOASES (active-set based) (Ferreau *et al.*, 2014), and TinyMPC (ADMM based) (Nguyen *et al.*, 2024). Additionally, the ODYS QP solver (Cimini; Bemporad; Bernardini, 2017) offers robust performance and certification for embedded systems. However, their direct application to nonlinear systems (such as hydrogen batteries and microgrids) is limited. These solvers generally require linearization (often via Taylor series) around a specific operating point, introducing approximation errors that can compromise performance and even closed-loop stability when the system deviates from that point. Strategies such as Real-Time Iteration (RTI) attempt to mitigate this by updating the linearization at each step; however, this approach demands extensive offline pre-computation and increases implementation complexity, requiring the user to handle model manipulation externally to the solver.

Bridging these two domains lies the Linear Parameter Varying (LPV) framework. This approach allows for the conversion of nonlinear dynamics into a pseudo-linear² format using the quasi-LPV (qLPV) embedding without loss of generality—meaning there is no model approximation error, as detailed in Scherer (2001), Sename, Gaspar, and Bokor (2013) and Sanches *et al.* (2025c). Theoretically, this strategy enables the exact³ representation of complex nonlinear behaviours while preserving the convexity of the optimization problem (formulated as a QP) for a given scheduled parameter trajectory. Despite the theoretical appeal of this method, there is currently a lack of dedicated NMPC solvers that exploit the LPV structure specifically for embedded real-time optimization.

Therefore, based on the preceding analysis, the appeal of solving complex nonlinear problems via efficient QP methods is evident. This discussion points towards the potential of utilizing qLPV embedding for such solutions. Furthermore, in line with literature suggestions (Boyd, 2004), it is postulated that the most efficient and robust numerical approach for the resulting QP in strict real-time systems is the Primal-Dual Interior-Point Method with Logarithmic Barriers. This technique transforms the constrained problem into a sequence of smooth, unconstrained problems by penalizing the approach to boundaries. This results in a predictable and stable number of Newton iterations, largely independent of the active constraint set. Moreover, for control applications, a suboptimal solution—obtained by truncating the barrier iterations with a fixed centering parameter—is often indistinguishable from the optimal one in terms of closed-loop performance, enabling massive speed gains.

Based on the previous discussion, the following hypothesis is drawn in relation to the

² It is linear with regard to the state-state transition and state-input relation, whereas (possibly) nonlinear from the parameter-state perspective.

³ The exactness of this representation for MPCs algorithms depends on the exactness of the scheduling trajectory estimation. Several methods are used for this, each one with its specific trade-off between accuracy and computational cost.

Master research problem: it is possible to achieve an optimal trade-off between the fidelity of nonlinear models and the computational simplicity of quadratic programs by combining qLPV embedding with an efficient Logarithmic Barrier Interior-Point Method to ensure fast convergence.

1.5 OBJECTIVES AND CONTRIBUTIONS

Taking into account the aforementioned research problem and the described hypothesis, this study has the goal of developing plug-and-play packages for the solution of nonlinear optimization programs for the coordination of renewable microgrid subsystems (in real-time, at the microsecond or millisecond scale), based on the Nonlinear Model Predictive Control methodology and the use of Linear Parameter Varying models.

Accordingly, this work aims to contribute to the frontiers of knowledge in the area of predictive control for energy processes, focusing on microgrids that utilize diverse energy vectors, considering load control and the use of green hydrogen. In particular, this work developed new predictive control strategies using the LPV framework, programmed into ready-to-use numerical packages (software) for real industrial environments. Such packages may enable the real-time solution of nonlinear optimization programs that ensure the proper and optimal operation of microgrids.

The main contributions of this work can be summarized as follows:

- We develop an LPV-based, plug-and-play solver for NMPC using the Interior-Point Method with logarithmic barrier.
- We validate this customized solver in several numerical and experimental examples.
- Then, we model a reduced-scale microgrid with green hydrogen, wind, and photovoltaic components.
- Finally, we experimentally validate the developed solver on the real-world microgrid testbed.

Moreover, the specific contributions are presented in the following:

- We embed the solver in low-cost microcontrollers using the C/C++ language.
- We validate the developed solver in real-time embedded systems.

These developments are centred around the main hypothesis that it is possible to convert complex and computationally costly NLP problems into equivalent QPs, which are simpler and faster to solve. To this end, this work relies on the LPV framework to embed the model's nonlinearities within a model that has a linear structure.

1.5.1 Scope of validation

A disclaimer regarding the scope of validation is necessary. Although this dissertation focus in the microgrid's application, other challenging nonlinear systems are also explored in the

development of this work, both for evaluating the accuracy and numerical cost of the proposed solution.

1.5.2 Organization of this document

This project is divided as follows. Firstly, Chapter 2 provides a literature review that seeks to reveal the research gap that this work aims to fill. Then, Chapter 3 presents `LPV-PILOT`, a dedicated solver for NMPC that was developed in this research. The corresponding results are also presented. Moreover, Chapter 4 presents the solver application into microgrids, specifically bringing an experimental validation in a low-cost small-scale microgrid. Lastly, Chapter 5 summarize the findings and draw the conclusions of the project, highlighting its contributions.

2 LITERATURE REVIEW

This Chapter establishes the theoretical foundation related to this dissertation. To this end, a critical review of the relevant literature is provided. The review is structured around four interconnected pillars:

1. The application of predictive control in microgrids.
2. The landscape of solvers for real-time optimization and embedded NMPC.
3. The theoretical framework of LPV systems.
4. Predictive control strategies specific to LPV systems.

By surveying the related works w.r.t. these topics, we seek to highlight the gap in the literature that this research aims to fill.

2.1 MPC AS ENERGY MANAGEMENT SYSTEMS FOR MICROGRIDS

Energy management in microgrids is typically organized into a hierarchical control architecture comprising primary, secondary, and tertiary levels. The primary control layer operates at the fastest timescale to guarantee voltage and frequency stability, while the secondary layer corrects any resulting steady-state deviations. The tertiary level, in turn, functions on a slower timescale, overseeing the optimal power flow and economic dispatch of the microgrid. It is precisely at this tertiary level (see Figure 3) that Model Predictive Control (MPC) has emerged as the dominant technology, owing to its inherent capability to proactively optimize system performance (Hu; Zhu; Guerrero, 2014; Bordons; Garcia-Torres; Ridao, 2020).

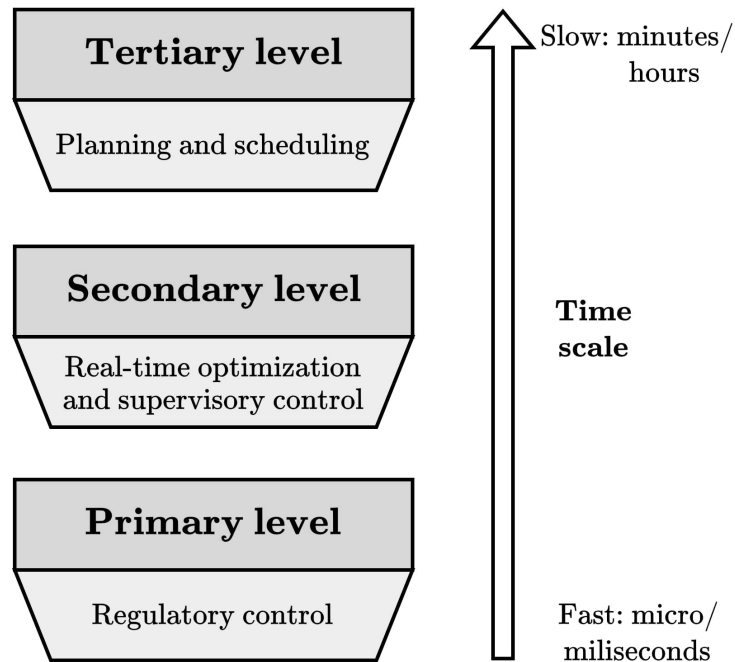


Figure 3 – Hierarchical control levels in microgrids.

In the context of microgrids, MPC formulations are frequently employed as the core optimization strategy within Energy Management Systems (EMS). These controllers are projected with the objective of minimizing a cost function J that corresponds to operational and economical objectives. These objectives typically include minimizing the cost of energy imported from the main grid, maximizing revenue from the export of surplus energy, minimizing the degradation of costly components like batteries and hydrogen systems, and maximizing the use of locally generated renewable energy. The work of Hu *et al.* (2023), for example, presents a detailed cost function for a microgrid with hydrogen storage, weighing the cost of energy exchanged with the grid, the use of the hydrogen system (which affects its lifetime), and the maintenance of desired levels of stored energy in batteries and the metal hydride tank.

A key challenge in the application of MPC for microgrids is handling the inherent uncertainties, particularly in forecasting renewable generation and load demand. The literature typically addresses this through two main paradigms: Stochastic MPC, which optimizes the expected value of the cost function across a probability distribution of uncertainties, and Robust MPC, which optimizes for the worst-case scenario (Velarde *et al.*, 2017). While the approach in this dissertation is deterministic in its online optimization, it incorporates a robustness mechanism via constraint tightening. This technique ensures that the physical system's constraints are met despite prediction errors inherent to the simplified model.

Table 1 summarizes a selection of some scholastically representative works in the context of MPC applications for microgrids, indicating their objectives, components and methodology. There are also other relevant works which were omitted for brevity, cf. (Mendes *et al.*, 2016; Morato *et al.*, 2021; Bernardi *et al.*, 2021; Hu; Zhu; Guerrero, 2014; Han *et al.*, 2018; Nassourou *et al.*, 2017).

Reference	(Parisio; Rikos; Glielmo, 2014)	(Pereira <i>et al.</i> , 2013)	(Mendes; Normey-Rico; Bordons, 2015)	(Velarde <i>et al.</i> , 2017)	(Alarcón <i>et al.</i> , 2022)
Objective	Operation	Economic management	Economic management	Power dispatch	Economic management
Components	Generation, storage and loads	PV, battery and H2 (electrolyser, fuel cell)	Grid, PV, Wind, Battery (ESS), Electric Vehicles (EVs)	PV, battery and H2 (electrolyser, fuel cell)	PV, wind, diesel generator, battery
MPC Type	Linear (MILP)	Linear (LQR) / NMPC	Linear (Economic MPC)	NMPC	Linear (MILP)
Uncertainty	Deterministic	Deterministic	Deterministic (with predictions/forecasts)	Stochastic (Multi-Scenario, chance-constrained)	Stochastic
Validation	Simulation/ Experimental	Experimental	Simulation	Experimental	Simulation

Table 1 – State-of-the-art snippet: MPC strategies for microgrids.

The works mentioned in Table 1 serve to elucidate the position of MPC as the state-of-the-art strategy for the energy management of modern microgrids. However, the necessity to

address the system's nonlinear dynamics (such as batteries, fuel cells and power converters) is evident. This dynamics usually requires an NMPC formulation.

As discussed in Chapter 1, the nonlinear formulation usually comes with a significant challenge: at each sampling instant, a computationally intensive and generally non-convex NLP problem must be solved online (Sanches *et al.*, 2025b). The involved computational burden directly conflicts with the core requirement of real-time operation on low-cost embedded hardware, which is a critical factor for the scalability and economic viability of microgrid solutions (Carlos *et al.*, 2020).

Therefore, applying advanced NMPC strategies for real-time systems involves practical feasibility concerns due to occasional sampling overrun, which can lead the system to instability and performance degradation. Fundamentally, this control problem turns into a numerical optimization challenge. Indeed, to generate an optimal control law within a strict time sampling is very challenging, and depends on the efficiency of the underlying optimization solver. As a result, evaluating the available solvers, their core methodologies, advantages and limitations is an essential task for the comprehension of the actual state-of-the-art and its gaps in embedded contexts.

2.2 REAL-TIME NMPC SOLVERS

The viability of applying NMPC schemes for embedded systems depends on softwares capable of computing real-time nonlinear optimization under strict sampling time periods. While this work focuses on efficient online optimization, the landscape of practical NMPC includes several other paradigms. For instance, Learning-based MPC leverages Neural Networks (NN) to learn the optimal control law offline (Chen *et al.*, 2018). If the problem is solved repeatedly, the initial training effort is compensated by nearly instantaneous online execution. Another approach is Explicit MPC (Bemporad *et al.*, 2002), which pre-computes the solution for all possible states, though it faces scalability issues for high-dimensional systems. Furthermore, the Practical Nonlinear Model Predictive Control (PNMPC) framework (Plucenio *et al.*, 2007) simplifies the nonlinear problem by decomposing the dynamics into free and forced responses, approximating the nonlinear behaviour with linear models to achieve computational feasibility.

Despite these diverse alternatives, many industrial applications still rely on direct online optimization due to its flexibility in handling changing constraints and model updates. Within this online optimization context, the software tools can be divided into two main categories: modelling frameworks (parsers) and solvers. The modeling frameworks, such as CasADi (Andersson *et al.*, 2019) and YALMIP (Lofberg, 2004) provide a high-level symbolic language to formulate complex optimization problems, computing automatically the necessary derivatives (gradients and Hessians). The solvers, on the other hand, are the numerical algorithm that effectively find the solution of the formulated problems. It is the case of, for example, the IPOPT for generic NPs (Curtis; Schenk; Wächter, 2010), and the qpOASES (Ferreau *et al.*, 2014) and HPIPM (Frison; Diehl, 2020) for QPs.

The state-of-the-art solvers for high-performance NMPC are typically based on Sequential Quadratic Programming (SQP) with Real-Time Iterations (RTI). Tools such as ACADO (Houska; Ferreau; Diehl, 2011) and more recently, its successor *acados* (Verschuere *et al.*, 2022), implement versions of SQPs derived from RTI strategies. In particular, the RTI method consists of performing only a single iteration of the Newton method per sampling interval. The nonlinear problem is, then, linearized around the previous solution's trajectory, resulting in a QP that is then solved to obtain the next control law. This approach exploits the temporal continuity of control problems, for which the optimal solution at one time instant is generally close to the solution at the previous instant, allowing for extremely fast convergence (Carlos *et al.*, 2020; Verschuere *et al.*, 2022; Lahr *et al.*, 2024).

However, the scenario changes when it comes to embedded platforms with limited computational resources (memory and processing) and environments subject to hard real-time constraints (by this, we mean operating within strict sampling periods - in the range of a few milliseconds). In these cases, solvers such as TinyMPC (Nguyen *et al.*, 2024), methods like the Fast MPC (Boyd, 2004) and code generation frameworks such as CVXGEN (Mattingley; Boyd, 2012) were designed to achieve fast computation on these platforms. A more detailed analysis indicates that these toolboxes are highly specialized in solving optimization problems that are already convex, such as QPs and Second-Order Cone Problems (SOCP) (Yamazaki *et al.*, 2017). TinyMPC, for example, utilizes the Alternating Direction Method of Multipliers (ADMM) and exploits the sparse structure of the MPC problem to achieve speed gains of almost an order of magnitude compared to solvers such as OSQP (Stellato *et al.*, 2020).

In this context, a critical gap can be perceived: the existing tools for embedded systems do not provide a native and integrated end-to-end solution for the NMPC problem. They are adequate for solving QPs, but they do not address the preceding step of transforming an NLP into a QP.

To this end, several recent works have addressed the Linear Parameter Varying framework seeking to build a bridge between the NLP and QP problems, uniting the accuracy of a nonlinear model, with the simplicity of its linear representation (Morato *et al.*, 2020a; Sanches *et al.*, 2025b). Such strategy will be explored in details in the following.

Accordingly, the present work intends to address this scholastic gap by developing a complete end-to-end solution. As shown along the following Chapters, this work proposes a scheme that departs from a generic baseline nonlinear model, the proposed solver formulates a QP problem via qLPV embedding and then solves this problem very efficiently on low-cost embedded systems using an interior-point algorithm.

2.3 LINEAR PARAMETER VARYING FRAMEWORK

LPV systems represent a powerful connection between linear control theory, which is well-established and has robust numerical approaches, and nonlinear systems, which describe the majority of physical systems of interest. In fact, an LPV system is formally described, in

state-space, by a model for which the state matrices are functions of a scheduling parameter vector, which is either exogenous or endogenous. These parameters are denoted by $\rho(k)$ and are considered as time-varying by definition. Hence, in discrete-time, the aforementioned state-space representation assumes the following format (Sename; Gaspar; Bokor, 2013; Scherer, 2001):

$$x(k+1) = A(\rho(k))x(k) + B(\rho(k))u(k), \quad (1)$$

where x and u represent the states and inputs, respectively, while $A(\cdot)$ and $B(\cdot)$ are parameter-dependent state matrices and $\rho \in \mathbb{R}^{n_\rho}$ represents the scheduling variables, which are bounded and known at each sampling instant k .

The main advantage of the LPV approach is that for a fixed $\rho(k)$ at a given sampling instant, the system can be viewed as if it were LTI, which allows the application of many tools for analysis and linear control synthesis. We stress that the input-state linearity in the LPV representation in (1) is preserved, even with varying parameters.

A technique of fundamental importance to this work is the use of quasi-LPV (qLPV) embedding. This method provides an exact representation of a wide variety of nonlinear systems (further details and examples are presented in Appendix A). In the qLPV approach, the scheduling variables $\rho(k)$ are defined as functions of the states $x(k)$ and/or the control input $u(k)$. By replacing the nonlinearities of the original model with these scheduling variables, a system with a linear structure is obtained, as in Equation (1), which is mathematically equivalent to the original nonlinear model within a defined operating region (Morato; Normey-Rico; Sename, 2020b). This approach differs fundamentally from traditional linearization, which is typically based on local Taylor expansions around some operational point, producing an approximation of the system that is valid only within a neighbourhood of the linearization point.

The field of LPV control is mature and well-established, with a vast literature ranging from stability and robustness analysis to controller design and practical applications (Sename; Gaspar; Bokor, 2013; Shamma, 2012). Comprehensive surveys (Hoffmann; Werner, 2014; Morato; Normey-Rico; Sename, 2020a) attest to the theoretical soundness and practical relevance of this framework, with numerous validated applications.

2.4 LINEAR PARAMETER VARYING FRAMEWORK FOR MODEL PREDICTIVE CONTROL ALGORITHM (LPV-MPC)

The combination of the Linear Parameter Varying (LPV) framework with the Model Predictive Control (MPC) methodology has resulted in a fertile research field, known as LPV MPC, which seeks to exploit the advantages of the LPV representation to simplify the nonlinear predictive control problem.

It is important to reinforce that the exactitude (i.e. accuracy) of the LPV representation depends on the scheduling variable. In the LPV-MPC scheme, it implies that accuracy of the future states prediction is directly dependent of a predicted scheduling trajectory $\hat{P}_k$, which

consists on an array of subsequent scheduling variables along the prediction horizon (defined as N_p):

$$\hat{P}_k = \text{col}(\rho(k|k), \rho(k+1|k), \dots, \rho(k+N_p-1|k)) , \quad (2)$$

where $\rho(k+i|k)$ is the i -th subsequent prediction of ρ at instant k .

Such prediction can be obtained through different methods, which have different numerical complexities and precision. The survey of Morato, Normey-Rico, and Sename (2020a) provides a comprehensive categorization of the main LPV-MPC strategies, which can be grouped as follows:

- **Robust Methods (Min-Max):** These approaches treat the unknown future evolution of the scheduling parameters $\rho(k+i|k)$ as a bounded uncertainty. The optimization problem is formulated to minimize the cost function for the worst-case parameter evolution. Although they robustly guarantee stability and constraint satisfaction, these methods are computationally very demanding, as they involve solving a min-max optimization problem that is often intractable online (Morato *et al.*, 2022a).
- **Tube-Based Methods:** To reduce online complexity, tube-based methods calculate a nominal trajectory for the system (typically assuming that the scheduling parameters remain constant) and, offline, design an uncertainty "tube" around this trajectory. The online control law is then designed to ensure that the actual system trajectory remains within this pre-computed tube. This approach shifts a large part of the computational effort to the offline design phase (Abbas *et al.*, 2018).
- **Sub-optimal Methods:** This class of methods adopts a more pragmatic approach to handling future scheduling parameters. Instead of optimizing over all possible future trajectories of ρ , they simply assume an estimate for its evolution. The most common approach adopted is the gain-scheduling (Alcalá *et al.*, 2020; Morato; Normey-Rico; Sename, 2020c), where it is assumed that the parameters will remain constant and equal to their value measured at the current instant k , i.e., $\rho(k+i|k) = \rho(k)$ for all i in the prediction horizon (Alcalá; Puig; Quevedo, 2019).

The conversion of a nonlinear model to an LPV representation transforms the robust optimization problem into a single deterministic QP at each time step, which drastically reduces computational complexity. Although the survey in Morato, Normey-Rico, and Sename (2020a) classifies this approach as 'sub-optimal'—since the model's prediction does not correspond to the true evolution of the nonlinear system—this should not be seen as a limitation, but rather as a deliberate and effective design choice. The prediction error introduced by the gain-scheduling assumption is formally addressed through a robust mechanism: constraint tightening (Morato *et al.*, 2024; Santos *et al.*, 2019). The state constraints are systematically "tightened" over the prediction horizon based on bounds on the rate of change of the scheduling parameters (Morato *et al.*, 2024). This ensures that, while the predicted trajectory approaches the tightened bounds, the actual system trajectory remains within the original limits, guaranteeing recursive feasibility

and constraint satisfaction. This combination represents an intelligent compromise between theoretical optimality and real-time computational feasibility.

Moreover, other scheduling variable estimation techniques, apart from the gain-scheduling, are the recursive extrapolation approach, based on local extrapolations (Morato; Normey-Rico; Sename, 2022) and the iterative (SQP) approach (Cisneros; Voss; Werner, 2016). These strategies are, then, discussed in further details later in Section 3.

2.5 OPEN INVESTIGATION THREADS

The literature review presented previously allows for the construction of a sequential narrative that culminates in the precise identification of the research gap that motivates the present dissertation. First, it was established that the optimal management of modern microgrids requires NMPC to handle their complex dynamics and operational constraints. Second, it was shown that implementing NMPC on embedded platforms is limited by a severe computational bottleneck, and that existing solvers for such platforms are designed for already-convex problems, not for the complete NMPC pipeline. In a third part, the LPV framework was presented as an elegant methodology to exactly transform the nonlinear model into a time-varying linear structure. Lastly, within the LPV MPC domain, some techniques for scheduling variable estimation were identified as options that offers different trade-off between control performance and computational feasibility for real-time applications.

The convergence of these points towards this work's research gap can be stated as follows: despite solid theoretical advances in LPV MPC and its immense promise for real-time applications, there is no dedicated, open-source, and optimized software package that implements this end-to-end methodology on embedded systems. Current implementations are largely theoretical or rely on combining generic, off-the-shelf tools that are not optimized for the specific structure inherent to the LPV MPC problem.

This study propose to address this research thread by taking into account the following methodological perspective. The framework and software package developed in this work can serve to connect the recent advances on LPV MPC theory and practical applications of embedded cyberphysical systems subject to severe computational constraints. In particular, as stated in Chapter 1, this work will particularly focus on the case of renewable microgrids. This work seeks to offer the academic and industrial communities a ready-to-use, validated, and high-performance tool to bring nonlinear optimal control to platforms where it is currently considered infeasible, with applications for the energy management tasks in microgrids.

In the following Chapter, we present the main result of this Master's dissertation: the new dedicated solver named "LPV-PILOT". Some numerical and experimental results are presented to validate its use in practice, highlighting its suitability for real-time nonlinear embedded systems.

3 LPV-PILOT, A DEDICATED SOLVER

Considering the general context of this work, as detailed in Chapters 1 and 2, we discuss, next, the main contribution related to the identified investigation thread regarding real-time NMPC solvers. In particular, considering the computational burden related to MPC in the nonlinear setting, our research focuses on developing a new nonlinear solver using the LPV framework to solve MPC convex optimization problem with rapid convergence rate and with portability for embedded systems in low-resources microcontrollers. Moreover, this Chapter presents the fundamental theory used for the development of this dedicated solver.

This Chapter constitutes a slightly modified version of a published article, cf. Sanches *et al.* (2025b) and some extracts based on Sanches *et al.* (2025d) and Sanches *et al.* (2025a).

3.1 STABILIZING LPV NMPC DESIGN

First, in this section, we revisit the fundamental aspects of gain-scheduled LPV MPC (Hoffmann; Werner, 2014; Morato; Normey-Rico; Sename, 2020b). To this end, let a nonlinear system represented in state-space be:

$$x(k+1) = f(x(k), u(k)), \quad (3)$$

where $x(k) \in \mathbb{R}^{n_x}$ and $u(k) \in \mathbb{R}^{n_u}$ stand, respectively, for the measured states and applied control inputs, with n_x and n_u being the number of states and control inputs, respectively. This system can be exactly mapped by a qLPV embedding, i.e. for every $x(k)$ and $u(k)$, there exist real matrices $(A(\rho(k)), B(\rho(k)))$ satisfying Eq. (1).

In particular, the qLPV embedding in Eq. (1) is imposed over an admissible region given by $x \in \mathcal{X}$ and $u \in \mathcal{U}$, for which it is valid that $\rho(k) = f_\rho(x(k), u(k)) \in \mathcal{P}$, with $\mathcal{P} \subset \mathbb{R}^{n_\rho}$ being a known convex scheduling set, and n_ρ the number of scheduling variables. Specifically, we assume that the sets $\mathcal{X}$ and $\mathcal{U}$ are compact and convex and indicate the system's (admissibility) constraints (typically related to physical characteristics). Regarding the nature of the LPV representation in (1), in relation to the original nonlinear model in (3), we consider along the following developments that the following hypotheses hold.

We assume that the real matrices $(A(\rho), B(\rho))$ are bounded and affine w.r.t. the scheduling parameters, i.e.

$$\begin{cases} A(\rho) &:= A_0 + \sum_{i=1}^{n_\rho} A_i \rho_i, \\ B(\rho) &:= B_0 + \sum_{i=1}^{n_\rho} B_i \rho_i. \end{cases}$$

Moreover, we consider that the scheduling parameters related to the LPV model in (1) are bounded and exhibit bounded rates of variations. That is, for all discrete-time samples $k > 0$, it follows that $\rho(k) = \rho(k-1) + \delta\rho(k) \in \mathcal{P}$ and $\delta\rho(k) \in \delta\mathcal{P}$, with corresponding box-type sets

defined as follows:

$$\begin{aligned}\mathcal{P} &:= \{\rho \in \mathbb{R}^{n_\rho} \mid |\rho_j| \leq \bar{\rho}_j, \forall j \in \mathbb{N}_{[1, n_\rho]}\}, \\ \delta\mathcal{P} &:= \{\delta\rho \in \mathbb{R}^{n_\rho} \mid |\delta\rho_i| \leq \bar{\delta\rho}_i, \forall i \in \mathbb{N}_{[1, n_\rho]}\}.\end{aligned}$$

Then, considering the nonlinear system in (1), the application of a **gain-scheduled LPV MPC** scheme requires a rolling performance cost $J_k = J(x(k), u(k))$, mapped along a finite prediction horizon of N_p samples, and the operational constraints ($\mathcal{X}$ and $\mathcal{U}$). Given this, the MPC will determine at each sample k , the optimal control input U_k^* for the prediction horizon. That is, $U_k^* = \text{col}(u^*(k|k), \dots, u^*(k + N_p - 1|k))$, where the first entry $u^*(k|k)$ is applied as the sampled feedback predictive control law.

Given the sampled state measured $x(k)$ and the corresponding scheduling variable $\rho(k) = f_\rho(x(k), u(k))$, the optimization problem is formalized as follows:

$$\begin{aligned}U_k^* &= \arg \min_{U_k} J_k, \\ \text{s.t.: } &x(k + j + 1|k) = A(\rho(k))x(k + j|k), \\ &\quad + B(\rho(k))u(k + j|k), \quad \forall j \in \mathbb{N}_{N_p-1}, \\ &x(k + j|k) \in (\mathcal{X} \ominus \mathcal{S}_x(j)), \quad \forall j \in \mathbb{N}_{[1, N_p]}, \\ &u(k + j|k) \in \mathcal{U}, \quad \forall j \in \mathbb{N}_{[0, N_p-1]}, \\ &x(k + N_p|k) \in \mathbf{X}_t,\end{aligned}\tag{4}$$

where $\mathcal{S}_x(j)$ is an uncertainty propagation set and, consequently, $\mathcal{X} \ominus \mathcal{S}_x(j)$ is a tightened constraint set, defined in the sequel (Section 3.1.1). The constraints in (4) account for the predicted system dynamics along the future horizon. Due to this, it assumes that the scheduling trajectory ρ remains constant along the horizon. Furthermore, these constraints contains also hard state and control input constraints, which are defined by the convex sets $\mathcal{X}$ and $\mathcal{U}$, which define system admissibility and the validity of the qLPV representation. Such constraints often represents the physical limits of the process (e.g., actuators and output saturation effects).

However, due to the method for trajectory estimate, an error is inherently present in the prediction considered within (4), when compared to the real state trajectories that arise when the control sequence is applied. Accordingly, to account for this issue, a constraints tightening mechanism is included in the solver, as described in the following.

3.1.1 Constraints tightening

We consider that the cost function is given by

$$J_k = \underbrace{V(x(k + N_p|k))}_{\|x(k + N_p|k)\|_P^2} + \sum_{j=0}^{N_p-1} \ell(x(k + j|k), u(k + j|k)),$$

where $\ell(x, u) := \|x\|_Q^2 + \|u\|_R^2$, with known Q , R and P positive-definite matrices. Particularly, Q and R define the trade-off between state stabilization penalization and required control effort.

Moreover, the admissibility constraints of the nonlinear system are given by box-type sets, which are defined by

$$\begin{aligned}\mathcal{X} &= \{x \in \mathbb{R}^{n_x} \mid |x_i| \leq \bar{x}_i, \forall i \in \mathbb{N}_{[1, n_x]}\}, \\ \mathcal{U} &= \{u \in \mathbb{R}^{n_u} \mid |u_l| \leq \bar{u}_l, \forall l \in \mathbb{N}_{[1, n_u]}\},\end{aligned}$$

with $\bar{x}_i$ and $\bar{u}_l$ being known state and input bounds, respectively.

Furthermore, we consider the state constraints uncertainty propagation set $\mathcal{S}_x(j)$ as compact and convex, such that $x(k+j|k) \in (\mathcal{X} \ominus \mathcal{S}_x(j))$ implies in $x(k+j) \in \mathcal{X}, \forall j \in \mathbb{N}_{[1, N_p]}$, i.e. constraining the predicted states to the tightened sets ensures that the true states are admissible. As the computation of uncertainty propagation sets is well-established in the literature (cf. Fiacchini, Alamo, and Camacho (2010), Rakovic *et al.* (2012) and Rego *et al.* (2020)), this work does not focus on it. Instead, along the following developments, we assume these sets to be simple hyper-boxes that are linearly propagated along the prediction horizon (conic tubes).

The uncertainty terms $\omega(k+j|k)$ typically grow in magnitude along the prediction horizon, i.e. w.r.t. $j \in \mathbb{N}_{[1, N_p]}$. Moreover, since the first state prediction is initialized with the current measurement ($x(k+1|k)$ is based on $x(k)$), the initial uncertainty term $\omega(k+1|k)$ is assumed to be null. In order to bound these variables, the literature shows that appropriate uncertainty propagation sets $\mathcal{S}_x(j)$ can be computed as zonotopes (Morato *et al.*, 2024; Santos *et al.*, 2019) or compact polyhedra. However, for simplicity of the developed solver, we replace them by some interior hyper-boxes approximation such that $\mathcal{X} \ominus \mathcal{S}_x(j)$ becomes a linearly-scaled contractive tube. That is, we simply take $\mathcal{S}_x(j) := \{s \in \mathbb{R}^{n_x} \mid |s_i| \leq \bar{s}_i(j), \forall i \in \mathbb{N}_{[1, n_x]}\}$ such that the following tightening relation holds for all $i \in \mathbb{N}_{[1, n_x]}$:

$$\begin{pmatrix} \bar{s}_i(1) \\ \bar{s}_i(2) \\ \bar{s}_i(3) \\ \vdots \\ \bar{s}_i(N_p) \end{pmatrix} := \begin{pmatrix} 0 \\ \left(\frac{1}{(N_p-1)}\xi_i\right)\bar{x}_i \\ \left(\frac{2}{(N_p-1)}\xi_i\right)\bar{x}_i \\ \vdots \\ \xi_i\bar{x}_i \end{pmatrix}, \quad (5)$$

with $\xi_i \in (0, 1), \forall i \in \mathbb{N}_{[1, n_x]}$ being a linear tightening factor related to each state variable.

3.1.2 Quadratic program formulation

With the exposed in previous section, the optimization problem in (4) takes the form of a Quadratic Program (QP), which is convenient due to its convexity. This kind of problem is solvable online using well-known convex optimization techniques. Moreover, we care to highlight that the prediction model constraint in (4) can be re-structured in the linear format of

$$\hat{X}_k = \mathbf{A}(\hat{P}_k)x(k) + \mathbf{B}(\hat{P}_k)U_k, \quad (6)$$

$$\mathbf{A}(\hat{P}_k) := \begin{bmatrix} A_{k|k} \\ A_{k+1|k} A_{k|k} \\ \vdots \\ \prod_{j=0}^{N_p-1} A_{k+j|k} \end{bmatrix} \quad (8)$$

$$\mathbf{B}(\hat{P}_k) := \begin{bmatrix} B_{k|k} & 0 & \cdots & 0 \\ A_{k+1|k} B_{k|k} & B_{k+1|k} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \left(\prod_{j=1}^{N_p-1} A_{k+j|k} \right) B_{k|k} & \cdots & \cdots & B_{k+N_p-1|k} \end{bmatrix} \quad (9)$$

being $\hat{X}_k := \text{col}(x(k+1|k), \dots, x(k+N_p|k))$ the predicted states, $\hat{P}_k = \text{col}(\rho(k), \rho(k+1|k), \dots, \rho(k+N_p-1|k))$ the scheduling trajectory estimate, U_k the input control sequence, and $\mathbf{A}(\hat{P}_k)$ and $\mathbf{B}(\hat{P}_k)$ the parameter-dependent matrices given in (8) and (9), respectively, where for brevity, we define the shorthand notation $A_{k+j|k} := A(\rho(k+j|k))$, and $B_{k+j|k} := B(\rho(k+j|k))$. In this work, we consider a gain-scheduled approach, then, the scheduling trajectory estimate at each sample k is set using a simplification of Eq. (2), presented as follows

$$\hat{P}_k = \text{col}(\rho(k), \rho(k), \dots, \rho(k)). \quad (7)$$

In practice, implementing the gain-scheduled LPV MPC in (4) is quite simple. The matrices presented in the sequel (in (8) and (9)) can be computed offline in an affine format, in order to efficiently reduce online computational time, as discussed in depth in Sec. 3.2.8. Then, the online procedure consists, at each discrete-time sample k , in:

1. Measuring the current state $x(k)$ and retrieving the associated scheduling variable $\rho(k) = f_\rho(x(k), u(k))$.
2. Determining the associated prediction matrices $\mathbf{A}(\hat{P}_k)$ and $\mathbf{B}(\hat{P}_k)$,
3. Solving the corresponding gain-scheduled optimization (4), thus obtaining U_k^* .

However, for a consistent operation, the so-called **theoretical certificates** are required. For that, the optimization program in (4) must be **recursively feasible** (i.e. if it is feasible for the initial conditions, at time instant $k=0$, it remains feasible for all subsequent samples $k \in \mathbb{N}$). Moreover, the resulting closed-loop dynamics, obtained by applying the sampled optimal control law $u^*(k|k)$ to the nonlinear process in (3), must be stable and the admissibility constraints must always be respected.

The specific literature shows that these properties are ensured by adding an appropriate pair of terminal cost $V(x(k+N_p|k))$ and set $\mathbf{X}_t \subset (\mathcal{X} \ominus \mathcal{S}_x(N_p))$, cf. (Morato; Holicki; Scherer, 2023; Morato *et al.*, 2024), to the optimization problem. Furthermore, we assume that these elements are both expressed quadratically in terms of the same positive-definite matrix, that is: $V(x) = x^\top P x$ and $\mathbf{X}_t := \{x \in \mathbb{R}^{n_x} \mid x^\top P x \leq 1\}$. Indeed, these **terminal ingredients** can be computed offline using, e.g., Linear Matrix Inequalities (LMIs) to generate the positive definite matrix argument P .

Although this dissertation has focused on the gain scheduling procedure for states prediction, other relevant strategies exist and are briefly presented in the sequel.

3.1.3 Iterative (SQP) approach

The **Iterative (SQP) approach**, which is also called “iterative qLPV method”, was first proposed in Cisneros, Voss, and Werner (2016) and relies on iteratively solving the QP problem multiple times per sample. Each iteration is used to refine the scheduling trajectory estimate (as eventually convergence is reached). The procedure is as follows:

- First, compute an initial scheduling trajectory $\hat{P}_k^0$ using the so-called *frozen* gain scheduling and solve the QP.
- Next, use the optimal solution U_k^* to compute the initial state prediction X_k^0 and compute a new and improved scheduling trajectory using $\hat{P}_k^1 = f_\rho(X_k^0, U_k^0)$.
- Repeat the previous steps until convergence is established, i.e. $\|\hat{P}_k^l - \hat{P}_k^{l-1}\| < \epsilon_{\text{cis}}$.

Typically, as debated in Morato, Normey-Rico, and Sename (2020b), this approach achieves very high precision. However, given that many QPs are solved per sample, it has the largest computational cost (among the presented approaches), which may be an issue for embedded real-time systems with strict sampling time.

3.1.4 Recursive extrapolation approach

The **recursive extrapolation approach**, proposed in Morato, Normey-Rico, and Sename (2022), is based on a Taylor’s approximation of the scheduling function $f_\rho(\cdot)$. In practice, it generates precise scheduling trajectory estimates with very low computational requirements. In short, at each sample, a local first-order derivative is computed as: $f_\rho^\partial(k) := \left. \frac{\partial f_\rho(x)}{\partial x} \right|_{x(k)}$. Then, the scheduling trajectories are computed based on scheduling trajectories from the previous sample, that is, $\forall j \in \mathbb{N}_{[1, N_p-1]}$

$$\rho(k+j | k) = \rho(k+j-1 | k-1) + f_\rho^\partial(k) \delta x(k+j-1 | k).$$

3.2 LPV-PILOT SOLVER

In this section, we present the dedicated LPV-Based solver for NMPC, so-called LPV-PILOT, which is available online ¹. It is fundamentally based on the classical Logarithmic-Barrier Interior-Point method for constrained optimization (Boyd, 2004). It solves (4) within a determined convex set, considering an appropriate change of variables. This section is organised as follows:

- (i) First, we discuss the change of optimization decision variables to reduce computational burden and the corresponding required offline preparations.

¹ github.com/viniciusmorenosanches/LPV-PILOT

- (ii) Then, we detail the logarithmic-barrier approach.
- (iii) Accordingly, the linear problem solution using $\text{LDL}^\top$ decomposition is detailed.
- (iv) Finally, we describe the included warm-start and fallback control strategies.

3.2.1 Stacked decision vector

The solution of the optimization problem in (1) requires, inherently, to evaluate the so-called “model equality constraint” at each sampling instant. This constraint, as stated in the prequel, can be formulated in vector form as gives Eq. (6). However, evaluating the full-state prediction matrices $\mathbf{A}(\hat{P}_k)$ and $\mathbf{B}(\hat{P}_k)$ during the implementation can be excessively demanding in numerical terms. Notice that to evaluate Eqs. (8) and (9), requires to perform several matrix multiplications.

Then, for reducing the computational cost associated to (4), we rely on Wang and Boyd (2009) to effectuate a suitable reformulation of variables in (4), resulting in the following:

$$\begin{aligned}
 Z_k^* &= \arg \min_{Z_k} J_k, \\
 \text{s.t.: } &\mathbf{A}_{\text{eq}}(\hat{P}_k)Z_k + \mathbf{B}_{\text{eq}}(\hat{P}_k)x(k) = 0, \\
 &x(k+j|k) \in (\mathcal{X} \ominus \mathcal{S}_x(j)), \forall j \in \mathbb{N}_{[1, N_p]}, \\
 &u(k+j|k) \in \mathcal{U}, \forall j \in \mathbb{N}_{[0, N_p-1]}, \\
 &x(k+N_p|k) \in \mathbf{X}_t.
 \end{aligned} \tag{10}$$

The main difference of this new formulation is the decision variable, which encompasses both the future control inputs and the state predictions: $Z_k := \text{col}(u(k|k), x(k+1|k), u(k+1|k), x(k+2|k), \dots, u(k+N_p-1|k), x(k+N_p|k))$.

Accordingly, the model equality constraint in (10) no longer requires matrix multiplication. In particular, the equality matrices $\mathbf{A}_{\text{eq}}(\hat{P}_k)$ and $\mathbf{B}_{\text{eq}}(\hat{P}_k)$ are given in Eq. (12) and (13). Notice, moreover, that, thanks to the affine nature of the original LPV model matrices and the considered gain-scheduled argument $\hat{P}_k := \text{col}(\rho(k), \dots, \rho(k))$, it is implied that the following affine inclusion holds:

$$\begin{cases} \mathbf{A}_{\text{eq}}(\hat{P}_k) &:= \mathbf{A}_{\text{eq}_0} + \sum_{i=1}^{n_\rho} \mathbf{A}_{\text{eq}_i} \rho_i(k), \\ \mathbf{B}_{\text{eq}}(\hat{P}_k) &:= \mathbf{B}_{\text{eq}_0} + \sum_{i=1}^{n_\rho} \mathbf{B}_{\text{eq}_i} \rho_i(k). \end{cases} \tag{11}$$

Then, by exploiting the structure of the modified optimization problem in (10), our solver only requires to compute, during the implementation, the linear maps in Eq. (11) to set up the equality constraints. Then, the matrices $\mathbf{A}_{\text{eq}_0}, \mathbf{A}_{\text{eq}_1}, \dots, \mathbf{A}_{\text{eq}_{n_\rho}}, \mathbf{B}_{\text{eq}_0}, \mathbf{B}_{\text{eq}_1}, \dots, \mathbf{B}_{\text{eq}_{n_\rho}}$ are constant and can be computed offline, during a preparation phase of the algorithm, before the actual solver runs online.

$$\mathbf{A}_{\text{eq}}(\hat{P}_k) := \begin{bmatrix} B_{k|k} & -I_{n_x} & 0 & \cdots & 0 \\ 0 & A_{k+1|k} & B_{k+1|k} & -I_{n_x} & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & A_{k+N_p-1|k} & B_{k+N_p-1|k} & -I_{n_x} \end{bmatrix} \quad (12)$$

$$\mathbf{B}_{\text{eq}}(\hat{P}_k) := \begin{bmatrix} A_{k|k} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (13)$$

3.2.2 Logarithmic-barrier

Looking for the solution of the optimization problem in (10), it is necessary to reformulate the hard constraints to an inequality constraint $G^\top Z_k \leq g$, with $G \in \mathbb{R}^{n_c \times N_p}$ being a linear constraint matrix and $g \in \mathbb{R}^{n_c}$ a upper-bound vector, considering n_c as the number of constraints in the optimization. We emphasize that the terminal constraint is typically quadratic, but here it is generically replaced by a nonlinear operator $M(\cdot)$. Then, with this reformulation, we obtain:

$$\begin{aligned} Z_k^* &= \arg \min_{Z_k} J_k, \\ \text{s.t.: } & \mathbf{A}_{\text{eq}}(\hat{P}_k)Z_k + \mathbf{B}_{\text{eq}}(\hat{P}_k)x(k) = 0, \\ & G^\top Z_k \leq g, \\ & M(Z_k) \leq m. \end{aligned} \quad (14)$$

As mentioned in the previous section, we applied a constraint tightening mechanism, by considering the constraints $x(k+j|k) \in \mathcal{X} \ominus \mathcal{S}_x(j)$, $\forall j \in \mathbb{N}_{[1, N_p]}$. This is done to ensure states admissibility face to uncertainties propagate along the predictions (due to the gain-scheduling argument modelling errors).

Each state sequence $\text{col}(x_i(k+1|k), x_i(k+2|k), \dots, x_i(k+N_p|k))$, $\forall i \in \mathbb{N}_{[1, n_x]}$, is defined element-wise as smaller (in magnitude) than $\text{col}(\bar{x}_i, \bar{x}_i - \bar{s}_i(2), \dots, \bar{x}_i - \bar{s}_i(N_p))$. Since the decision variable Z_k accounts for both inputs and states, the optimization problem (14), by means of the inequality constraint $G^\top Z_k \leq g$, implicitly ensures that $u(k+j|k) \in \mathcal{U}$, $\forall j \in \mathbb{N}_{[1, n_u]}$ and $x(k+j|k) \in \mathcal{X} \ominus \mathcal{S}_x(j)$, $\forall j \in \mathbb{N}_{[1, n_x]}$.

Moreover, to handle these inequality constraints, we rely in the logarithmic-barrier strategy (Boyd, 2004). This strategy moves the constraints to the cost function, producing an unconstrained problem which penalizes constraint violation and, consequently, guarantee admissibility. The advantage of having an unconstrained problem lies in the usage of well-known standard convex optimization techniques. The optimal decision variable Z_k^* is constrained. In particular,

we consider a barrier term $\phi(Z_k)$ given as follows:

$$\phi(Z_k) = -\frac{1}{t} \left[\sum_{i=1}^m \log(g_i - G_i^\top Z_k) + \sum_{l=1}^{n_l} \log(m_l - M_l(Z_k)) \right], \quad (15)$$

where the sub-indices i, l indicate the i/l -th row of the corresponding inequality constraint matrices. The term m denotes the total number of linear inequality constraints derived from the polytopic or half-space representation of the feasible set, while n_l represents the number of additional specific constraints. Moreover, $t > 0$ represents an internal barrier trade-off parameter set to determine the aggressiveness of the barrier function w.r.t. the constraints.

In practical terms, the considered barrier function $\phi(Z_k)$ grows indefinitely as Z_k approaches the boundaries of the admissibility region.

Then, the adapted cost function is given by $J_{\Phi_k} := tJ_k + \phi(Z_k)$, where the (prediction) equality $\mathbf{A}_{\text{eq}}(\hat{P}_k)Z_k + \mathbf{B}_{\text{eq}}(\hat{P}_k)x(k) = 0$ is analytically introduced to J_k . We thus proceed by solving the following unconstrained optimization problem, which is equivalent to (14):

$$Z_k^* = \arg \min_{Z_k} \left[\underbrace{tJ_k + \phi(Z_k)}_{J_{\Phi_k}} \right]. \quad (16)$$

Henceforth, the resulting optimization problem in (16) is solved iteratively by our software using Newton's method principle of convergence, as discussed along the sequel.

Since (16) is unconstrained and convex, the convergence can be achieved by steps in the descent direction, denoted Δ_x . To this end, we compute Δ_x from the solution of the typical linear problem $\nabla_{J_{\Phi_k}}^2 \Delta_x = -\nabla_{J_{\Phi_k}}$. Particularly, the proposed solver uses an $\text{LDL}^\top$ decomposition to solve the linear problem, as developed in Section 3.2.5.

The convergence of (16) is guaranteed from Newton's method, since Q , R and P are positive-definite matrices (and, thus, J_k is a convex function) and $\mathcal{X}$ and $\mathcal{U}$ are convex sets. We stress that the minimisation of the barrier term $\phi(Z_k)$ leads to a constrained solution as the hyper-parameter t grows.

3.2.3 Interior-point algorithmic structure

Once the reformulated (unconstrained) optimization problem is set in (16), we can rely in interior-point algorithms (Roos; Terlaky; Vial, 2005; Bomze *et al.*, 2010) to obtain the minimiser Z_k^* . Traditionally, these algorithms are hierarchically organised into two nested loops, which we refer as **inner** and **outer loop**:

- **Inner loop:**

- First, one computes the optimization search direction by evaluating the gradients $\nabla_{J_{\Phi_k}}$ and Hessians $\nabla_{J_{\Phi_k}}^2$ of the barrier-augmented cost function (16).

- Then, one solves the related unconstrained optimization problem (16) for a given t , which is an intermediary problem coordinated and provided by the outer loop.

- **Outer loop:**

- Here, the trade-off parameter t is adjusted, enabling the solution to approach the constraints boundaries (without crossing them).

When the trade-off parameter t increases w.r.t the problem dimension $m = (n_x + n_u)N_p$ (i.e., $m/t \leq \epsilon$), the solution provided by the inner loop is very close to the solution of the modified optimization in (14), given that $\lim_{t \rightarrow +\infty} \phi(\cdot) \rightarrow 0$.

3.2.4 Proposed Log-Barrier Interior-Point (LBIP) implementation

This section describes the the proposed logarithmic-barrier interior-point algorithm, that is the core of our solver. Moreover, we highlight some particularities that differentiates our algorithm from typical implementations.

Firstly, as suggested in Wang and Boyd (2009), we use a fixed barrier parameter t . Specifically, the referenced paper demonstrates that by fixing the trade-off parameter t as a large value in fact decreases the Newton steps necessary for convergence of the inner loop, without significantly compromising the solution accuracy. Such parameter can be adapted depending on each application. In practice, this strategy leads to sufficiently accurate solutions for most control engineering problems. This is because the primary goal is not to find the exact mathematical optimum of the cost function, which is itself an abstraction of the true control objectives, but rather a solution that is sufficiently close and computable within real-time constraints. This simplification significantly reduces the solver's computational complexity, allowing it to converge in just a few inner-loop iterations within a single outer-loop execution.

In order to highlight the particularities of the proposed solver, the complete optimization solution using the LBIP approach is presented in Algorithm 1.

The required inputs of this algorithm are the initial candidate Z_0 , which helps to warm-start our solver and accelerates the convergence, the sampled state measurements $x(k|k)$, the optimal solution from the last sample U_{k-1}^* . Also, our implementation accounts for a given state reference target x_r and a load disturbance value $d(k)$ available, in order to address tracking and feed-forward compensation objectives. The algorithm also consider an object `mpc`, which is used to update the prediction matrices $A_{eq}(\hat{P}_k)$ and $B_{eq}(\hat{P}_k)$, at each sampling instant.

The algorithm starts by enabling a Newton loop logic indicator (`continue_newton`). Then, a candidate initial solution z is picked according to the argument received in Z_w . However, if z violates constraints, a new computation of z is required to ensure appropriate feasibility.

Then, to define the Newton steps, the gradient $\nabla_{J_{\phi_k}}$ and the Hessian $\nabla_{J_{\phi_k}}^2$ are computed. These Newton's loops persist until the sub-optimality criterion $0.5\lambda^2 > \epsilon$ is satisfied, which indicates convergence. For this, we define the Newton decrement λ^2 as $-\nabla_{J_{\phi_k}}^\top \Delta x$, which is

Algorithm 1 LBIP implementation to solve (16)

```

1: Input: Initial candidate (warm-start)  $Z_w$ . Previous state  $x(k|k)$ . Previous optimal solution  $U_{k-1}^*$ .
   State reference  $x_r$ . Measured disturbance  $d(k)$ . MPC Object mpc.
2: continue_newton  $\leftarrow$  true
3:  $z \leftarrow Z_w$ 
4: if  $z$  is not feasible then
5:    $\perp$  Compute  $z$  feasible
6:    $\lambda^2 \leftarrow 1$ 
7: while continue_newton & ( $\epsilon \leq 0.5\lambda^2$ ) & (iter  $\leq$  max_iter) do
8:   Compute  $\nabla_{J_{\phi_k}}$  and  $\nabla_{J_{\phi_k}}^2$  analytically
9:   Compute  $\Delta_x$  from solving  $\nabla_{J_{\phi_k}}^2 \Delta_x = -\nabla_{J_{\phi_k}}$ 
10:   $\lambda^2 \leftarrow -\nabla_{J_{\phi_k}}^\top \Delta_x$ 
11:   $l \leftarrow 1$ 
12:   $z \leftarrow z + l\Delta_x$ 
13:  while  $z$  is not feasible do  $\triangleright$  Perform feasibility search
14:     $l = l\beta$ 
15:     $z \leftarrow z + l\Delta_x$ 
16:  iter  $\leftarrow$  iter + 1
17: if iter > max_iter then
18:    $z \leftarrow \text{FALLBACK\_CONTROL}(Z_0, z, x(k-1|k), x_r, \text{mpc})$ 
19: Extract  $u$  from  $z$ 
20: Output: Optimal control  $U_k^* \leftarrow u$ 

```

updated every Newton step iteration. Then, we compute the corresponding descent direction (obtained from the solution of $\nabla_{J_{\phi_k}}^2 \Delta_x = -\nabla_{J_{\phi_k}}$ using the $\text{LDL}^\top$ decomposition) Δ_x , which updates the iterated optimal decision variable as $z = z + l\Delta_x$. Furthermore, to guarantee feasibility, we perform a linear backtracking search to correct the step size l , using the scaling factor $\beta \in [0,1]$ as parameter.

Finally, when the sub-optimality criteria is established (and, thus, $0.5\lambda^2 > \epsilon$), we extract the optimal decision variable u from the optimal vector z and the function returns the optimal control U_k^* .

3.2.5 Linear decomposition

As mentioned previously, the linear problem to find the descent direction Δ_x is solved via $\text{LDL}^\top$ decomposition, which is well-known and established in the literature, cf. (Stanimirović, 2012; Stanimirović; Tasić, 2012). In the following, we detail our particular implementation, as given in Algorithm 2.

In short, the $\text{LDL}^\top$ decomposition is a technique to decompose a given symmetric matrix A in three parts: a triangular inferior matrix L , a diagonal matrix D and the transpose of L . Such decomposition must always satisfy the condition $A = LDL^\top$. This process is referred to as an $\text{LDL}^\top$ factorisation in Algorithm 2. Particularly, we do not compute D completely, but just a vector of the diagonal of D , considering that all the other elements are null and can thus be neglected.

Algorithm 2 $\text{LDL}^\top$ decomposition (Solve $\nabla_{J_{\phi_k}}^2 \Delta x = -\nabla_{J_{\phi_k}}$)

- 1: **Input:** Hessian of (16) $\nabla_{J_{\phi_k}}^2$. Gradient of (16) $-\nabla_{J_{\phi_k}}$.
 - 2: $A \leftarrow \nabla_{J_{\phi_k}}^2$
 - 3: $b \leftarrow -\nabla_{J_{\phi_k}}$
 - 4: $[L, D] \leftarrow \text{LDLFACTORIZATION}(A)$ $\triangleright$ Algorithm 4 in Appendix B
 - 5: $y \leftarrow \text{FORWARDSUBSTITUTION}(L, b)$ $\triangleright$ Algorithm 5 in Appendix B
 - 6: $z \leftarrow \text{DIAGONALSOLVE}(D, y)$ $\triangleright$ Algorithm 6 in Appendix B
 - 7: $x \leftarrow \text{BACKSUBSTITUTION}(L^\top, z)$ $\triangleright$ Algorithm 7 in Appendix B
 - 8: **Output:** Descent direction $\Delta_x \leftarrow x$
-

Since L is triangular inferior with a unit diagonal, we can solve $Ly = b$ using forward-substitution. However, different from the usual procedure, we do not divide y_i by $L(i, i)$, because it is imposed that $L(i, i) = 1$ for any i .

Lastly, we solve $Dz = y$ with the vector of D using an element-wise division (diagonal solution). Finally, the output x is obtained from $L^\top x = z$, using a back-substitution procedure². In sum, our $\text{LDL}^\top$ decomposition is simple and yet very effective, that is, it is computationally cheap for larger dimension problems involving matrices.

3.2.6 Warm-start

Another implemented strategy that is used to accelerate the convergence rate of the developed solver is the warm-start, widely spread in the majority of MPC-oriented optimization solvers, cf. (Wang; Boyd, 2009; Camacho; Bordons, 2013; Peccin *et al.*, 2021). This is, at each sampling instant, we provide to the optimization solver a *good* initial candidate solution, denoted Z_0 .

In practice, during the closed-loop implementation, this **warm-start** candidate solution is a shifted copy of the solution from the last sampled. That is, the optimal sequence from the previous sample $Z_{k-1}^* = \text{col}(u^*(k-1|k-1), x^*(k|k-1), \dots)$ is shifted to the left, the first two entries are suppressed, and the last two elements are duplicated, which yields³: $Z_0 = \text{col}(u^*(k|k-1), x^*(k+1|k-1), \dots, u^*(k-N_p-2|k-1), x^*(k+N_p-1|k-1), u^*(k-N_p-2|k-1), x^*(k+N_p-1|k-1))$.

In general, this strategy is effective under the assumption that under perfect prediction, the optimal sequence does not change significantly between consecutive time steps, specially when the last two elements of the decision vector account for the inputs and states at the terminal instant (meaning the system steady-state). Accordingly, for the next sample, all elements of this decision vector — except the first and last — remain unchanged and only need to be shifted.

Such warm-start technique enables the solver to converge very rapidly, focusing primarily in making small adjustments to compensate for prediction errors, rather than solving the whole

² Again, the related function also does not need the division of x_i by $U(i, i)$ because $U = L^\top$ and it also has unit diagonal.

³ For instance, if the optimal sequence from a previous sample was given by $Z_{k-1}^* = \text{col}(2, 3, 4, 5, 6)$, the chosen warm-start sequence becomes $Z_0 = \text{col}(3, 4, 5, 6, 6)$.

optimization problem from scratch.

3.2.7 Fallback control strategy

The last strategy that is used for the adaptability of our solver to strict real-time⁴ hard constraints lies in the fallback control. Indeed, a fundamental aspect for MPC embedded applications is the computational time, since the optimal solution of (16) must be computed under the sampling time period. Otherwise, this might lead to degrade performance or even instability.

Many practical systems are subject to fast dynamics, such as autonomous vehicles, electrical systems, robotics and other critical applications. In these cases, the sampling intervals T_s are often considerably small - in the range of a few milliseconds, at most (Sanches *et al.*, 2024).

Then, to avoid any potential sampling overrun, i.e. when the computational time $t_c(k)$ is greater than the sampling interval T_s , we added a fallback control strategy. In practice, if at a given sampling instant, the computational time to execute Algorithm 1 exceeds a threshold (determined by the user) of the sampling time interval, the Newton steps are interrupted and the fallback controller is then triggered to compute an alternative law.

This threshold is, actually, set in the solver as a hyper-parameter `max_iter` (eight line of Algorithm 1), which determines the maximum number of iterations tolerated to run the Newton loops. Assuming that the time per iterations is constant (in average), the maximum number of iterations indirectly limit the maximum computational time for executing Algorithm 1.

In this way, when the computational time constraint is violated, i.e. the number of iterations exceeds `max_iter`, the fallback algorithm is called. This algorithm executes a lightweight routine that selects the “best” control action amongst two options: the shifted optimal command from the previous sample Z_0 (i.e. the warm-start decision at instant k), or the *incomplete* sampled command z .

The decision of which control action is best to be applied as a fallback law is taken by considering the associated MPC cost J_k . In practice, we determine which of the two possible fallback laws implies in the smallest associated performance cost for the one-step-ahead system behaviour.

The fallback control strategy algorithm implemented within our software is presented in Algorithm 3. We stress that once the fallback is triggered, the stability certificates are possibly no longer preserved, since the fallback may not ensure the associated Lyapunov function indeed decreases. However, in most practical cases, such as in Sanches *et al.* (2025c), this strategy indeed works and often also preserves the decay of the associated Lyapunov candidate.

3.2.8 Overview of the proposed solver

After the exposed in this Chapter, we highlight the key aspects of the proposed solver as follows:

⁴ Here, “real-time” refers to meeting sampling deadlines via a fallback strategy, as deterministic worst-case execution times (WCET) for the iterative solver are not guaranteed.

Algorithm 3 fallback_control

```

1: procedure FALLBACK_CONTROL( $u_{\text{prev}}, u_{\text{incomplete}}, x_{\text{prev}}, x_{\text{ref}}, \text{mpc}$ )
2:    $\text{pred\_x\_1} \leftarrow \text{mpc.A} \cdot x_{\text{prev}} + \text{mpc.B} \cdot u_{\text{prev}}$ 
3:    $\text{pred\_x\_2} \leftarrow \text{mpc.A} \cdot x_{\text{prev}} + \text{mpc.B} \cdot u_{\text{incomplete}}$ 
4:    $J_1 \leftarrow (x_{\text{ref}} - \text{pred\_x\_1})^\top \cdot \text{mpc.Qe} \cdot (x_{\text{ref}} - \text{pred\_x\_1})$ 
5:    $J_2 \leftarrow (x_{\text{ref}} - \text{pred\_x\_2})^\top \cdot \text{mpc.Qe} \cdot (x_{\text{ref}} - \text{pred\_x\_2})$ 
6:   if  $J_1 \leq J_2$  then
7:     return  $u_{\text{prev}}$ 
8:   else
9:     return  $u_{\text{incomplete}}$ 

```

- The algorithm is based in Interior-Point algorithms (Boyd, 2004).
- It uses Logarithmic-Barrier to constraint the closed-loop system.
- It uses complementary techniques for accelerating convergence rate, such as using t constant (as suggested in Wang and Boyd (2009)), custom linear decomposition algorithm and warm-start.
- A fallback control strategy is applied in order to avoid any occasional sampling overrun. This practice is fundamental for MPC embedding in systems with hard real-time constraints.

3.3 RESULTS

Although the focus of this dissertation is in the microgrid's context, we also experimented other challenging nonlinear systems in order to evaluate the solver's high accuracy and low computational time associated. Therefore, this section presents important results obtained using the proposed solver.

3.3.1 Twin Rotor MIMO System (TRMS)

The Twin Rotor MIMO System (TRMS) is a classical example of challenging system for control engineering, since it has severe nonlinear dynamics. The TRMS consists of a pendulum balanced by two rotors coupled on the top, one being a tail rotor, which balance the vertical axis of the system, while the other one (main rotor) balance the system horizontally. The TRMS is represented in Figure 4 (extracted from Rotondo, Nejjari, and Puig (2013)).

The control objective is to ensure that the horizontal $\theta_h(t)$ and vertical $\theta_v(t)$ angles track sinusoidal references. To achieve this, the angular velocities of the horizontal (ω_h) and vertical (ω_v) fans are controlled by manipulating the voltage inputs u_h and u_v . The control actions are computed by the LPV-PILOT solver, with the TRMS system simulated in MATLAB.

The TRMS dynamic model is described by the following differential equation from Rotondo, Nejjari, and Puig (2013):

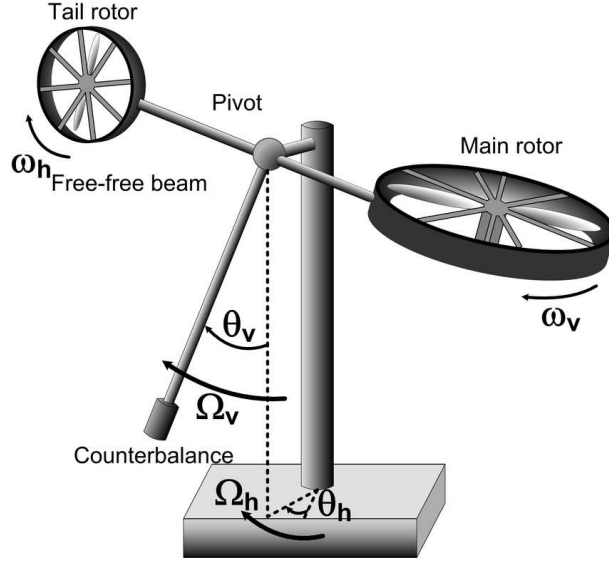


Figure 4 – TRMS structure (Rotondo; Nejjari; Puig, 2013).

$$\begin{bmatrix} \dot{\omega}_h(t) \\ \dot{\Omega}_h(t) \\ \dot{\theta}_h(t) \\ \dot{\omega}_v(t) \\ \dot{\Omega}_v(t) \\ \dot{\theta}_v(t) \end{bmatrix} = \mathbf{A}_c(\rho(t)) \begin{bmatrix} \omega_h(t) \\ \Omega_h(t) \\ \theta_h(t) \\ \omega_v(t) \\ \Omega_v(t) \\ \theta_v(t) - \theta_v^0 \end{bmatrix} + \begin{bmatrix} \rho_{12}(t) & 0 \\ 0 & \rho_{13}(t) \\ 0 & 0 \\ 0 & \rho_{14}(t) \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_h(t) \\ u_v(t) \end{bmatrix} \quad (17)$$

with the scheduling parameters $\rho(t) = [\rho_1(t), \rho_2(t), \dots, \rho_{12}(t)]^\top$ being extracted from Rotondo, Nejjari, and Puig (2013) and presented in Appendix C. Ω_v and Ω_h represents, respectively, the vertical and horizontal angular velocity of the rigid body. The affine matrix (dependent of the scheduling variable) is given by:

$$\mathbf{A}_c(\rho(t)) = \begin{bmatrix} \rho_1(t) & 0 & 0 & 0 & 0 & 0 \\ \rho_2(t) & \rho_3(t) & \rho_4(t) & \rho_5(t) & \rho_6(t) & \rho_7(t) \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho_8(t) & 0 & 0 \\ 0 & \rho_9(t) & 0 & \rho_{10}(t) & \rho_{11}(t) & \rho_{12}(t) \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

The optimal control problem is formulated based on (14). The sampling time is set to $T_s = 100$ ms, which is chosen to be significantly smaller than the system's settling time, ensuring adequate capturing of the closed-loop dynamics.

The computational effort for solving the algorithms came from a personal computer (PC) equipped with an Intel Core i5-12400T CPU @ 3.2 GHz, 16 GB RAM, running Windows 10 Pro 64-bit. The following table synthesizes the MPC parameters used in this simulation.

The results are presented in Figure 5. The figure shows that the closed-loop system trajectories succeed in tracking with high accuracy the sinusoidal reference provided. Such reference

MPC Parameter	Value
N_p (prediction horizon)	5 samples
Q (state weighting)	$\text{diag}(1, 50, 1, 1, 50, 1)$
Q (terminal weighting)	$\text{diag}(1, 50, 1, 1, 50, 1)$
R (control weighting)	$\text{diag}(1, 1)$

Table 2 – MPC parameters: TRMS experiment.

excites the system at different operating points, which demonstrate the solver’s suitability to control complex, nonlinear systems. Furthermore, the computational time per sample was evaluated

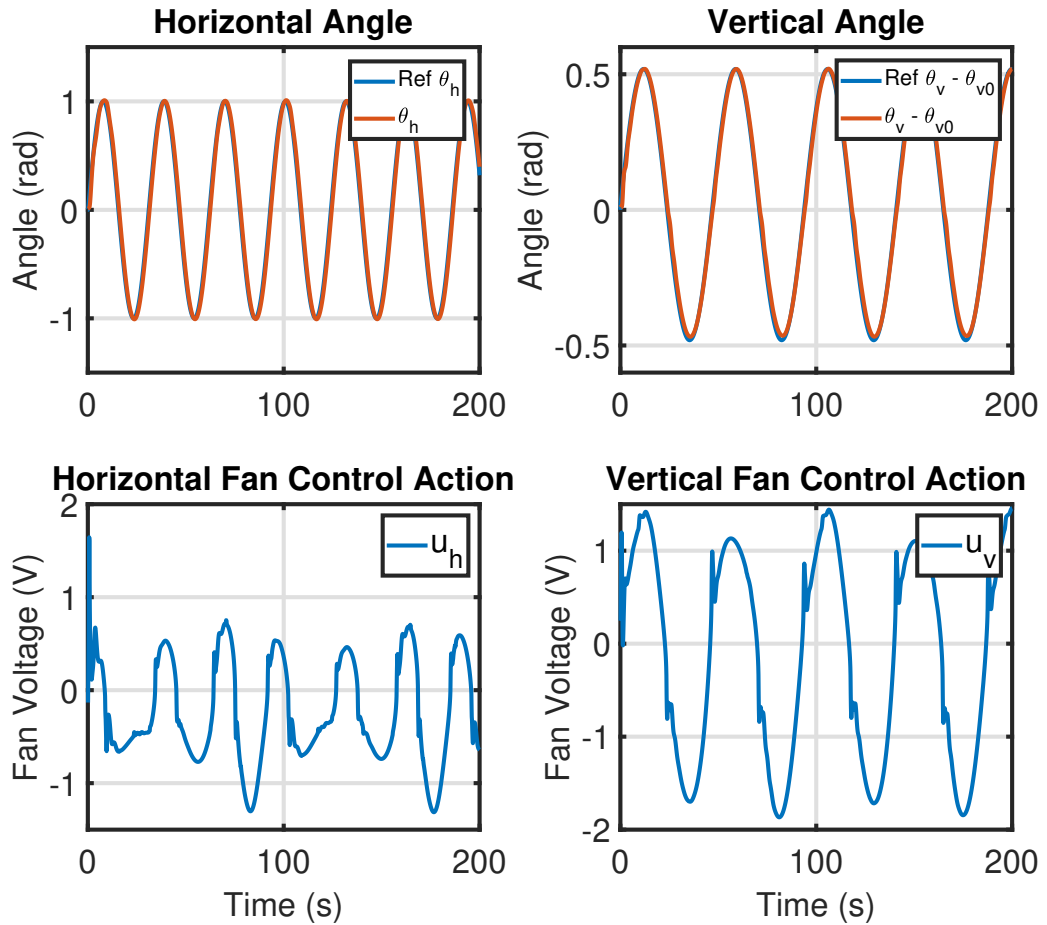


Figure 5 – Twin Rotor MIMO System control using LPV-PILOT.

and the corresponding minimal, maximal, mean and standard deviation values are presented in Table 3. The observed computational time highlights the fast convergence rate of the proposed solver, even when controlling systems with complex dynamics. This is evidenced by the reduced average value of 0.51 ms per sample.

$t_{c\text{-min}}$	$t_{c\text{-mean}}$	$t_{c\text{-max}}$	$t_{c\text{-std}}$
0 ms	0.51 ms	31.2 ms	2.81 ms

Table 3 – Computational time statistics for the TRMS benchmark.

3.3.2 Automatic helicopter flight

To validate the performance of the embedded controller, a representative automatic helicopter flight scenario was developed. The problem is described as follows: the helicopter is initially steady at the top of a building at the position $(X, Y) = (50, 50)$ m; it must fly over another three buildings ahead in the X -direction, and land in the rooftop of a fifth building, 100 m high, located at $(X, Y) = (250\text{m}, 100\text{m})$. The horizontal and vertical acceleration (a_x, a_y) of the helicopter is controlled automatically by the controller embedded in the ESP32 microcontroller, while the path planning is pre-computed offline to ensure collision avoidance.

The helicopter motion dynamics for the helicopter's position (p_x, p_y) and velocity (v_x, v_y) is given by the following equations:

$$\begin{aligned}
 \frac{dv_x(t)}{dt} &= a_x(t) \pm k_x |v_x(t)|^2, \\
 \frac{dp_x(t)}{dt} &= v_x(t), \\
 \frac{dv_y(t)}{dt} &= a_y(t) \pm k_y |v_y(t)|^2 - g, \\
 \frac{dp_y(t)}{dt} &= v_y(t),
 \end{aligned} \tag{18}$$

with $k_x = 0.10\text{m}^{-1}$ and $k_y = 0.08\text{m}^{-1}$ being the air friction coefficient for the horizontal and vertical motion, respectively. To handle the nonlinearities in (18) we relied on a qLPV embedding to represent the system. To this end, we considered $\rho_1(k) = -k_x v_x(k)$ and $\rho_2(k) = -k_y v_y(k)$. Then, by using Euler's discretisation we reach a state-space LPV representation of the helicopter behaviour:

$$\begin{aligned}
 \begin{bmatrix} p_x(k+1) \\ v_x(k+1) \\ p_y(k+1) \\ v_y(k+1) \end{bmatrix} &= \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 + T_s \rho_1(k) & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 + T_s \rho_2(k) \end{bmatrix} \begin{bmatrix} p_x(k) \\ v_x(k) \\ p_y(k) \\ v_y(k) \end{bmatrix} \\
 &+ \begin{bmatrix} 0 & 0 \\ T_s & 0 \\ 0 & 0 \\ 0 & T_s \end{bmatrix} \begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -T_s \end{bmatrix} [g].
 \end{aligned} \tag{19}$$

where $T_s = 100\text{ms}$ is the sampling time.

The physical constraints were considered: the helicopter acceleration limits are defined by $\mathcal{U} = \{u \in \mathbb{R}^2 : |a_x| \leq 20 \text{ m/s}^2, |a_y| \leq 20 \text{ m/s}^2\}$ and the states constraints are given by $\mathcal{X} = \{x \in \mathbb{R}^4 : x_{\min} \leq x \leq x_{\max}\}$, with the lower bound $x_{\min} = \{-100\text{m}, -40\text{m/s}, 0\text{m}, -40\text{m/s}\}$ and the upper bound $x_{\max} = \{100\text{m}, 40\text{m/s}, 100\text{m}, 40\text{m/s}\}$. Note that the negative constraints related to x_1 represent locations to the left of the origin.

For this scenario, a closed-loop system embedded with an ESP32 microcontroller (low computational resources: memory and processing) is proposed. For this purpose, a processor-in-the-loop simulation of an automatic helicopter flight is set up according to the scheme in Figure 6.

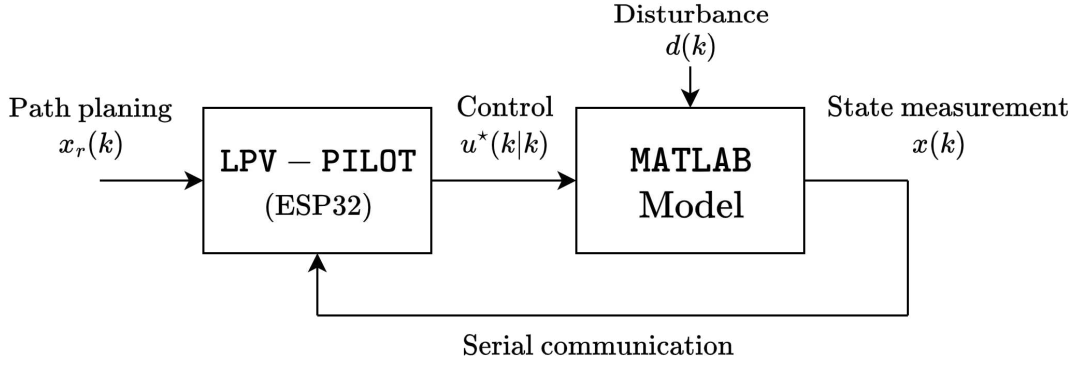


Figure 6 – Processor-in-the-loop scheme.

The code embedded in ESP32 runs as a C implementation and is derived from the MATLAB code of the LPV-PILOT. Focusing on performance, particularly given the microcontroller's limited memory and processing power, the MATLAB code was manually translated and specifically optimized for the helicopter landing scenario by exploiting the sparsity structure of the involved matrices. The experimental configuration involves a processor-in-the-loop (PIL) scheme, with the LPV-PILOT algorithm embedded in C/C++ in an ESP32, with the helicopter model running in a computer in parallel. A picture of this scheme is presented in Figure 7.

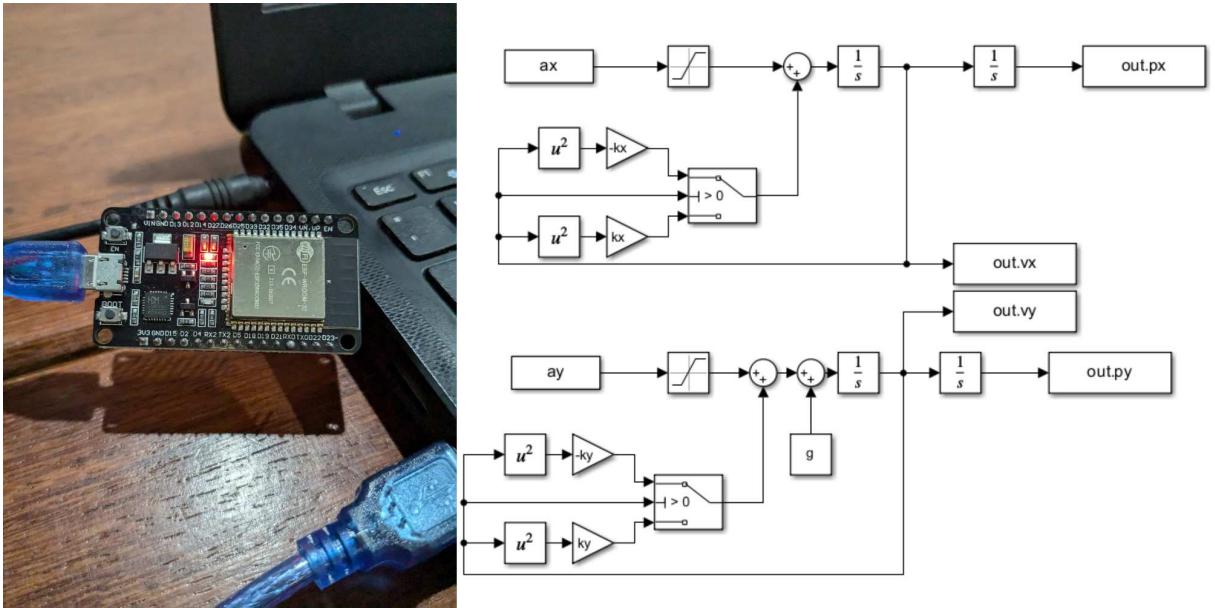


Figure 7 – PIL scheme: ESP32 microcontroller and SimuLink.

Finally, the MPC parameters used in this simulation are presented in Table 4. While the cost function and restrictions are formulated with respect to the optimization problem in (14).

MPC Parameter	Value
N_p (prediction horizon)	5 samples
Q (state weighting)	$\text{diag}(1.0, 0.1, 1.0, 0.1)$
Q (terminal weighting)	$\text{diag}(1.0, 0.1, 1.0, 0.1)$
R (control weighting)	$\text{diag}(0.01, 0.01)$

Table 4 – MPC parameters: helicopter experiment.

With all the experiment set, the simulation was performed in the presented PIL scheme and the results are presented in the following: Figure 8 illustrates the 2D helicopter motion among the buildings, Figure 9 presents the horizontal and vertical motion along the X and Y coordinates separately, and Figure 10 shows the computational time required to solve (16) per sample.

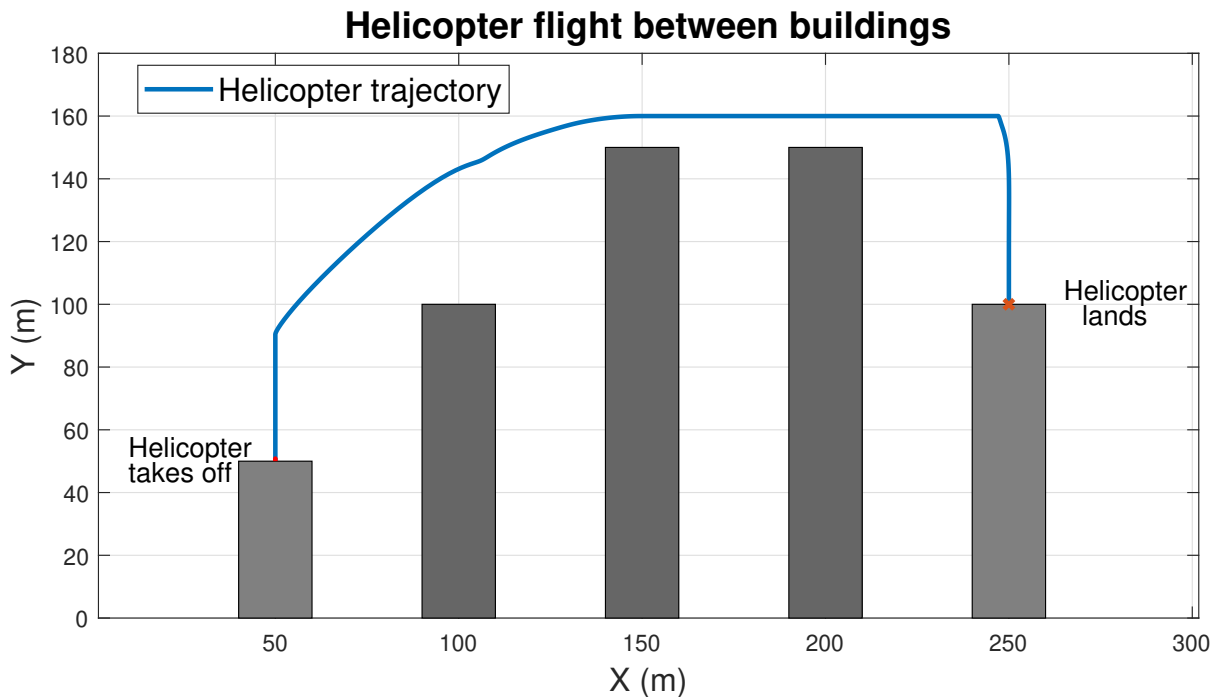


Figure 8 – Helicopter landing between buildings using LPV-PILOT.

Indeed, the code embedded in the ESP32 successfully controlled the helicopter, allowing it to fly between buildings and land smoothly at the final position. However, the most significant result concerns the computational time (Figure 10), which remained below the sampling time ($T_s = 100$ ms) throughout the entire simulation, with a mean execution⁵ time of 12.23 ms for solving (16).

⁵ Per comparison, we stress that in Sanches *et al.* (2025c) we presented a preliminary version of the embedded code, in similar scenario, achieving an average computational time of 85 ms—approximately eight times slower than the current result.

Horizontal and Vertical Motion

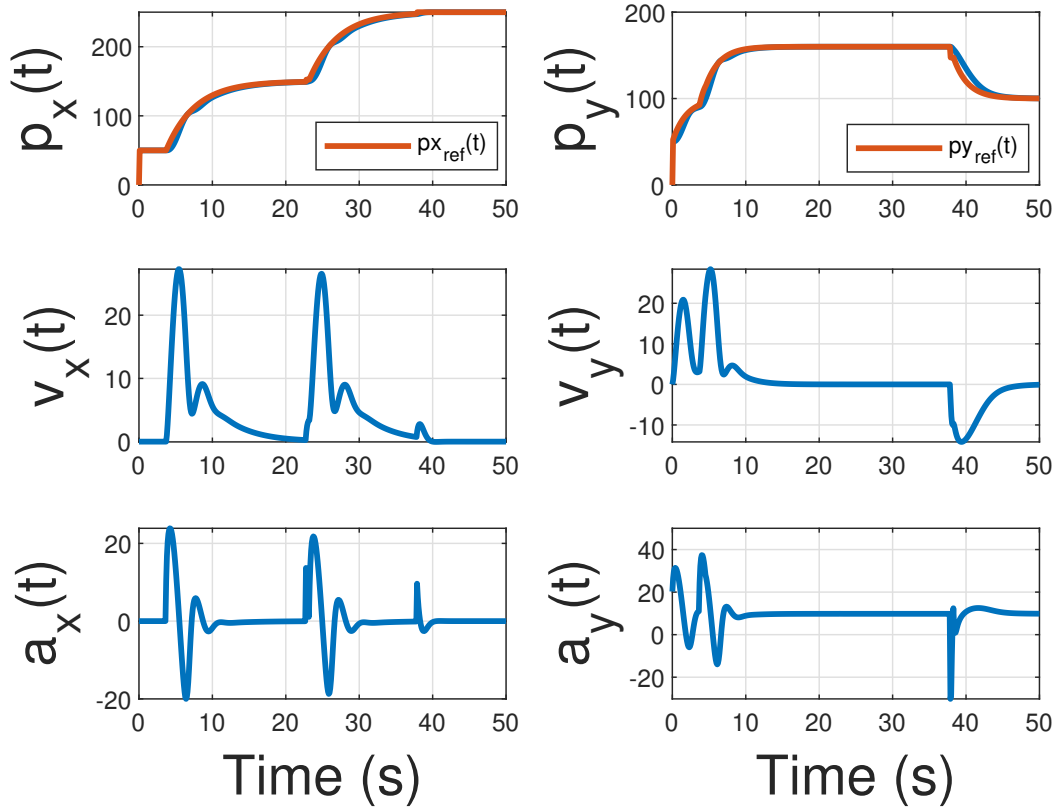


Figure 9 – Helicopter motion along the X and Y coordinates.

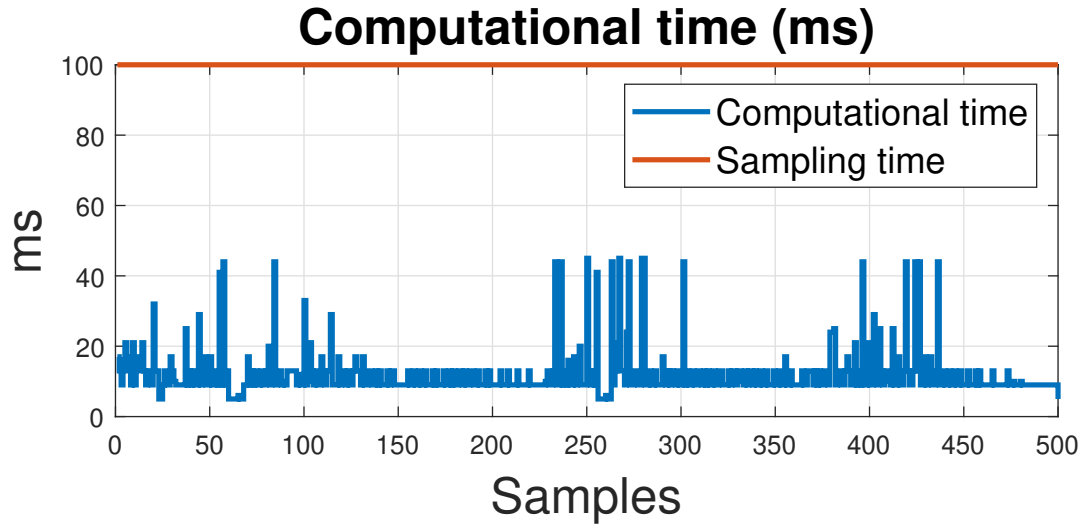


Figure 10 – Computational time for helicopter landing control.

3.3.3 Comparison with CasADi

In order to evaluate and benchmark our proposed LPV-PILOT solver, we provide, next, a thorough comparison result. In particular, we test the numerical effectiveness and control performance of the NMPC schemes implemented using LPV-PILOT and CasADi (Andersson *et*

al., 2019). CasADi is a well-validated solver in the context of NMPC applications in academia. It is widely adopted for optimal control, for both offline optimization and real-time problems.

The key difference to be highlighted is that the CasADi-*ipopt* NMPC seeks to solve an NLP, while our LPV-PILOT is devoted to solving a QP derived from an appropriate qLPV embedding with a corresponding frozen/gain-scheduled parameter trajectory guess and robustness ingredients.

To evaluate the performance of both NMPC from CasADi and LPV MPC from LPV-PILOT, the regulation of an inverted pendulum on a cart is proposed. Such system is a classical and challenging control example, and its scheme is represented in Figure 11.

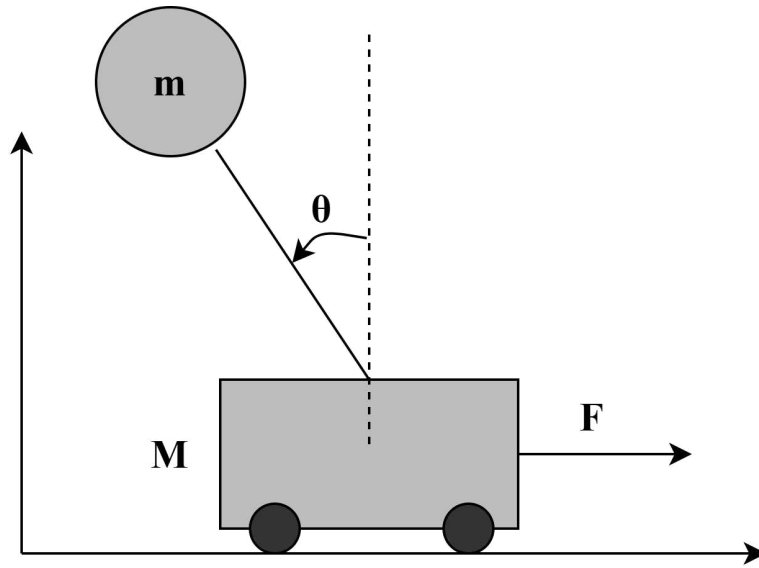


Figure 11 – Inverted pendulum on a cart scheme.

The goal of this system is to balance the pendulum angle $\theta(t)$ by controlling the force $F(t)$ applied in the cart. The challenge of this system is to regulate the angular position at the same time as tracking the cart position $p(t)$. The model describing the relation between the force applied in the cart $F(t)$, the cart position $p(t)$, the pendulum angle $\theta(t)$, angular velocity $\omega(t)$ and cart velocity $v(t)$ is defined:

$$\begin{aligned}
 \dot{p} &= v, \\
 \dot{\theta} &= \omega, \\
 \dot{v} &= \frac{-ml \sin(\theta) \dot{\theta}^2 + mg \cos(\theta) \sin(\theta) + F}{M + m - m(\cos(\theta))^2}, \\
 \dot{\omega} &= \frac{-ml \cos(\theta) \sin(\theta) \dot{\theta}^2 + F \cos(\theta) + (M + m)g \sin(\theta)}{l(M + m - m(\cos(\theta))^2)},
 \end{aligned} \tag{20}$$

where M and m are the mass of the cart and the pendulum, respectively, l is the pendulum length and g is the gravity acceleration. The particularity of this model lies in its highly accentuated nonlinearity and the presence of only two equilibrium points, making it sensitive to disturbances

and control inaccuracies that can easily lead to instability. As a result, classical controllers struggle to regulate the system and often fail to achieve satisfactory performance.

Then, CasADi and the proposed solver LPV-PILOT were tested through this system to verify its performance in regulating the inverted pendulum. CasADi uses the nonlinear model presented in (20). However, to apply the LPV-PILOT it is necessary to incorporate (20) into the LPV framework. To do so, we consider a set of scheduling variables that capture the nonlinear dependencies of the system on the state trajectory:

$$\begin{aligned}\rho_1(k) &= \frac{\sin(\theta(k)) \omega(k)}{M + m - m \cos^2(\theta(k))}, \\ \rho_2(k) &= \frac{\cos(\theta(k)) \sin(\theta(k))}{(M + m - m \cos^2(\theta(k))) \theta(k)}, \\ \rho_3(k) &= \frac{\cos(\theta(k)) \sin(\theta(k)) \omega(k)}{M + m - m \cos^2(\theta(k))}, \\ \rho_4(k) &= \frac{\sin(\theta(k))}{(M + m - m \cos^2(\theta(k))) \theta(k)}, \\ \rho_5(k) &= \frac{1}{M + m - m \cos^2(\theta(k))}, \\ \rho_6(k) &= \frac{\cos(\theta(k))}{M + m - m \cos^2(\theta(k))}.\end{aligned}$$

Using these scheduling variables, the nonlinear system is embedded into the LPV structure by defining the state-space equation:

$$\begin{aligned}\begin{bmatrix} p(k+1) \\ \theta(k+1) \\ v(k+1) \\ \omega(k+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & T_s(mg\rho_2(k)) & 1 & T_s(-ml\rho_1(k)) \\ 0 & T_s\left(\frac{(M+m)g}{l}\rho_4(k)\right) & 0 & 1 + T_s(-m\rho_3(k)) \end{bmatrix}}_{A(\rho(k))} \begin{bmatrix} p(k) \\ \theta(k) \\ v(k) \\ \omega(k) \end{bmatrix} \\ &\quad + \underbrace{\begin{bmatrix} 0 \\ 0 \\ T_s\rho_5(k) \\ T_s\left(\frac{1}{l}\rho_6(k)\right) \end{bmatrix}}_{B(\rho(k))} F(k)\end{aligned}\tag{21}$$

where T_s is the sampling time, which is chosen to be sufficiently (around 20 times) faster than the settling time. The choice for T_s is crucial for the closed-loop control and, as consequence, impact directly in the final computational time due to its relation with the prediction horizon.

Once everything is set, we simulate both solvers into the same conditions. The model and controller parameters are presented in Table 5. The optimization problem is formulated using (14) with the LPV model and constraints detailed in this section.

Therefore, the closed-loop pendulum behaviour comparing both solvers is presented in Figure 12, while the cost comparison is depicted in Figure 13. Moreover, an online computational time comparison is made and presented in Figure 14. The related performance metrics are

Category	Parameter	CasADi (NMPC)	LPV-PILOT
Model	M	1 kg	
	m	0.1 kg	
	l	0.8 m	
	T_s	0.1 s	
	X constraints	$ X \leq [15, \pi, 10, 15]$	
	U constraints	$ U \leq 20$	
	Initial condition	$X_0 = [1, 0.1, 0, -0.045], U_0 = 0$	
MPC	N_p	50 samples	
	ξ	0%	45%
	Q (state)	diag([9.6%, 38.4%, 0.038%, 0.038%])	
	Q (terminal)	diag([9.6%, 38.4%, 0.038%, 0.038%])	
	R	3.8%	

Table 5 – Simulation parameters used in both control strategies.

presented in Table 6, with $C_{n_{pts}} = \sum_{k=0}^{n_{pts}} \ell(x(k), u(k))$ being the true aggregated cost observed over time along n_{pts} discrete-time samples and RMS the Root Mean Square metric.

Solver	$C_{n_{pts}}$	RMS (J_k^*)	$t_{c\text{-mean}}$
LPV-PILOT	1940.6	0.2336	29.1 ms
CasADi-ipopt	1937.8	0.2482	34.0 ms

Table 6 – Performance metrics comparison between LPV-PILOT and CasADi controllers.

The results in Figure 12 and 13 show that the response obtained by the LPV-PILOT is not exactly the same as the one obtained by CasADi, which search for the exact solution of the nonlinear programming problem. However, the dynamic behaviour is, from a practical point of view, identical. Moreover, the slight advantage of CasADi (in optimality terms) is compensated by the LPV-PILOT average computational time, that is 14.41% faster than CasADi.

3.3.4 Path-following in a real-world propeller driven sailboat: experimental validation

Lastly, the LPV-PILOT was evaluated under a real-world system, with real-time hard constraints, cf. (Sanches *et al.*, 2025a). To this end, we explored the path-following problem in a real-world propeller driven sailboat. The objective is controlling the boat's rudder angle to maintain the vessel's course towards a determined direction. To this end, we used Matina III, a vessel located in Bombinhas, a coastal city in the south of Brazil, which is presented in the Figure 15.

For path-following control purposes, the sailboat was pre-equipped with a 12V DC motor pulley-coupled to the helm wheel (as illustrated in Figure 16). Then, we retrofit this DC motor to be controlled by an ESP32 microcontroller via a PWM H-Bridge, using a BTS7960 commercial driver. To the vessel's position (latitude p_y and longitude p_x), we used a NEO-6M

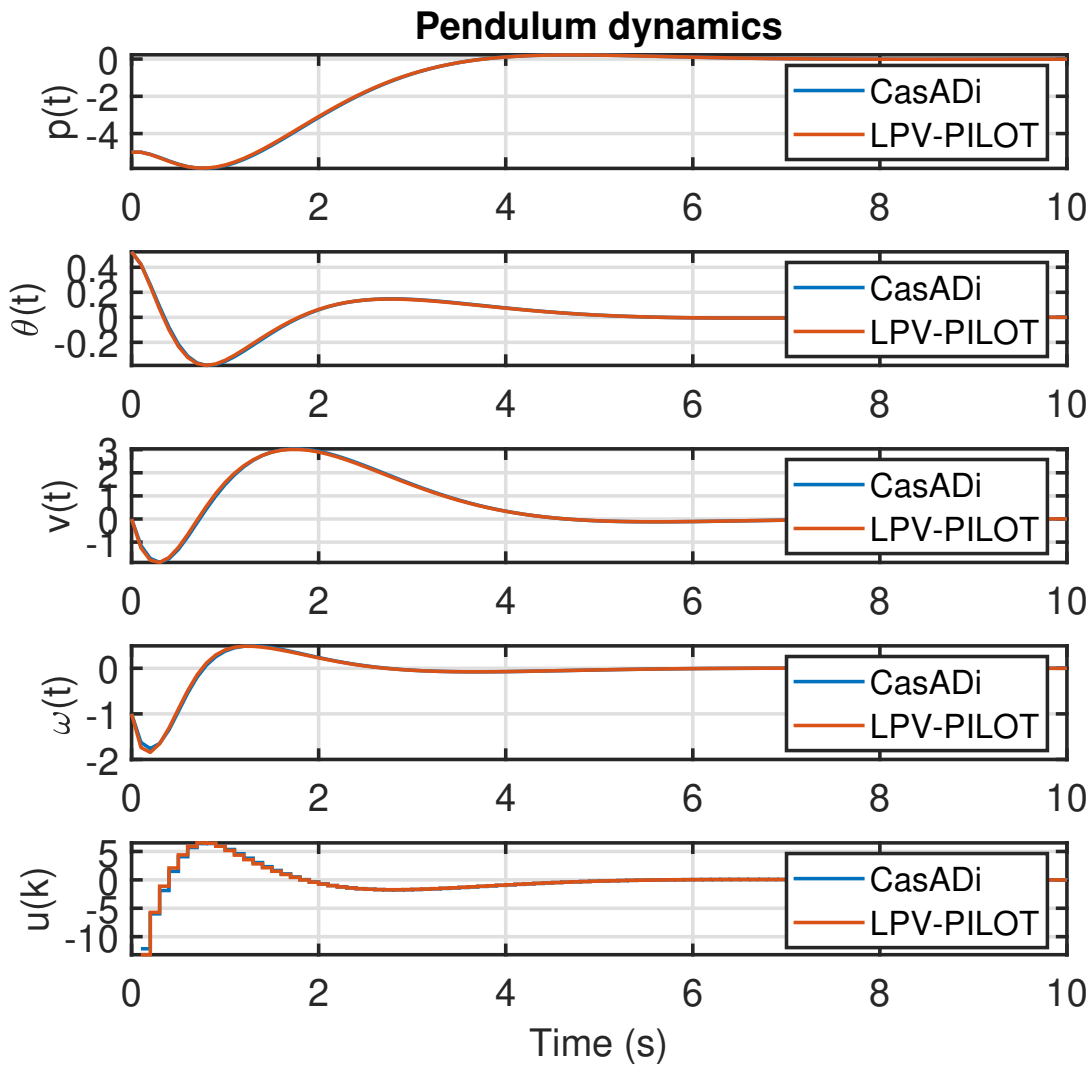


Figure 12 – Pendulum dynamics comparison.

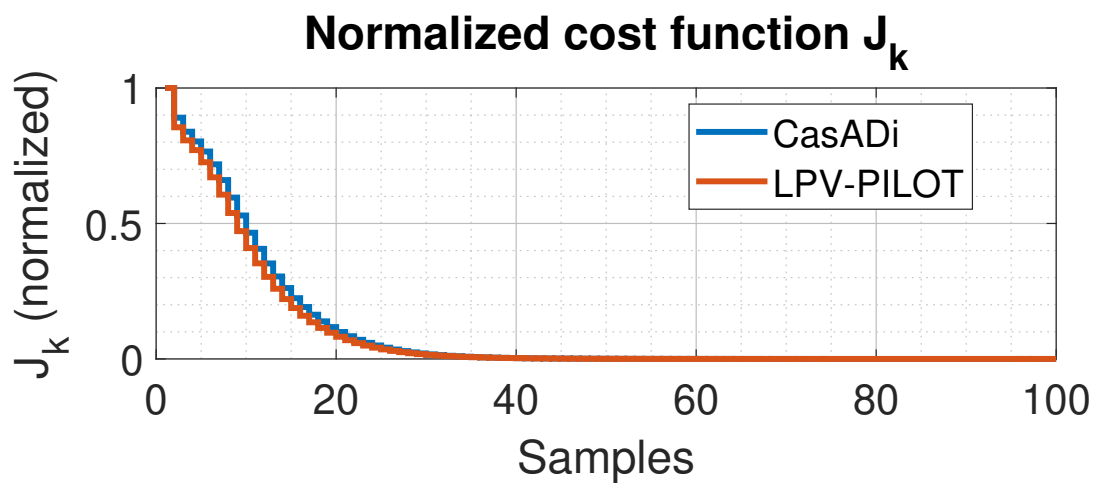


Figure 13 – Cost function evolution for the inverted pendulum benchmark.

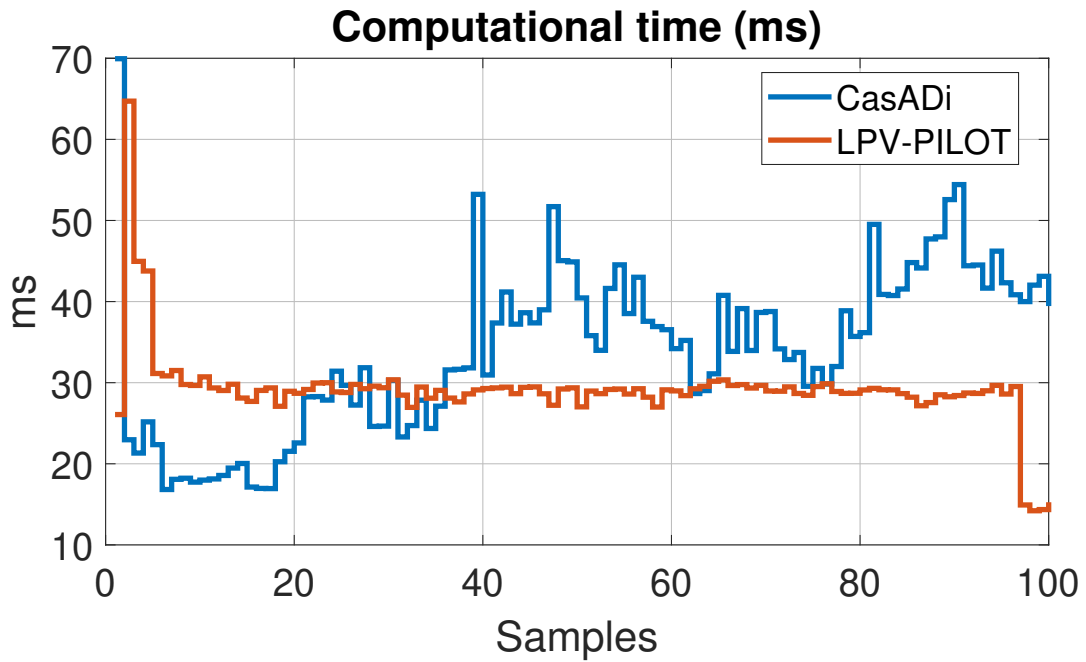


Figure 14 – Computational time comparison (CasADi $\times$ LPV-PILOT for Inverted Pendulum).



Figure 15 – Matina III: a 32-foot sailboat (*Brasilia 32*).

GPS module, which communicates with the ESP32 via UART (serial). The vessel's position is digitally filtered to increase measurement precision. Lastly, the rudder angle is measured by a potentiometer connected to the rudderstock at the vessel's stern (Figure 17).

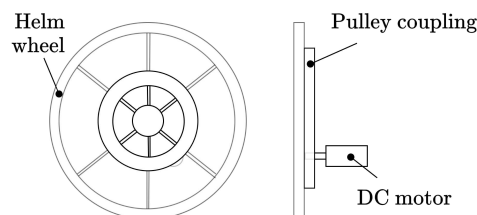


Figure 16 – Motor-helm wheel coupling illustrative scheme.

The electrical circuit scheme of the embedded ESP32 with the BTS7960 motor driver, NEO-6M GPS and potentiometer is presented in Figure 18.

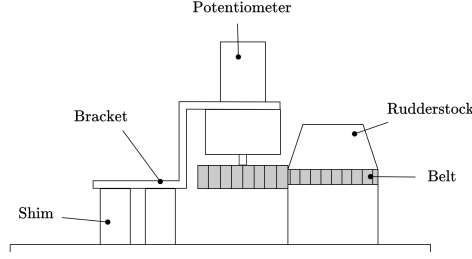


Figure 17 – Rudder angle measurement using potentiometer.

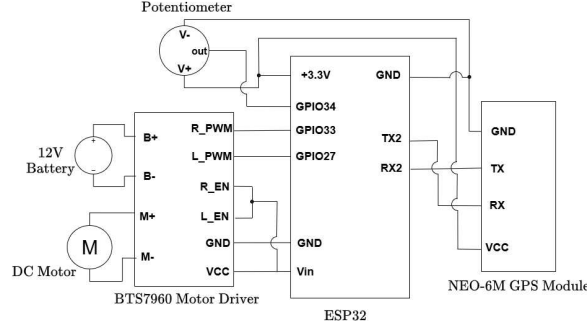


Figure 18 – Electrical circuit and ESP32 board.

The circuit is centered around the ESP32 microcontroller. To control the actuator, the ESP32 sends Pulse Width Modulation (PWM) signals to the BTS7960 motor driver, which manages the power delivered by the 12V battery to the DC motor coupled to the helm wheel. The motor's direction and angular speed are controlled by the R_PWM and L_PWM signals from GPIO pins 33 and 27, respectively.

Overall, the proposed closed-loop system receives the vessel's position and heading from a GPS, while operates a DC motor to turn the helm-wheel adequately for path-following purposes (Sanches *et al.*, 2025a). To control the DC motor, we relied on an LPV NMPC that uses LPV-PILOT for computing the optimal control command. The complete hardware-in-the-loop (HIL) scheme is presented in Figure 19.

The vessel's control-oriented model was identified experimentally, and is given by:

$$\begin{aligned}
 \begin{bmatrix} \gamma_{\text{error}}(k+1) \\ \theta_{\text{error}}(k+1) \\ \dot{\theta}_{\text{error}}(k+1) \end{bmatrix} &= \begin{bmatrix} 1 & (T_s \rho(k)) & 0 \\ 0 & 1 & T_s \\ 0 & 0 & (1 + \frac{T_s}{\tau}) \end{bmatrix} \begin{bmatrix} \gamma_{\text{error}}(k) \\ \theta_{\text{error}}(k) \\ \dot{\theta}_{\text{error}}(k) \end{bmatrix} \\
 &+ \begin{bmatrix} 0 \\ 0 \\ T_s \frac{K}{\tau} \end{bmatrix} [\delta(k)], \quad (22)
 \end{aligned}$$

where γ_{error} is the path lateral error, θ_{error} is the heading error and $\dot{\theta}_{\text{error}}$ is the heading error rate. $\delta(k)$ is the rudder angle, provided by the NMPC LPV algorithm. K and τ are fixed identified

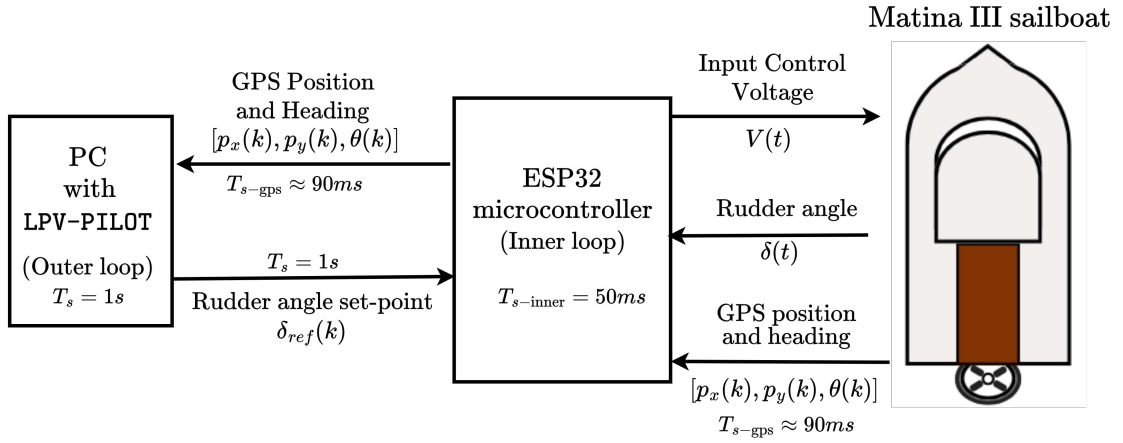


Figure 19 – Control system architecture.

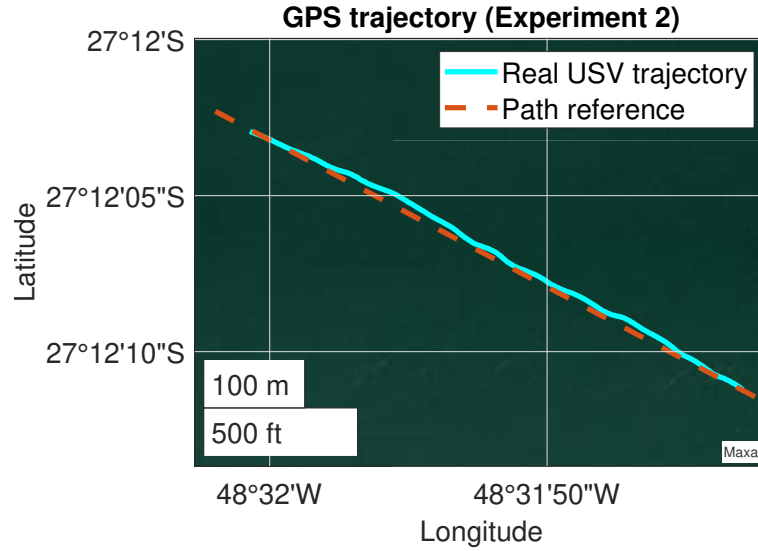


Figure 20 – Satellite view of the path-following.

parameters, T_s is the sampling time, and $\rho(k)$ is the scheduling variable, defined by:

$$\rho(t) = \begin{cases} v(t), & \text{if } |\theta_{\text{error}}(t)| < 10^{-5}, \\ \frac{v(t) \sin(\theta_{\text{error}}(t))}{\theta_{\text{error}}(t)}, & \text{otherwise.} \end{cases} \quad (23)$$

The optimization control problem is formulated using (14), with the model and constraints developed in this section. Then, a representative result of path-following was extracted from the original paper in Sanches *et al.* (2025a) and is presented in the following. Moreover, a video illustrating the proposed path-following is available online ⁶.

Figure 20 shows a satellite point of view of the path-following. It highlights the accuracy of the controller and effectiveness in maintaining the vessel in the trajectory.

⁶ <https://youtu.be/uFlsUjcfmY4>

Moreover, the computational time (Figure 21) remains favorable to real-time operation, achieving $t_{c,\text{mean}} = 39.86$ ms. This result is numerically significant, in the sense that the prediction horizon was increased from $N_p = 30$ s to $N_p = 50$ s (50 samples), i.e. the decision variable Z_k is defined in $\mathbb{R}^{(n_x \times n_u)N_p}$.

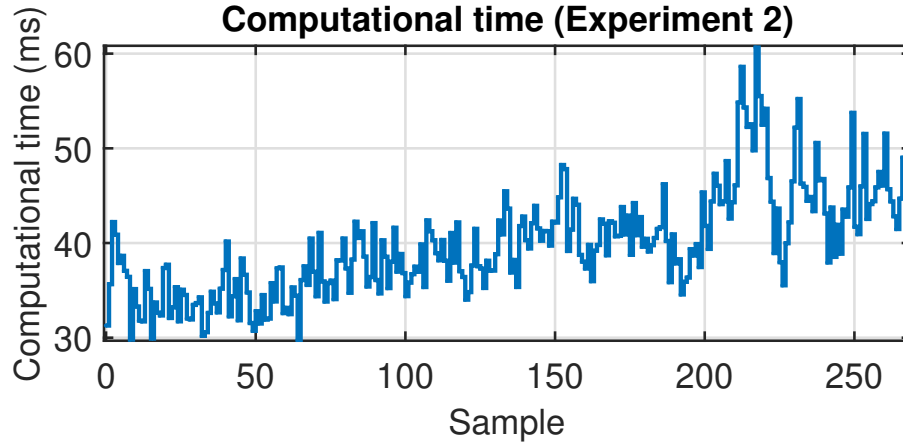


Figure 21 – Computational time (Experiment 2).

This experiment highlight the well-functioning and high accuracy of the proposed path-following algorithm using LPV-PILOT in a real-world context.

3.4 A SUMMARY OF FINDINGS

This chapter presented the LPV-PILOT, a dedicated new solver for NMPC using LPV framework. It presents the fundamental theory behind the algorithm, as well as highlight its key aspects, denoting how such strategies effectively increase convergence rate.

Moreover, numerical and experimental results were presented, highlighting the solver potential of solving challenging optimization problems within very strict sampling times. Such results denote the high suitability of the proposed algorithm for embedded nonlinear systems with real-time constraints.

In the sequel, we present how this solver was applied to the microgrid context, as well as the respective results validating the proposed solution.

4 RENEWABLE MICROGRID APPLICATION

Taking into account the introduction and contextualization of the problem (Chapter 1), the literature review and identified research gap (Chapter 2), the proposed solution and results (Chapter 3), we have set the fundamental theoretical basis necessary for developing the MPC application for renewable microgrid systems. Hence, in this chapter, we present the development, instrumentation, modelling and identification of a low-cost small-scale renewable microgrid used for control validation and outreach activities (scientific disclosure). Initially, we evaluate the solver for a numerical scenario and, then, we experimentally validate its suitability for the real plant. The results presented in this chapter will highlight the strengths of the solver and its importance for renewable microgrids.

Some extracts of this Chapter were extracted from the paper of Sanches *et al.* (2025d).

4.1 LOW-COST SMALL-SCALE RENEWABLE MICROGRID

The previous chapters outlined the importance of microgrids in the context of the energy transition. Specifically, we discussed the control challenges imposed by the nonlinear dynamics of such systems and the existing gap in efficient embedded solvers. Consequently, Chapter 3 introduced `LPV-PILOT`, a dedicated solver that addresses these issues by combining LPV representations with efficient interior-point optimization techniques, which was validated across various benchmark scenarios. Moreover, we highlighted this topic as a key area in recent research, especially in the field of control engineering.

Despite efforts to develop new algorithms for energy management in microgrids, their validation in real scenarios faces an important challenge. In practice, the experimental validation of MGCCs is difficult due to meteorological conditions. For example, consider an MGCC designed to manage a solar-based microgrid on a rainy day. The adequate weather conditions for testing the specific algorithm can take days, weeks, or even months to occur – which hinders experimental testing and validation.

Therefore, we thoroughly validate a low-cost scaled renewable microgrid setup, dedicated for experimental control tests. Our test-bench also serves for outreach activities, i.e. scientific disclosure on the role of automatic control in the energy transition. The experimental kit is composed of a wind turbine, a photovoltaic (PV) cell, a green hydrogen storage system (electrolyser, H_2 tank and fuel cell) and an emulated battery, whose SoC dynamics is represented in a central microcontroller that communicates with the microgrid system.

Accordingly, local models and control schemes are developed for each component, such that different weather conditions can be emulated for research purposes. In particular, some of the microgrid subsystems are time-varying and nonlinear, and thus Linear Parameter Varying (LPV) descriptions are applied, together with local gain-scheduling control, following the approach of Mohammadpour and Scherer (2012).

Then, in this section, we detail each of the renewable subsystems of the proposed small-

scale microgrid, as well as their modelling, instrumentation and local embedded control aspects. These local controllers are responsible for the low-level management of each subsystem, allowing these inherently intermittent resources to be integrated and managed within the emulated practical setup. The MGCC has, then, the task of managing the energy bus and coordinating the local subsystems, which is implemented via appropriate communication protocols (described later).

Our didactic MG setup comprises the following DERs:

- A **wind turbine emulator**: a DC motor (controlled via PWM) with propellers, used to generate different wind profiles, and a wind turbine (rotor) that spins due to the artificially generated wind, converting kinetic energy into electricity (Section 4.1.1).
- A **PV emulator**: this component is composed by bright LED bulb (controlled via PWM), which is used to emulate different solar irradiance profiles, and a PV cell that converts the incident light to electrical energy (Section 4.1.2).

The corresponding renewable generation (by these DERs) is, then, sent to the MGCC bus, which decides whether to store it or use it directly to meet some emulated load profile. The corresponding ESSs are the following:

- A green hydrogen subsystem: a controllable **electrolyser** that produces H_2 from distilled water, a **tank** that stores the generated green H_2 , and a **fuel cell** that can be controlled to consume the hydrogen and generate electrical energy (Sec. 4.1.3).
- An emulated battery: a virtual storage system that accounts for the energy produced by the DERs and virtually stores it for later usage (Sec. 4.1.4).

In Figure 22, we provide a schematic overview¹ of the developed renewable microgrid setup, with its DERs and ESSs components and the MGCC bus.

4.1.1 Wind turbine

The first renewable DER in our scaled microgrid is a small horizontal axis wind turbine (HAWT). Our HAWT setup is composed of blades connected to a rotor which, in turn, is coupled to a generator (Silva *et al.*, 2025). The rotor converts the kinetic energy from the wind speed (determined by a DC motor) into electrical energy, which is “sent” to the main bus of our MGCC.

In order to artificially generate the wind, the DC motor brushless operates, basically, as a fan, which is controlled via PWM by a low-cost Arduino UNO board. To accurately generate a specific wind profile, it acquires real-time output from the wind turbine generator, and drives the brushless motor using an electronic speed control (ESC) integrated circuit. The prototype physical structure is made of plastic via additive manufacturing, as presented in Figure 23.

The actuation and measurement circuit of the HAWT subsystem is presented in Figure 24. The local embedded board sends a PWM signal through a digital pin to the ESC circuit, which, in turn, generates a three-phase power signal to the brushless motor, adjusting its voltage and

¹ Note that Figure 22 differs from Figure 1 in the type of battery, load profile and in the removal of the main grid.

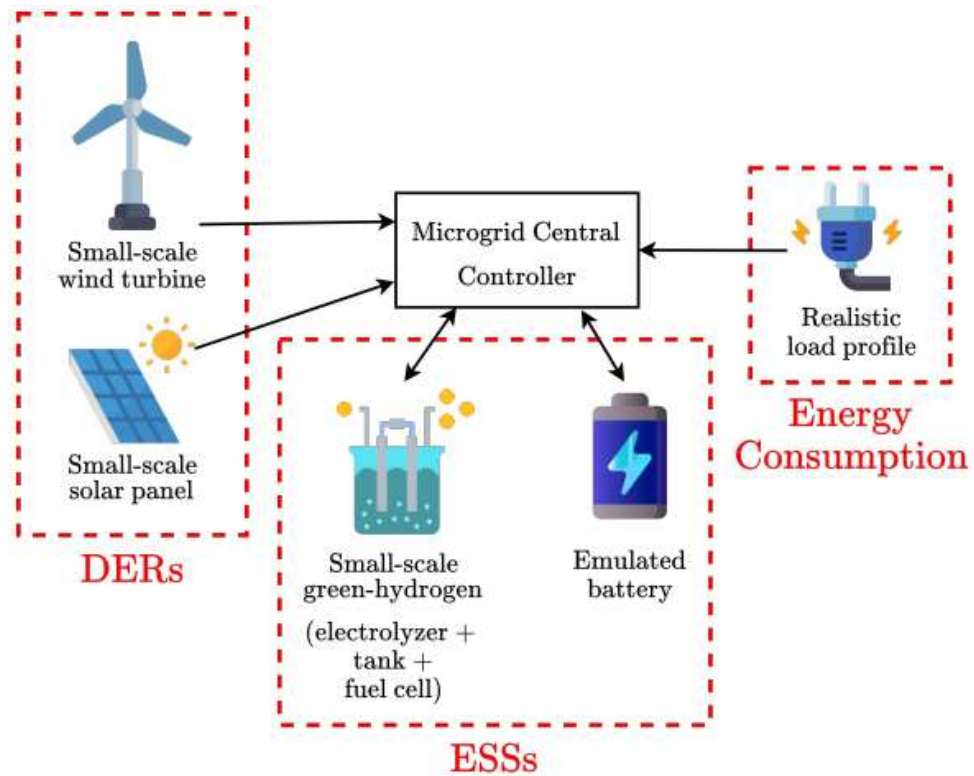


Figure 22 – Small-scale didactic microgrid schematic.

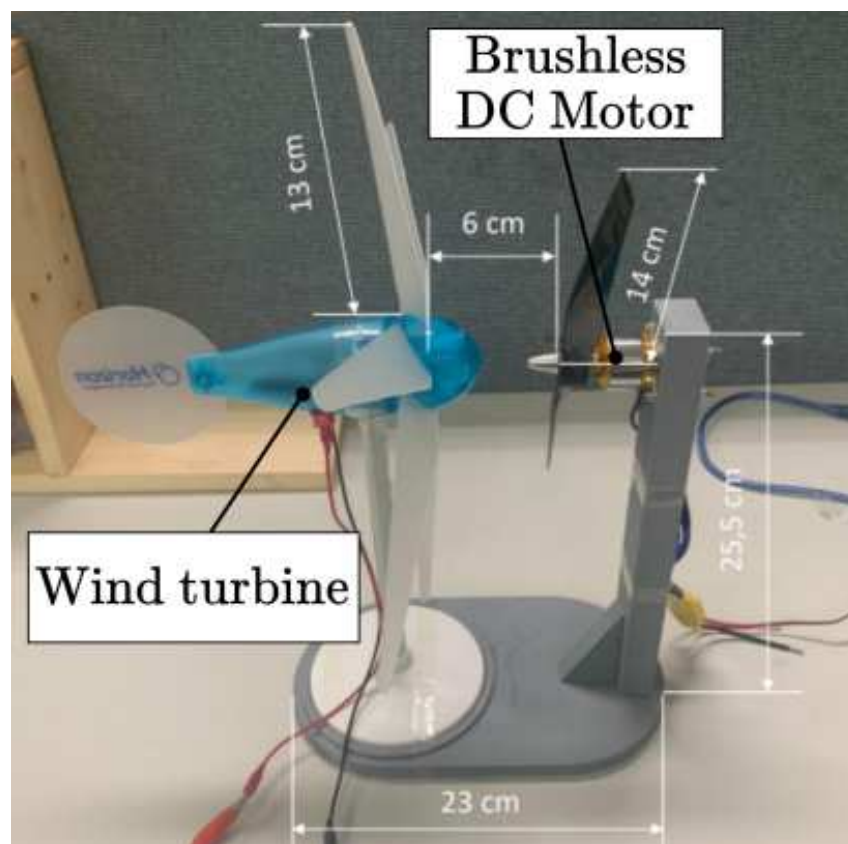


Figure 23 – Physical structure of the small-scale HAWT.

[illegible]

of the input voltage, i.e. $v(t) = f_{wt}(u_{wt}(t))$. The generated wind power $P_{wt}(t)$ is computed, likewise, as a function of the outlet voltage $P_{wt}(t) = f_{P_{wt}}(y_{wt}(t))$. Moreover, to represent the generator's output voltage $y_{wt}(t)$ due to the input voltage that drives the fan $u_{wt}(t)$, we consider local models. In particular, we performed several experimental tests to match local responses using $u_{wt}(t) \in [0, 5]$ V with 0.5 V steps.

$$\frac{dy_{wt}(t)}{dt} = -\overbrace{\left(\frac{1}{\tau(\rho(t))}\right)}^{a(\rho(t))} y_{wt}(t) + \overbrace{\left(\frac{K_e(\rho(t))}{\tau(\rho(t))}\right)}^{b(\rho(t))} u_{wt}(t), \quad (24)$$

Using a standard Least-Squares procedure, the affine terms were identified. In Fig 25, the experimental data is compared to the simulated identified model, for which a fit accuracy of 93.10% is achieved (Normalised Root Mean Square Error-NRMSE metric).

Once the wind turbine model is identified, a local PI LPV- H_∞ control scheme is used to determine the control action $u_{wt}(t)$ such that the generated output $y_{wt}(t)$ tracks realistic

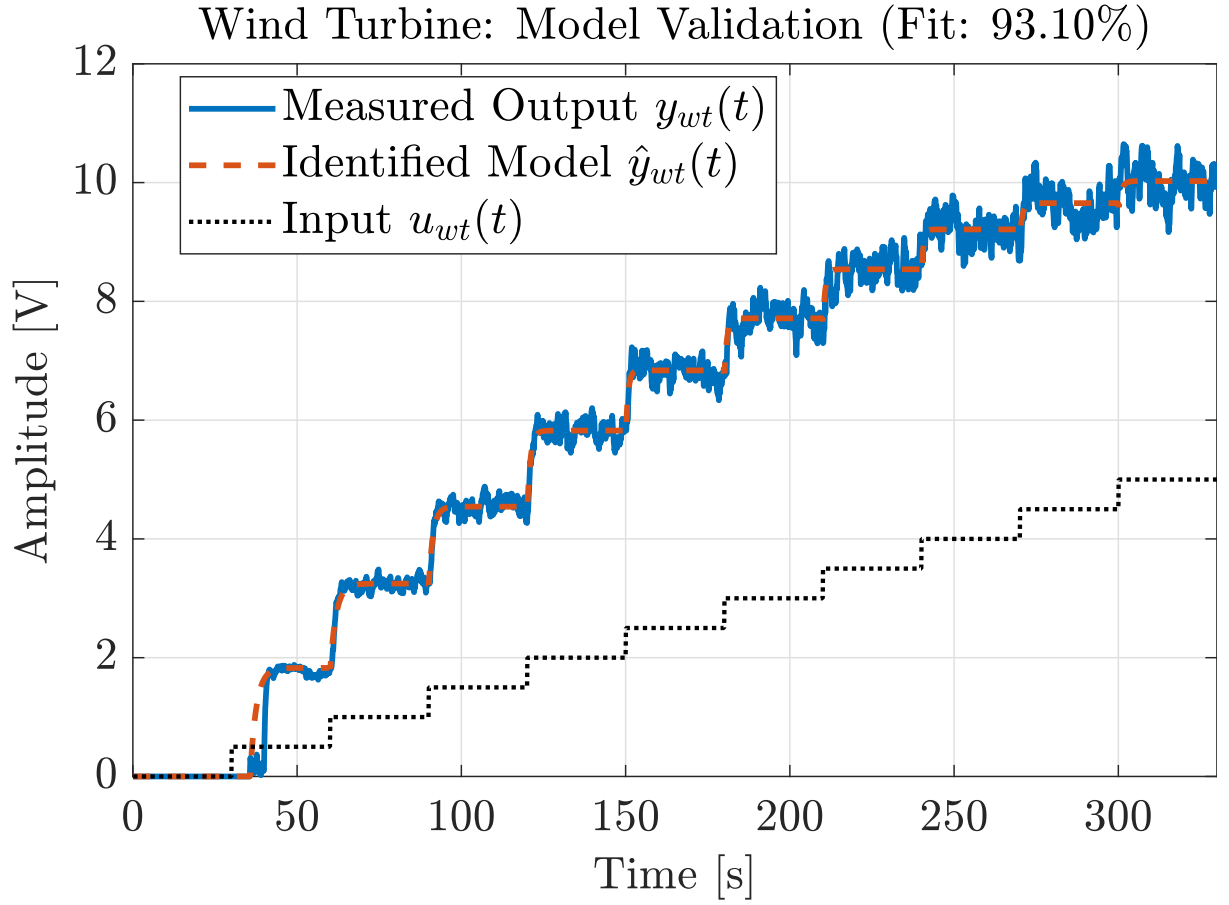


Figure 25 – Wind turbine: Identification and validation.

emulated wind energy profiles. This local controller is outside the scope of this dissertation, but it can be consulted in the work of Silva *et al.* (2025).

4.1.2 Solar panel

The second renewable DER in our setup comprises a single small PV cell and a corresponding controllable LED. The PV panel converts the luminous energy into electrical energy due to the photovoltaic effect (Mueller; Morato; Normey-Rico, 2025). In particular, controlled illumination conditions are provided by a high-power 3W (7000 K) LED, driven by a dedicated Arduino UNO board (via PWM) to emulate real solar profiles.

In the developed PV kit, to ensure standardised conditions of light incidence, the LED is positioned perpendicularly to the PV cell at a fixed distance. Moreover, the entire system is housed with an enclosure that isolates it from ambient light. The structure, presented in Figure 26, was developed using additive manufacturing with a laser-cut acrylic enclosure.

Following similar methodology as with the HAWT setup, we also abstract the solar incidence $i(t)$ as a function of the PWM signal $u_{pv}(t)$, i.e. $i(t) = f_{pv}(u_{pv}(t))$. Accordingly, the electrical power generated from the photovoltaic effect P_{pv} is modelled as a static function to the PV panel output voltage $y_{pv}(t)$, that is $P_{pv} = f_{P_{pv}}(y_{pv}(t))$. This output voltage is acquired by the embedded board using an ADC pin; a schematic representation is presented in Figure 27.

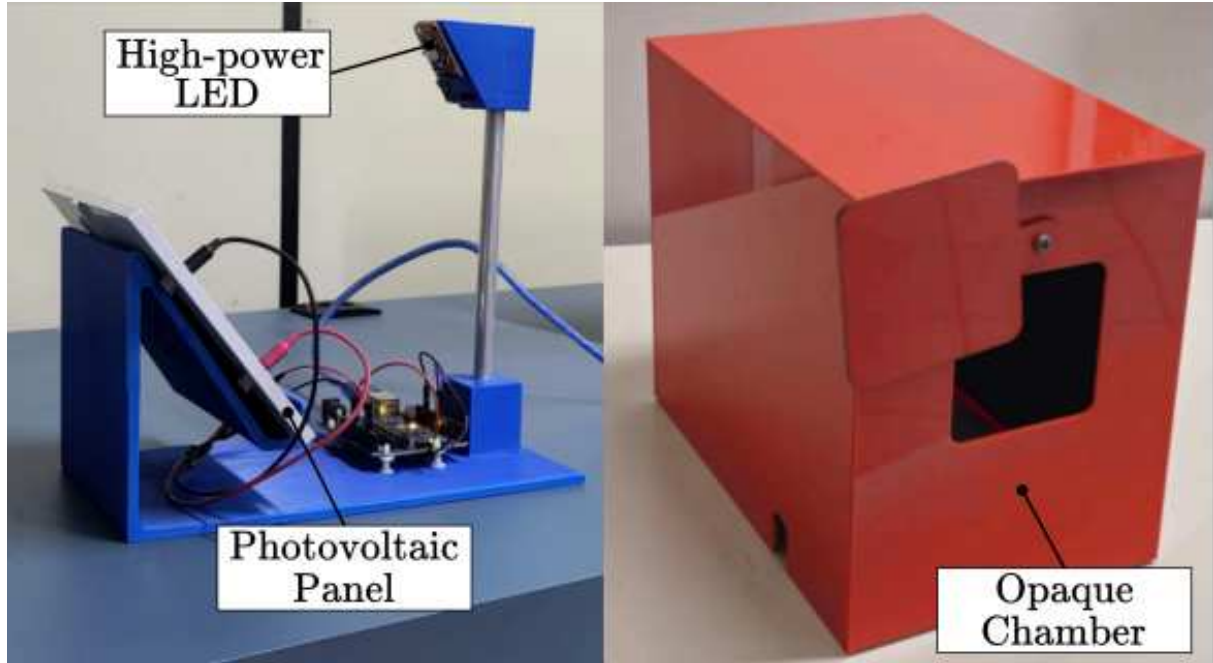


Figure 26 – Physical structure of the scaled PV system.

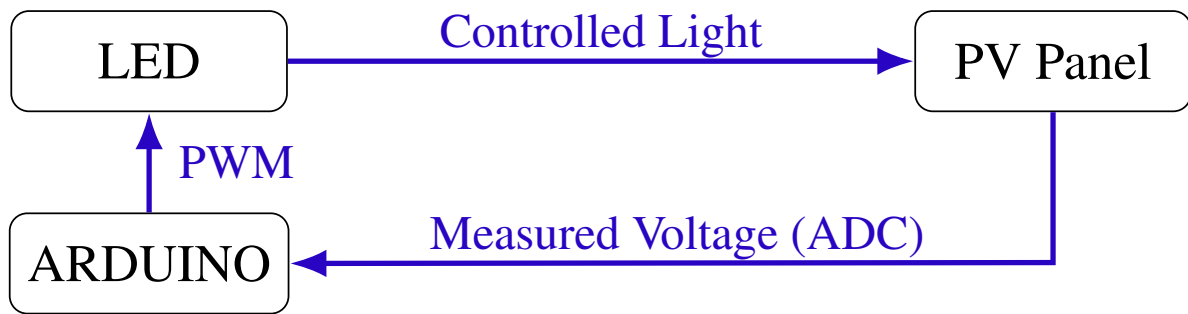


Figure 27 – Schematic of PV panel connections.

To model the output dynamics $y_{pv}(t)$ due to the PWM input $u_{pv}(t)$, we consider, again, local first order representations, which are experimentally identified using Least-Squares and interpolated using an affine LPV embedding with the same format as Eq. (24). The experimental data is compared with the identified model in Figure 28, which yields a 93.15% NRMSE fit accuracy.

In terms of the local control strategy implemented in the embedded board, we stress that a PI law was developed based on the identified LPV model such that desired solar power profiles are tracked, cf. (Mueller; Morato; Normey-Rico, 2025). Notice that, just as with the HAWT system, the local boards enable to test the MGCC operation by emulating different realistic meteorological conditions.

4.1.3 Green hydrogen branch

Complementing the DERs, the developed didactic microgrid features a scaled green hydrogen ESS, which comprises an electrolyser, a tank, and a fuel cell. These components,

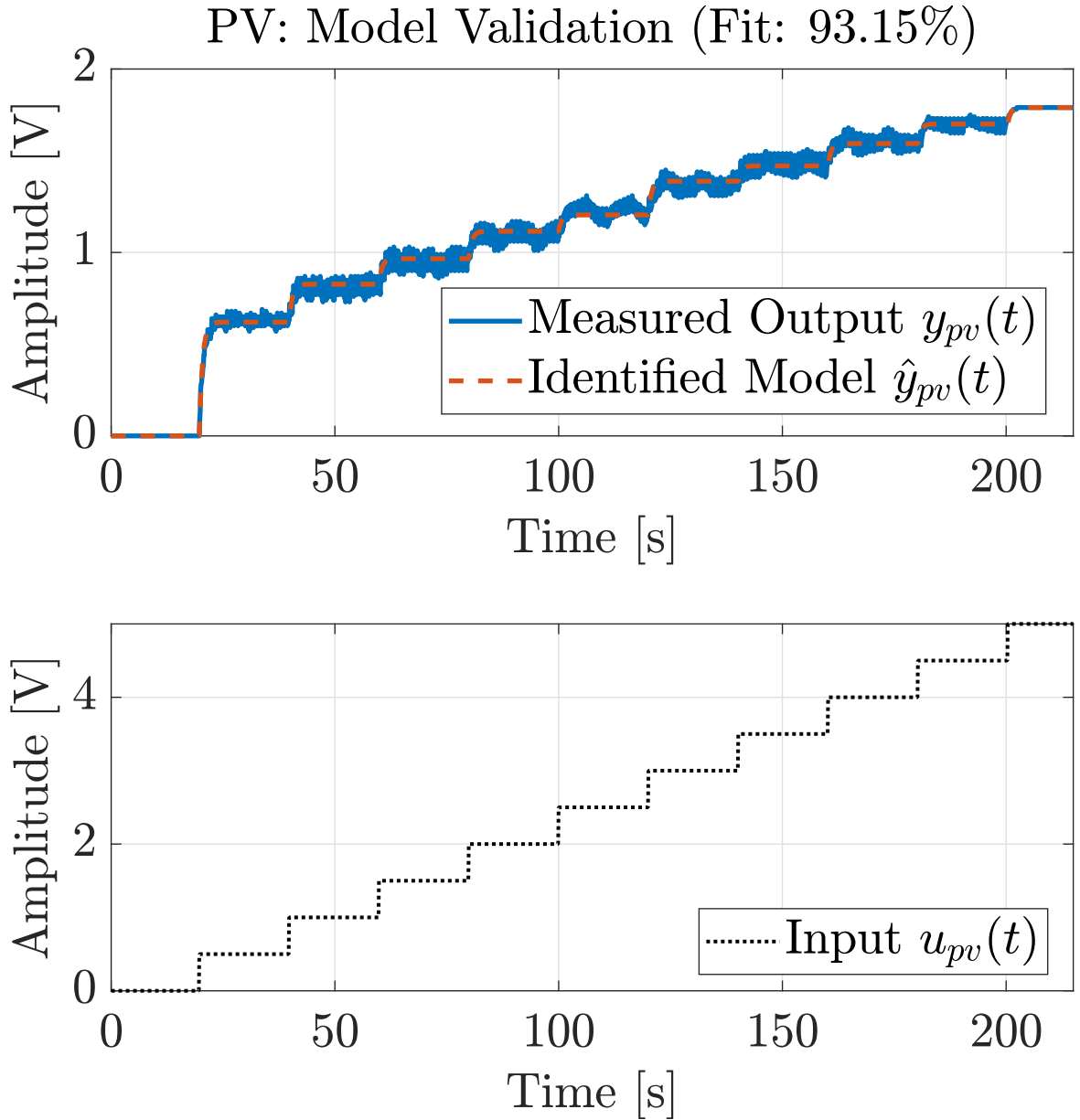


Figure 28 – Solar panel: Identification and validation.

made by Horizon Educational, are off-the-shelf kits used for STEM (Science, Technology, Engineering, and Mathematics) educational purposes. In particular, our setup (presented in Figure 29) also considered an embedded board, a generic load and a servomotor that regulates a hydrogen flow valve, with an instrumentation layer completely dedicated to the hydrogen branch.

The electrolyser functions as an electrochemical transducer that breaks water particles via electrolysis. To this end, a direct current is applied across its electrodes, forcing the non-spontaneous dissociation of water molecules into their constituent gases: hydrogen (H_2) at the cathode and oxygen (O_2) at the anode. This process effectively converts electrical energy into chemical potential energy stored within the hydrogen.

In our experimental microgrid, the power supply for the electrolyser is represented by an

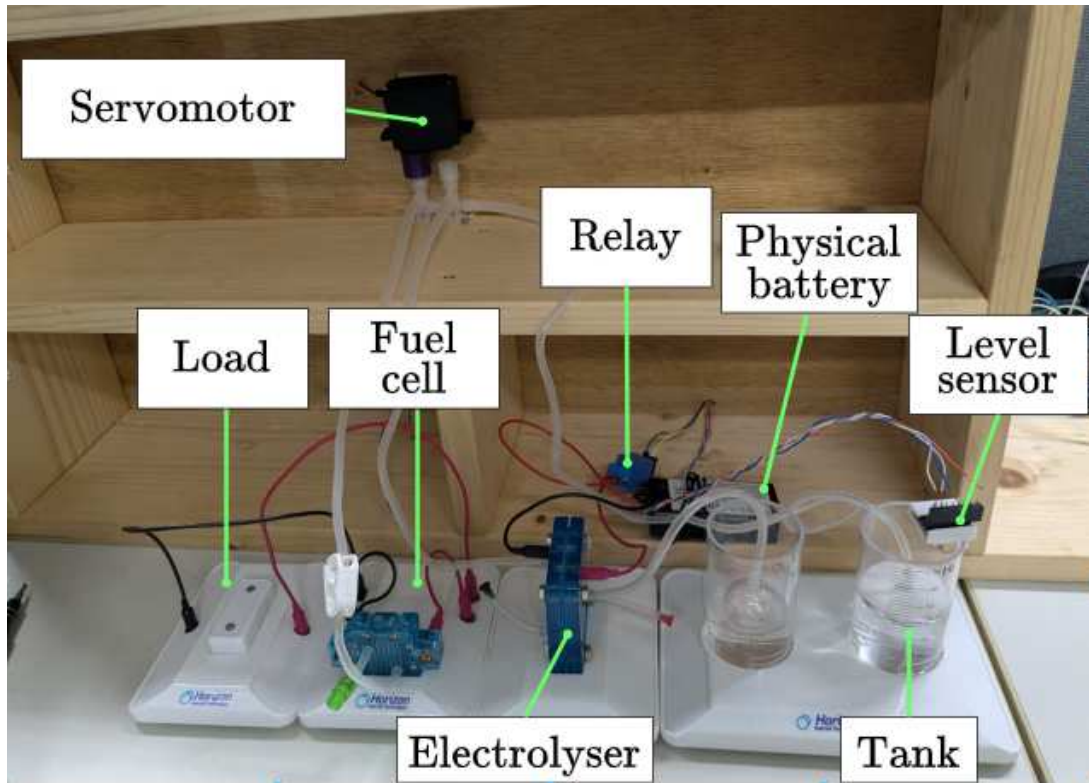


Figure 29 – Setup of the green hydrogen branch.

emulated battery (Sec. 4.1.4). However, in the physical layer, the electrolyser is actually powered by a physical battery pack, which is switched on or off using a local relay. To ensure consistent experimental conditions, this physical battery is kept fully charged. During the operation, the MGCC estimates (based in an identified model) the actual discharge of the physical battery and updates the state of the emulated battery accordingly, ensuring a realistic emulation of the real energy balance in the MG.

The generated gases are stored in two separate tanks: the hydrogen and oxygen tanks (positioned at the right and left side, respectively, in Figure 29). The energy stored in the H_2 gas chamber can be used by the MGCC to dispatch energy, which is done by turning the fuel cell on or off. The latter acts as the inverse mechanism of the electrolyser, converting the chemical potential energy into electricity. During this process, hydrogen (H_2) is supplied to the anode, where it is split into protons and electrons. A membrane allows only protons to pass through to the cathode, forcing the electrons to travel through the external circuit, powering the connected loads. The fuel cell is regulated by a servomotor (controlled by the local embedded board), which either allows or blocks the consumption of gas according to the MGCC decision.

To monitor the H_2 storage level (an important state variable for the MGCC, as discussed later), our setup employs a water displacement method within a reservoir capped at approximately 17 mL. The hydrogen is collected in an inverted chamber submerged within a graduated reservoir; as gas accumulates, the water level rises proportionally. To measure such displacement, a VL6180X Time-of-Flight (ToF) distance sensor is mounted above the reservoir, targeting a 3D-printed floating disk to ensure reflectivity. Although the sensor has a native resolution of

1 mm (which corresponds to approximately 1 mL in our setup), the microcontroller applies a trimmed average filter over 35 samples per cycle. This oversampling technique mitigates sensor noise and enhances the effective resolution to 0.5 mL, providing precise state feedback for the MGCC.

Since the developed MG kit has no physical power bus (or inverters), the produced and consumed energy relations are estimated. In particular, the energy that is consumed to produce H_2 is estimated from the relay switch signal $u_{el}(t)$, determined by the local board. The power generated by the fuel cell, $P_{H_2}(t)$, is computed based on the voltage measured over a constant resistive load R . The output voltage, $y_{H_2}(t)$, is measured by the Arduino UNO using an analogue input pin. Hence, this relationship is modelled as a memoryless function, defined by the electrical law, i.e. $P_{H_2}(t) = \frac{y_{H_2}^2(t)}{R}$. Lastly, the hydrogen level x_{H_2} is determined as a function of both the electrolyser and the fuel cell switch. Due to the inherent mass balance, it is implied that

$$x_{H_2}(k+1) = \delta_{H_2}x_{H_2}(k) + \epsilon_{el}K_{el}u_{el}(k) - \epsilon_{fc}K_{fc}u_{fc}(k), \quad (25)$$

where δ_{H_2} is a loss coefficient related to the hydrogen storage, ϵ_{el} and ϵ_{fc} are conversion coefficients for the electrolyser and fuel cell processes, respectively, while K_{el} is the flow rate gain for the hydrogen level rising and K_{fc} for the level going down. These coefficient and gains were identified and validated using the experimental data and standard Least-Squares, as shown in the results presented in Figure 30. We obtained a linear model with mean fit accuracy of 79.63% (NRMSE metric).

With regard to the mass-balance model in (25) and the instrumentation of our H_2 branch, we stress that hydrogen generation and consumption are implied to be mutually exclusive. Therefore, u_{el} and u_{fc} cannot be active simultaneously (i.e., if $u_{el} = 1$, then $u_{fc} = 0$, and if $u_{fc} = 1$, then $u_{el} = 0$).

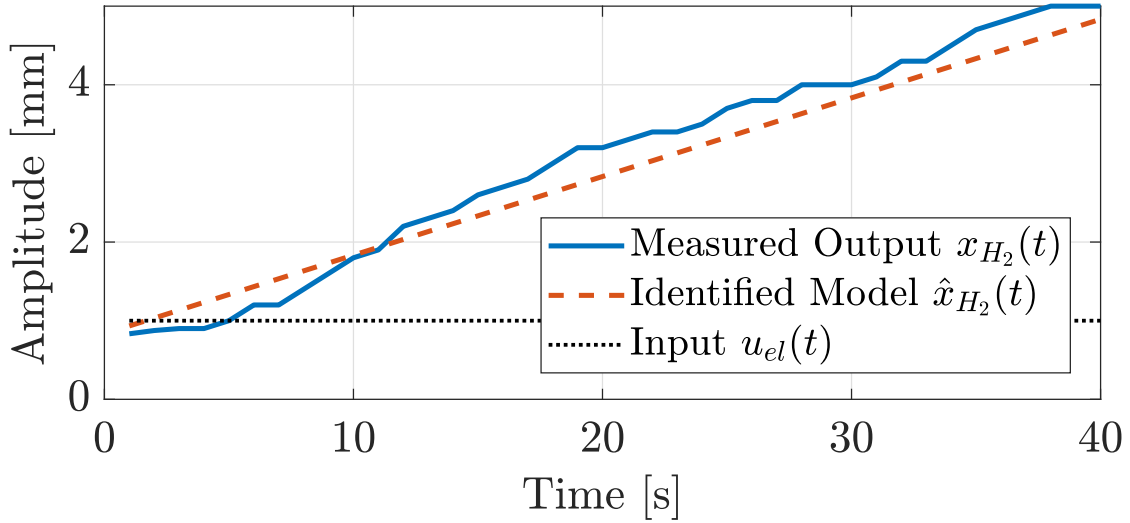
4.1.4 MGCC and emulated battery

Considering that measuring the state of charge (SoC) of real battery systems is inherently difficult (and may yield unreliable measurements), our scaled microgrid also comprises a battery ESS which is emulated rather than physically implemented. To this end, we use a discrete-time model that approximates well the dynamics observed in real batteries, and then the MGCC matches the involved power variations with the other sources: the inlet power from the fuel cell, HAWT and PV, and the power drained by the electrolyser. The battery SoC is emulated, w.r.t:

$$x_{SoC}(k+1) = (1 - \alpha)x_{SoC}(k) + \eta_c P_{\text{charge}}(k) - \frac{1}{\eta_d} P_{\text{discharge}}(k), \quad (26)$$

where $x_{SoC}(k) \in [0,1]$ denotes the normalised SoC of the battery at a given discrete-time sample k , $\alpha \in [0,1]$ represents the self-discharge factor, $P_{\text{charge}} \in (0,400]W$ and $P_{\text{discharge}} \in (0,400]W$ correspond to the charge and discharge power, respectively, which are determined by the MGCC, and $\eta_c \in (0,1]$ and $\eta_d \in (0,1]$ denote the charge and discharge efficiencies, respectively.

H2 gen. using the electrolyser: Model Val. (Fit: 78.99%)



H2 consum. using the fuel cell: Model Val. (Fit: 80.27%)

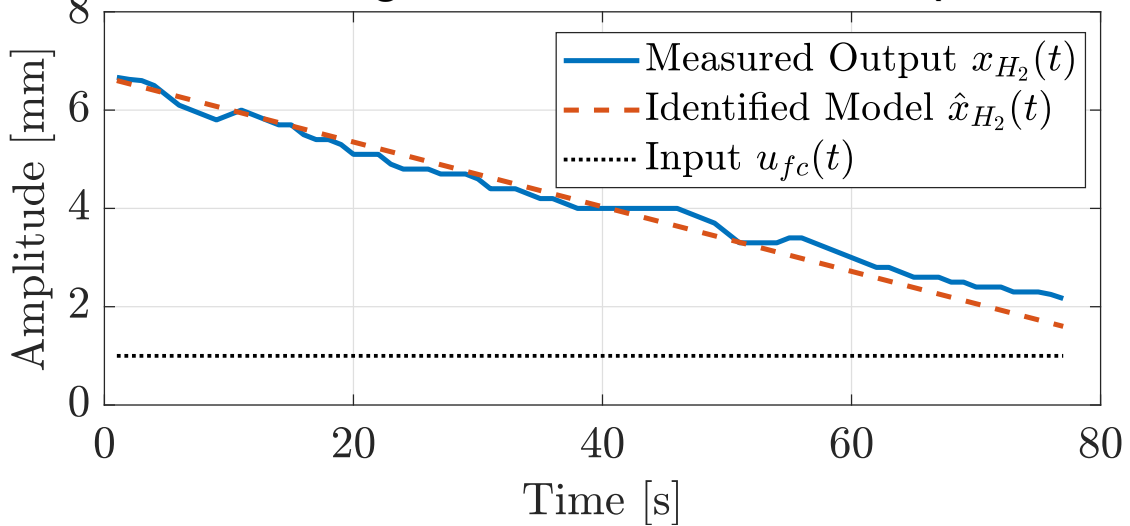


Figure 30 – Green H_2 branch: identification and validation.

4.1.5 Involved communication protocols

Since our scaled microgrid has several embedded boards, related to each local subsystem, a dedicated communication architecture was developed to ensure a coherent operation. In particular, a “central” Arduino board is responsible for acquiring all power measurements from the individual modules (HAWT, PV, electrolyser and fuel cell) and transmitting them to the MGCC, which runs on a computer environment (for prototyping and testing purposes). Based on these measurements, the MGCC computes the optimal control decisions (e.g.: charge or discharge the battery and turn on or off the electrolyser or the fuel cell), at each sampling instant, and sends them back to the central board, which, then, distributes the appropriate signals to all modules, as well as the local reference signals (wind and solar profiles). Finally, the load profile (demand) is emulated and handled in the MGCC (running on a computer).

Two communication protocols are employed in our architecture: a serial universal asynchronous receiver-transmitter (UART) scheme and an inter-integrated circuit (I2C) bus, as illus-

trated in Figure 31. The connection between the central Arduino and the computer is enabled using Serial UART over a USB connection, which is the standard interface for this type of setup (Arduino $\longleftrightarrow$ computer) due to its reliability, simplicity, and the fact that it requires no additional hardware.

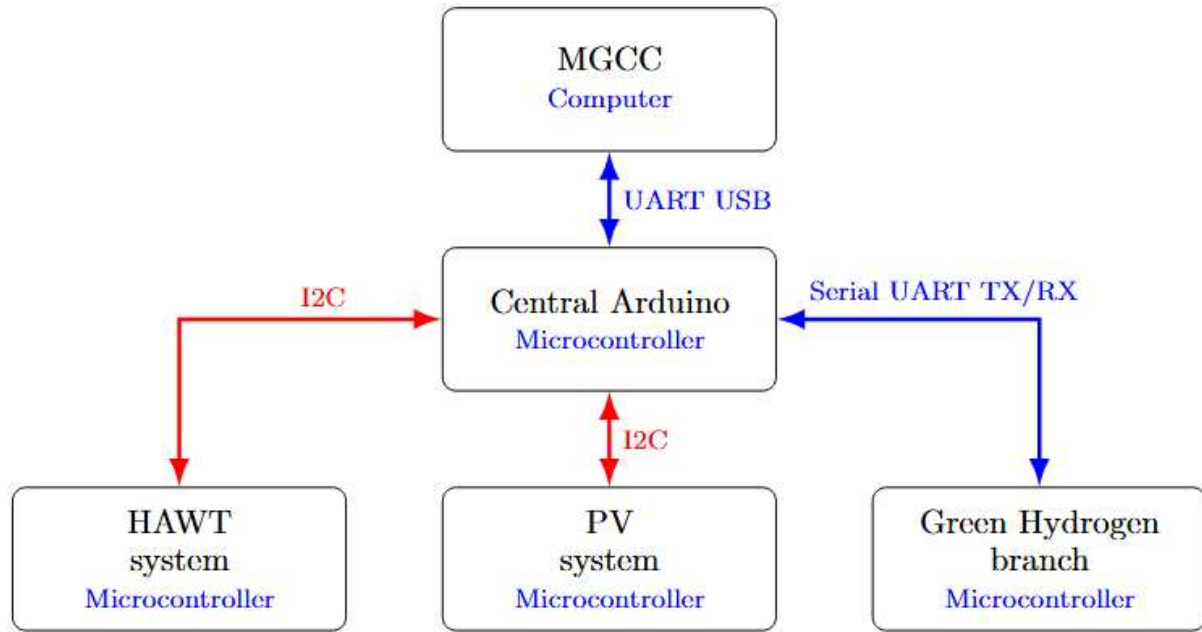


Figure 31 – Communication architecture.

Regarding the connection between the central Arduino and the local board that regulate the wind turbine and PV modules, we use I2C communication since it allows multiple slave devices to share the same two-wire bus with reliable synchronous data transfer. However, the communication between the central board and the green H_2 module (with electrolyser and fuel cell) cannot be done via I2C, since there can only be one master node in the bus (and the hydrogen board has already the master role for the tank level measurements). Therefore, we also employ a UART communication to interface the central board with the hydrogen setup. This solution is possible because the central board used in our microgrid (an Arduino UNO R4 WiFi) has three independent Serial ports, enabling point-to-point UART links which do not interfere with the I2C bus.

Furthermore, note in Figure 31 that there is no communication link between the central Arduino and the emulated battery module. That is because we have implemented the battery model within the central Arduino itself. This design choice follows from the fact that the battery is represented just by a discrete-time model, which does not justify the use of a dedicated microcontroller. Therefore, we chose to implement the emulated battery directly in the central Arduino.

4.2 MICROGRID MODELLING

The developed low-cost small-scale renewable microgrid is presented in Figure 32. This physical structure was developed in the INCT-CAPE laboratory at UFSC, where it is currently located.

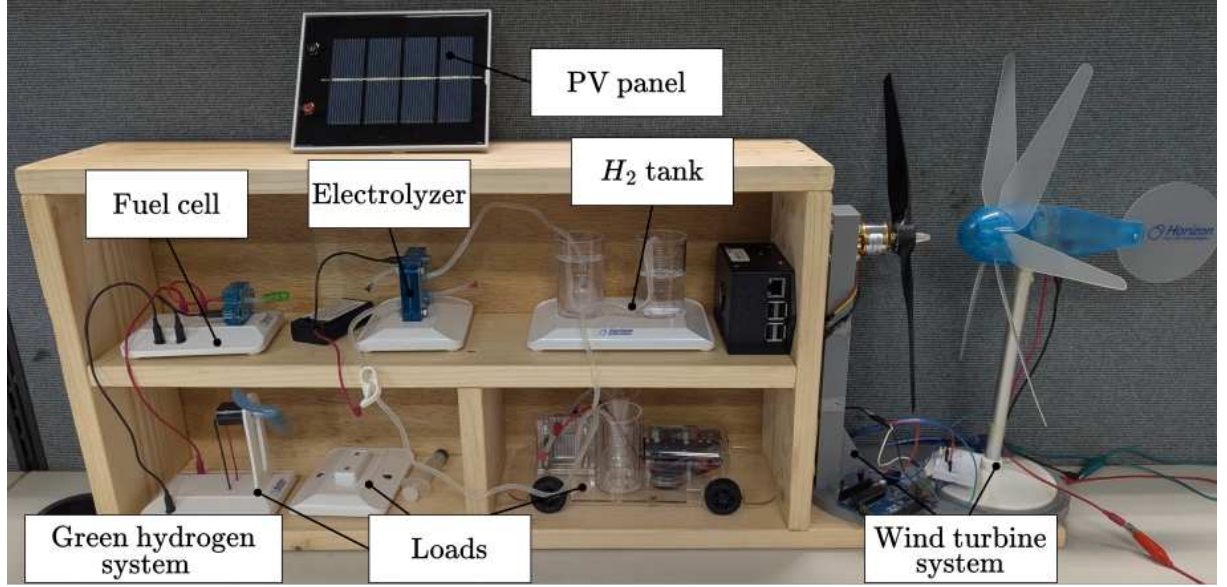


Figure 32 – Physical structure of the small-scale microgrid.

Since all components were properly built, instrumented, its models were identified and the local controllers are validated, we can join all systems dynamics into a MGCC-oriented model involving all variables. To develop this model, we must emphasize the closed-loop objectives:

- The most important aspect is to guarantee that the generated power $P_{\text{total}}(k)$ satisfies the demand $P_{\text{demand}}(k)$, i.e. the microgrid is self-sustainable and can meet demand. To this end, it is important to optimally manage the storage systems to store and dispatch energy whenever it is best.
- Meanwhile, each storage system must be regulated such that the corresponding states are quantitatively near to some determined operational point. For instance, the battery state of charge and the H_2 level must satisfy linear constraint rules, e.g. $-0.1 \leq (x_{\text{SoC}}(k) - 0.5) \leq 0.1$ and $-0.2 \leq (x_{H_2}(k) - 0.6) \leq 0.2$.

The hydrogen level can be obtained by Eq. (25). Through a Least-Squares procedure, we identified the loss coefficient $\delta_{H_2} = 1$, since there is no significant leak in the hydrogen tank for short periods of time. The rising flow rate was identified (Figure 30) as $K_{el} = 130 \times 10^{-9} m^3/s$ and the lowering flow rate is $K_{fc} = 82.63 m^3/s$. Moreover, to simplify the procedure, we identified directly the terms $\epsilon_{el} K_{el} = 0.1 \text{ mm/s}$ and $\epsilon_{fc} K_{fc} = 0.0658 \text{ mm/s}$.

The battery state of charge (SoC) dynamic is presented in Eq. (26). Differently to the hydrogen level, this process is emulated, and the coefficients are, then, defined to be realistic

based in a real battery. In particular, we considered a 69.44 Wh capacity with coefficients: $\eta_c = 3.92 \times 10^{-5}\%/W$ and $\frac{1}{\eta_d} = 4.08 \times 10^{-5}\%/W$; the self-discharge factor is $\alpha = 0.005$.

Finally, for the total MG power, we take into account the sum of all energy sources: (1) the renewable ones, which generate energy to the microgrid; (2) the battery charge, which consumes the microgrid energy, and discharge power, which produces energy from the battery storage; and (3) the electrolyser, which consumes energy to produce H_2 , and the fuel cell power, which produces energy from the H_2 consumption. Hence, the total generated dynamic power by our microgrid is given by the following memoryless equation:

$$\begin{aligned} P_{\text{total}}(k) = & P_{wt}(k) + P_{pv}(k) + \\ & K_{p-fc}u_{fc}(k) - K_{p-el}u_{el}(k) + \\ & P_{\text{discharge}}(k) - P_{\text{charge}}(k), \end{aligned} \quad (27)$$

where K_{p-fc} and K_{p-el} are the gains that determine the fuel cell generation power and electrolyser consumption, respectively. For obtaining a realistic behaviour, we determined this gains based on a real H_2 plant: $K_{p-fc} = 154.57W$ and $K_{p-el} = 100W$. Similarly, we considered $P_{wt}(k) = 50y_{wt}(k)$ and $P_{pv}(k) = 150y_{pv}(k)$ to properly scale the dimensions of power generation and consumption. It is important to notice that P_{total} represents solely the generated power, without subtracting the demand P_{demand} . The actual control objective of the microgrid is to ensure that $P_{\text{total}}(k) = P_{\text{demand}}(k)$.

Based on all the prior developments, we obtain the state-space model for the complete small-scale microgrid, which can directly be used for control purposes (MGCC). Specifically, the generated power P_{total} is modeled as a state variable to accommodate the requirements of the specific solver employed for control. Then, considering $x(k) = \text{col}(x_{SoC}(k), x_{H_2}(k), P_{\text{total}}(k))$ as state variables and $u(k) = \text{col}(P_{\text{charge}}(k), P_{\text{discharge}}(k), u_{el}(k), u_{fc}(k))$ as the control inputs, we obtain

$$\begin{aligned} x(k+1) = & \underbrace{\begin{bmatrix} 0.995 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_A x(k) \\ & + \underbrace{\begin{bmatrix} 3.92 \times 10^{-5}\rho_1 & -4.08 \times 10^{-5}\frac{1}{\rho_2} & 0 & 0 \\ 0 & 0 & 0.1T_s\rho_3 & -0.0658T_s\frac{1}{\rho_4} \\ -\frac{1}{\rho_5} & \rho_6 & -100\frac{1}{\rho_7} & 154.57\rho_8 \end{bmatrix}}_B u(k) \\ & + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \rho_9 & \rho_{10} \end{bmatrix}}_{B_\omega} \underbrace{\begin{bmatrix} P_{wt}(k) \\ P_{pv}(k) \end{bmatrix}}_{\omega(k)}. \end{aligned} \quad (28)$$

where T_s is the sampling time and the parameters $\rho_i \in [0,1]$, for $i = 1, \dots, 10$, are the scheduling variables related to the time-varying efficiencies (Bernardi *et al.*, 2021; Morato *et al.*, 2022b).

In this scenario, we considered $T_s = 0.1$ s, and the whole experiment takes four minutes to conclude, which represents a day (24 h) in real scale. Consequently, each sampling period in the small-scale microgrid represents 36 s of real-sized counterpart.

We stress that the involved disturbance variables (represented by $\omega(k) = \text{col}(P_{wt}(k), P_{pv}(k))$) in (28) are directly related to the renewable energy carriers. Moreover, the total produced power is included as a state variable for simplicity. Likewise, it could be considered an output variable that is a memoryless function of the control inputs. Notice that the third row of the matrix A is null, which means the third state has no dynamic behaviour.

Ultimately, the involved (tertiary/supervisory) control objective is to ensure that the states track a reference signal $x_{ref}(k)$, which contains the battery SoC target, the hydrogen level target and a required electrical demand. In our setup, this demand is artificially generated considering morning and evening peaks.

4.3 MICROGRID CENTRAL CONTROLLER (MGCC)

The microgrid central controller (MGCC) is developed based on the MGCC-oriented model, expressed in Eq. (28). From this model, we formulated an LPV NMPC based in the development of Chapter 3. The main objective of the MGCC is to meet demand, while maintaining the storage systems in a specific range. This control objective can be expressed in the following equation:

$$\begin{aligned} \min_{\mathbf{U}_k} \quad & \sum_{k=0}^{N_p-1} (\|x(k) - x_{ref}(k)\|_Q^2 + \|u(k)\|_R^2) + \|x(N_p) - x_{ref}(N_p)\|_P^2 \\ \text{s.t.} \quad & x(k+1) = A(\rho)x(k) + B(\rho)u(k) + Bd(k) \\ & x(k) \in \mathcal{X}, \quad u(k) \in \mathcal{U}, \quad \forall k \in \{0, \dots, N_p - 1\} \\ & x(N_p) \in \mathbf{X}_t \end{aligned} \quad (29)$$

Where $\mathcal{X}$ and $\mathcal{U}$ represent the polyhedral constraints for the storage state-of-charge, hydrogen level and power converter limits, respectively, as detailed in Section 4.2. The terminal weight P and terminal set $\mathbf{X}_t$ are designed to ensure closed-loop stability. The Eq. (29), with the microgrid complete model in Eq. (28) are, then, adapted to the format of the optimization problem in Eq. (14). In this case, the state-space matrices A and B are constant, as the efficiency parameters are considered fixed for short time intervals. Consequently, the scheduling variables are set to $\rho_i = 1$, for $i = 1, \dots, 10$.

Nevertheless, the MPC formulation takes into account the operational constraints presented in Section 4.2. The disturbances (renewable energy available) are considered constant during the prediction horizon.

To implement our MGCC, we considered the MPC tuning parameters presented in Table 7. To solve the MPC problem (i.e. determine control law online), we applied our `LPV-PILOT` solver. The optimization problem is formulated within a continuous domain to preserve convexity and ensure solver feasibility and suitability. Therefore, a post-processing rounding strat-

MPC Parameter	Value
N_p (prediction horizon)	15 samples
Q (state weighting)	$\text{diag}(1.0, 1.0, 100.0)$
Q (terminal weighting)	$\text{diag}(1.0, 1.0, 100.0)$
R (control weighting)	$\text{diag}(0.01, 0.01, 0.01, 0.001)$

Table 7 – MPC parameters: small-scale microgrid.

egy is implemented. The continuous control signals computed by the NMPC, denoted here as $U_3(k) \in [0, 1]$ and $U_4(k) \in [0, 1]$, are projected onto the binary set $u_{el}(k) \in \{0, 1\}$ and $u_{fc}(k) \in \{0, 1\}$ through a threshold rule:

$$u_{bool}(k) = \begin{cases} 1 & \text{if } u_{cont}(k) \geq 0.5 \\ 0 & \text{if } u_{cont}(k) < 0.5 \end{cases}, \quad (30)$$

where $u_{bool}(k)$ represents the boolean output (in this case u_{el} and u_{fc}), and $u_{cont}(k)$ the MPC's continuous output (in this case U_3 and U_4).

Although this approach provides a suboptimal solution, it avoids the necessity of solving a Mixed-Integer Nonlinear Programming (MINLP) problem while maintaining satisfactory control performance. However, for future works we intend to explore the MINLP problem for microgrids, based on related works (Vergara-Dietrich *et al.*, 2019; Oliveira Luna *et al.*, 2024).

Finally, to evaluate the proposed control scheme, three experimental scenarios were conducted:

- **Experiment 1:** A numerical simulation using a model that considers continuous inputs $u_{el}, u_{fc} \in [0, 1]$. In this ideal scenario, the MPC model perfectly matches the plant behaviour.
- **Experiment 2:** A numerical simulation using the identified real-world model (detailed in Section 4.2), where the rounding strategy is applied.
- **Experiment 3:** An experimental validation using the physical small-scale microgrid setup at the INCT-CAPE laboratory.

4.3.1 Experiment 1: Results

The experiment 1 results are presented in Figure 33 and 34. Moreover, Table 8 presents the Root Mean Square (RMS) performance metrics for the SoC and hydrogen level tracking error ($\text{RMS}(e_{SoC})$ and $\text{RMS}(e_{H_2})$) and for the total power error ($e_{P_{\text{total}}} = P_{\text{total}} - P_{\text{demand}}$) for the three experiments.

First, it is possible to see that, in this configuration, the total generated power P_{total} tracks accurately the demand P_{demand} during the whole simulation ($\text{RMS}(e_{P_{\text{total}}}) = 14.39 \text{ W}$). Such

Experiment	$\text{RMS}(e_{\text{SoC}})$	$\text{RMS}(e_{\text{H}_2})$	$\text{RMS}(e_{P_{\text{total}}})$
Exp. 1	21.06%	0.03 mm	14.39 W
Exp. 2	20.16%	0.01 mm	18.28 W
Exp. 3	15.36%	0.6825 mm	59.91 W

Table 8 – Performance metrics for the LPV-NMPC controller applied to the MGCC.

precision is possible because the electrolyser/fuel cell input (u_{fc} and u_{el} in Figure 34) assumes continuous values which, in reality, are not possible.

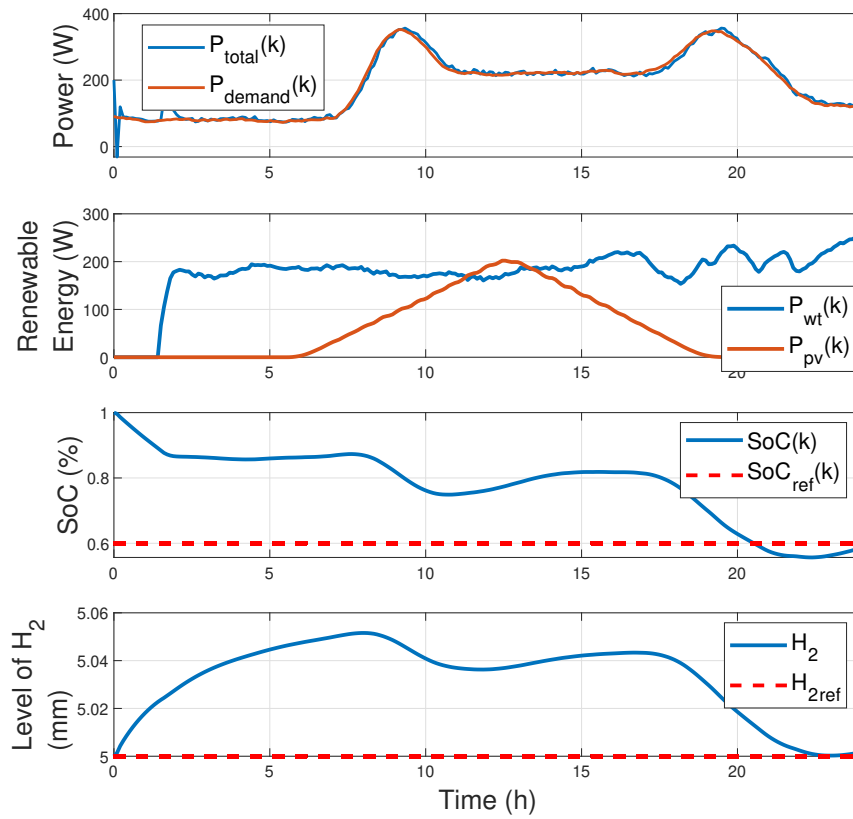


Figure 33 – Experiment 1: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.

We also observe that the SoC and level of H_2 respects the defined range around the reference, which is evaluated by RMS metric presented in Table 8: $\text{RMS}(e_{\text{SoC}}) = 21.06\%$ and $\text{RMS}(e_{\text{H}_2}) = 0.03 \text{ mm}$. Moreover, the mean computing time for this simulation was computed and the obtained result is 9.6 ms, which denotes the high convergence rate of the proposed solver.

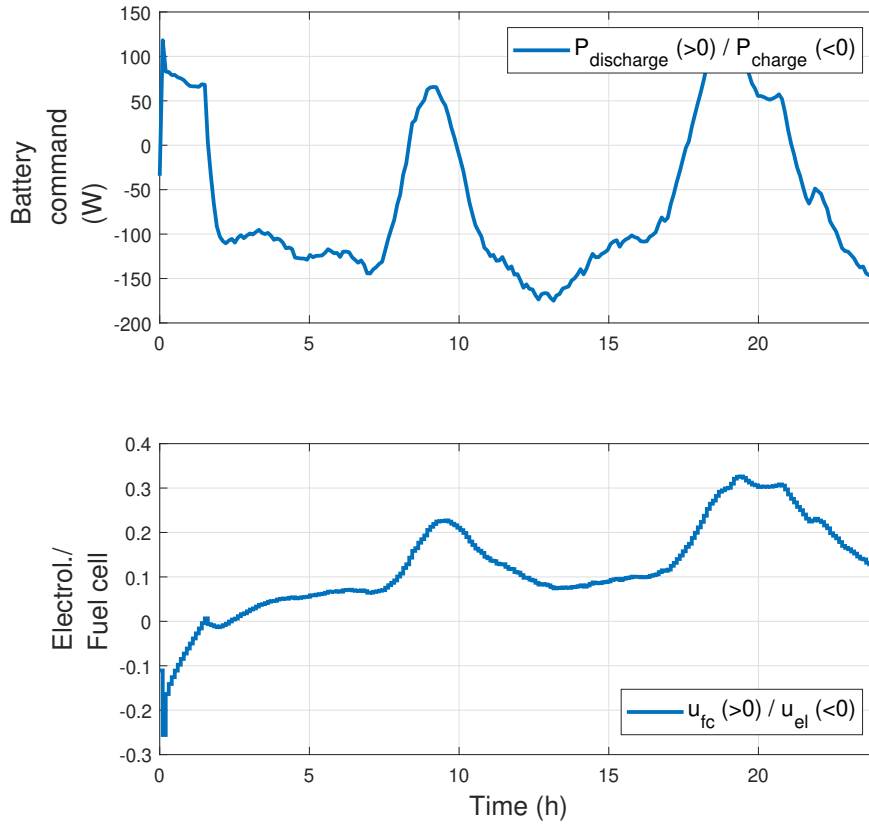


Figure 34 – Experiment 1: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.

4.3.2 Experiment 2: Results

The results of the experiment 2 are presented in Figures 35 and 36. In this scenario the rounding strategy is applied and, then, the electrolyser/fuel cell input is boolean.

Despite the mismatch between the model and the plant behaviour, we observe in Figure 35 that the demand tracking is not significantly affected, as the RMS metric increases only 4 W in the total power tracking. This is because the closed-loop response of the MPC starts to compensate the lack of H_2 usage with the battery. Note that, in the Figure 36, the fuel cell is activated only twice, while the electrolyser is not used because its continuous value never is higher than 0.5.

In this experiment, the mean computing time is 9.8 ms, which is practically equal to the previous result, i.e. there is no difference in the computational burden for solving the optimization problem.

4.3.3 Experiment 3: Results

Lastly, the proposed controller was evaluated in the most representative scenario: the real physical small-scale microgrid located in the INCT-CAPE laboratory (Figure 32). The results are

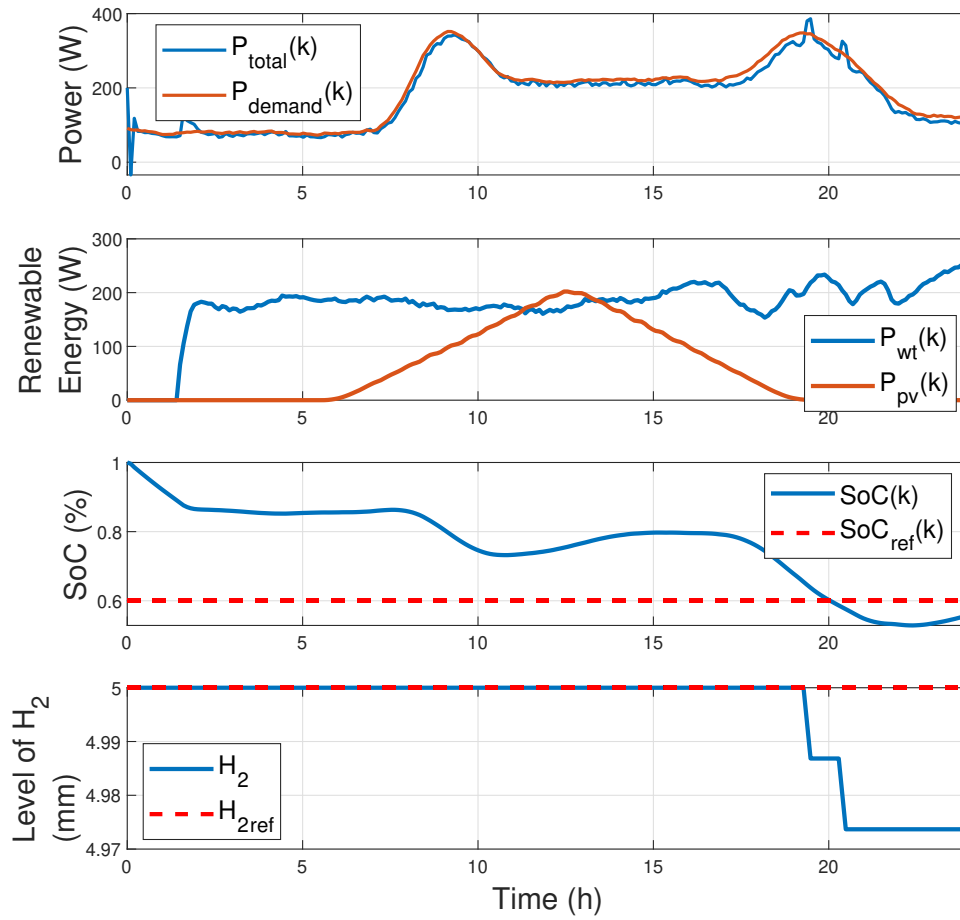


Figure 35 – Experiment 2: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.

presented in Figures 37 and 38. Moreover, a corresponding video² illustrating the microgrid's operation is available online.

This experiment evaluates the proposed solution of this dissertation under real conditions, and these results highlights its suitability for nonlinear embedded systems with strict sampling time.

Based on these results, it is possible to observe how the MGCC optimally manages the energy usage in the microgrid. Firstly, as observed in Figure 37, around 06h00, we observe an increase in the hydrogen level and battery SoC, right before the demand increasing (around 07h00), when the MGCC needs to consume the battery and hydrogen to meet the demand. This ESS consumption is necessary at this time because the renewable source is not enough, since the solar energy is low. Then, when the solar energy reaches its peak, the MGCC decides to charge the battery and turn on the electrolyser, preparing for its usage at the evening demand peak, when the renewable sources are again insufficient.

² <https://youtu.be/Xpw3xwKONrg>

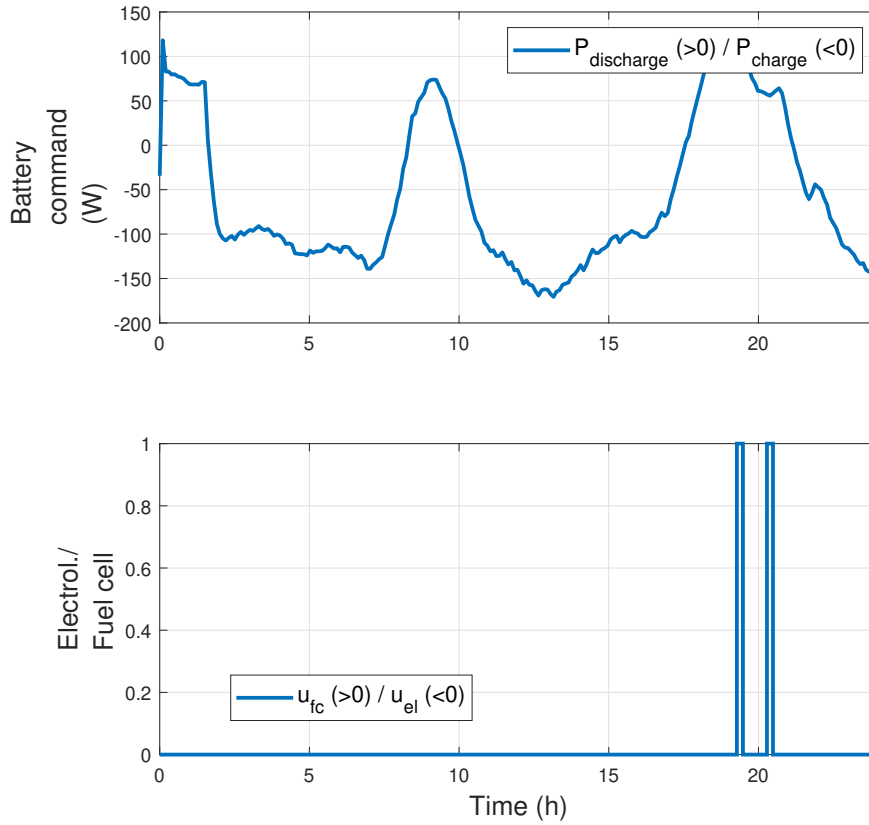


Figure 36 – Experiment 2: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.

Moreover, the mean online computing time was evaluated around 9 and 10 ms, with the maximum execution time remaining below 50 ms. These results are consistent with previous experiments results and suitable for the real time application, since the computing time is lower than the sampling time (100 ms). There was no sampling overrun in this experiment.

However, despite the qualitative analysis, the tracking performance decreased quantitatively. The $\text{RMS}(e_{H_2})$ increased to 0.6825 mm (compared to approximately 0.01 mm and 0.03 mm in previous experiments), and the total power tracking error ($\text{RMS}(e_{P_{\text{total}}})$) increased to 59.91 W. These results may be attributed to unknown uncertainties inherent to the real-world plant. This observation highlights the potential for exploring adaptive control strategies in future research to further enhance robustness. Similarly, Mixed-Integer formulations for NMPC provide a practical framework for handling logical constraints, a topic to be explored in future research. Indeed, the proposed LPV-NMPC algorithm rendered a suboptimal solution for this case, due to the H_2 branch and its associated boolean behaviour, that is adapted by the logical rule in Eq. (30).

In summary, Experiment 3 validates the proposed approach in a realistic grid-connected scenario. The controller successfully managed the power flow, ensuring energy balance and voltage stability while respecting the operational constraints of the hydrogen loop. Most importantly,

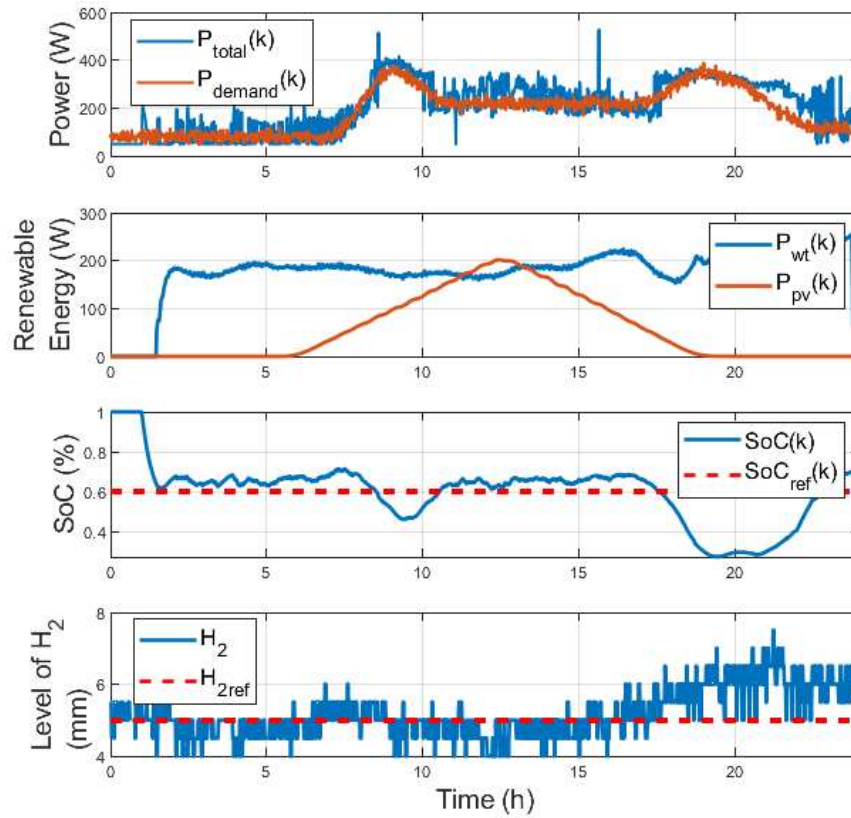


Figure 37 – Experiment 3: Tertiary/supervisory control of the microgrid (MGCC operation): Output variables.

the execution times remained consistently well below the sampling period, corroborating the suitability of the LPV-PILOT solver for embedded, fast NMPC applications in modern power electronics.

4.3.4 Results summary

This chapter presented the development (instrumentation, modelling, identification and control) of a low-cost small-scale microgrid and the validation of LPV-PILOT in this system. Correspondent numerical and experimental validation results were illustrated to evaluate the closed-loop performance. Such results highlight the solver's suitability for nonlinear embedded systems operating under real-time hard constraints. The validation of this algorithm advances the model-based predictive control field by reducing the computational cost for online optimization.

Notwithstanding, we emphasise that the developed microgrid test-bed has a practical interest related to control education and scientific dissemination. Currently, it is already being used in the context of outreach activities related to renewable energy and control, as well as in classroom experiments in system identification and basic control topics.

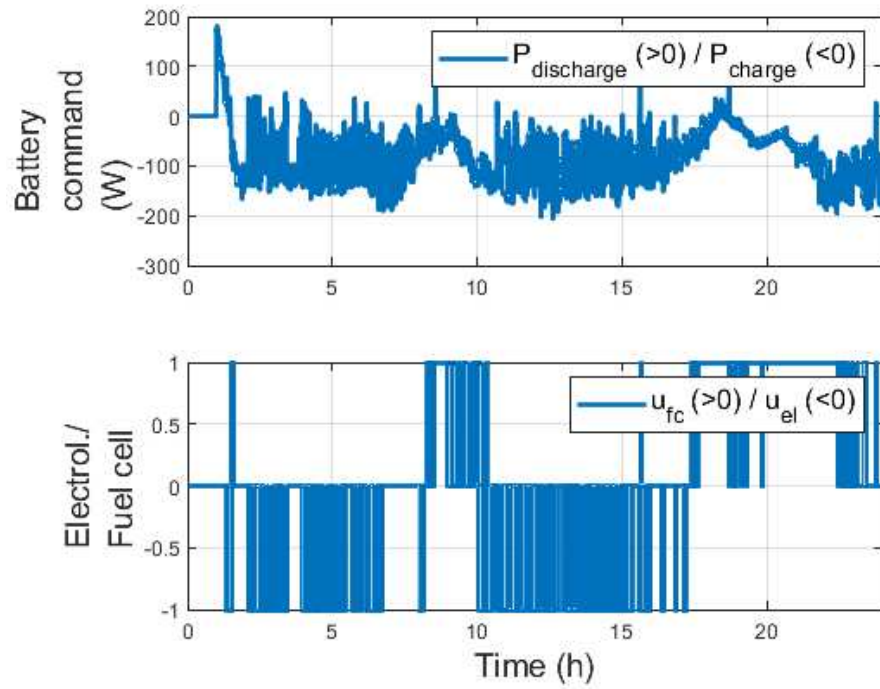


Figure 38 – Experiment 3: Tertiary/supervisory control of the microgrid (MGCC operation): Input variables.

5 CONCLUSION

This chapter concludes the Dissertation's manuscript by summarizing the developed work and pointing the main findings and final remarks.

5.1 SUMMARY OF THE DISSERTATION

This Master's dissertation lies within the context of the energy transition and underlines the importance of microgrids for the optimal energy management and usage of renewable energy. In order to investigate this research topic, in Chapter 1, the context of this work was introduced and the main concepts related to the investigation were set.

Then, in Chapter 2, a thorough literature review was presented, which indicated that MPC indeed is the state-of-the-art controller for microgrids management, the so-called MGCCs. Nevertheless, MPC algorithms exhibits an elevated computational cost, which is highlighted as an important challenge when applied for embedded systems subject to hard real-time constraints. Specifically, the computational time required to solve the involved online optimal problem might be greater than the sampling time of the system, which then leads to infeasibility and may even render the closed loop unstable. Such problem was addressed in the literature review and a gap was identified: the existing tools (algorithms) for embedded systems do not provide a native and integrated end-to-end solution for the NMPC problem.

Aiming to address this research, a new open-source and customised solver for NMPC was developed in Chapter 3, based on recent advances within the LPV framework. This development relies on the hypothesis that qLPV embedding and the log-barrier interior-point method can be combined to produce efficient NMPC algorithms, ensuring both model accuracy and low computational burden. The proposed solver, named as `LPV-PILOT`, is compatible with MATLAB and supports C/C++ implementation for microcontroller applications. By employing established optimization methods and strategies for fast convergence (such as log-barrier, interior-point, and warm start), the solver enhances convergence rates, directly addressing the primary obstacle in applying MPC to real-time systems: the computational burden of online optimization.

Furthermore, the proposed algorithm incorporates specific features to ensure theoretical guarantees and operational robustness:

- Terminal ingredients and constraint tightening mechanisms to preserve closed-loop stability for the class of constrained LPV systems, even in the presence of bounded prediction uncertainties.
- A fallback strategy to handle potential sampling overruns, thereby ensuring real-time feasibility.

Before addressing the main focus of this Dissertation (the microgrids), the solver's performance was (initially) validated by means of four distinct engineering realistic benchmarks:

1. A numerical simulation involving a complex system (TRMS);

2. A HIL simulation to assess the solver's compliance with strict real-time constraints in embedded hardware;
3. A high-fidelity nonlinear pendulum-on-a-cart benchmark, where it is compared against *CasADi*. Results demonstrate the advantages of the proposed solver, particularly regarding its high accuracy and reduced computational time.
4. An experimental validation involving a real physical system: a propeller-driven sailboat at sea. Results pointed to the solver's capability of controlling nonlinear and real-time hard constrained embedded systems.

Furthermore, the main objective of this dissertation was, finally, addressed: a low-cost small-scale microgrid was developed in Chapter 4 and the solver was tested in both simulated (identified model) and real (real plant) conditions. Then, the results were drawn and analysed focusing on the computational burden and accuracy aspects.

5.2 CONCLUDING REMARKS

With the developed solver, several experiments were performed in several nonlinear systems, as previously mentioned. The obtained results underlined the accuracy and low computational burden of the proposed solution. In summary, the solver was tested in both microcontrollers and PCs, and the mean executing time was smaller than 100 ms for all experiments, with the mean time remaining around 10 ms for most applications. These results reinforce the solver's suitability for nonlinear systems and for fast online computation, thanks to the LPV framework and to the optimization techniques applied.

Indeed, the hypotheses drawn in Chapter 1 were verified. That is: the LPV framework (in particular, the qLPV embedding) is adequate for representing system's nonlinearities accurately, without increasing the computational complexity of solving NMPC problems. Moreover, the interior-point method with the logarithmic barrier proved itself as a very suitable method for solving the QP problem originated from the proposed LPV-NMPC. The comparison between our *LPV-PILOT* and *CasADi* showed that this scheme is practically as accurate as nonlinear solvers, but more efficient, i.e. with lower computational time.

Therefore, taking into account the aforementioned exposition, we summarize the main advantages and limitations of the developed method. The primary advantage of the *LPV-PILOT* solver is its computational efficiency, which enables online NMPC execution on embedded hardware. On the other hand, a limitation is that the performance depends on the quality of the LPV model approximation.

This work addressed the boundaries of NMPC theory and extensions to practical engineering settings, which is done by considerably mitigating the computational burden of the involved optimization problems by means of suitable transformations and standard methods. The enable improvements, in practice, have enhanced the suitability of NMPC for real-time embedded systems.

Lastly, regarding the specific application to renewable microgrids, the proposed algorithm has provided adequate performance by optimally managing the energy generation and consumption in the small-scale microgrid, being capable of accurately tracking the demand profile during the whole simulation, as evidenced by the metric of $\text{RMS}(e_{P_{\text{total}}}) = 59.91 \text{ W}$ for the experiment with the real plant. Moreover, the development of this small-scale microgrid and its validation for control purposes plays a fundamental role in education (classroom activities) and outreach actions (scientific disclosure) on control applications for renewable energy systems.

Regarding future works, there are several possibilities to extend this research. First, we intend to compare our scheme with Explicit MPC (Bemporad *et al.*, 2002; Andrade *et al.*, 2020, 2024) and Parametrized NMPC (Alamir, 2012), exploring the aspect of real-time optimization in NMPC algorithms. Another important step is to explore further applications for renewable microgrids, expanding the analysis to a wider range of systems (such as the real-scale microgrid at the INCT-CAPE lab) and algorithms (such as the Mixed-Integer NMPC).

5.3 SCIENTIFIC CONTRIBUTIONS

Derived from the work related to this Master's dissertation, several scientific papers were written and submitted for international journals and conferences, as listed next.

Full papers in conference proceedings:

1. SANCHES, V. M.; MORATO, M. M. ; NORMEY-RICO, J. E. ; SENAME, O. An Interior-Point LPV MPC Solver for Real-time Systems: Experimental Processor-in-the-loop Validation. In: *6th IFAC Workshop on Linear Parameter Varying Systems (LPVS'25)*, Porto, 2025. Proceedings of the 6th IFAC Workshop on Linear Parameter Varying Systems, 2025. v. 1. p. 1-6.
2. SANCHES, V. M.; CUNHA, C. G. ; NORMEY-RICO, J. E. ; MORATO, M. M. Controle Preditivo LPV Parametrizado para Sistemas Embarcados de Tempo Real. In: *XVII Simpósio Brasileiro de Automação Inteligente (SBAI)*, São João del Rei, 2025. Anais do XVII Simpósio Brasileiro de Automação Inteligente, 2025. v. 1.
3. SANCHES, V. M.; BAGGIO, R. W. O. ; COSTA FILHO, M. V. A. ; MORATO, M. M. Controle Avançado em Caldeira de Vapor: Uma Abordagem Estabilizante com MPC LPV. In: *XVII Simpósio Brasileiro de Automação Inteligente (SBAI)*, São João del Rei, 2025. Anais do XVII Simpósio Brasileiro de Automação Inteligente, 2025. v. 1.

Submitted Papers and Preprints

4. SANCHES, V. M.; MORATO, M. M.; BORRELL, A. M.; NORMEY-RICO, J. E.; SENAME, O. An LPV-PILOT in Control: Real-time Nonlinear Predictive control

using interior-point log-barrier Optimisation techniques and the LPV framework. 2025. (Preprint).

5. SANCHES, V. M.; MACHADO, J.; SANCHES, S. R.; MORATO, M. M.; NORMEY-RICO, J. E. Path-Following in USVs using LPV NMPC: Experimental Validation in a Propeller-Driven Boat. 2025. (Preprint).
6. MORATO, M. M.; HOLICKI, T.; SANCHES, V. M.; SCHERER, C. Nonlinear MPC design using the qLPV approach and IQCs-based terminal ingredients. 2025. Preprint available at HAL, ref. hal-04772419v2.
7. SANCHES, V. M.; DA SILVA, S.; DA CUNHA, C. G.; FISCHER, J. D.; MUELLER, G. L.; MORATO, M. M.; NORMEY-RICO, J. E. A Low-Cost Scaled Renewable Microgrid: Modelling, Instrumentation and Validation. 2025. (Preprint).
8. FISCHER, J. D.; SANTANA, G. H. S.; SANCHES, V. M.; MORATO, M. M.; NORMEY-RICO, J. E. Quality of Experience Metrics for the Automatic Tuning of MPCs for Renewable Microgrids. 2025. (Preprint).

REFERENCES

- ABBAS, Hossam S; HANEMA, J; TÓTH, Roland; MOHAMMADPOUR, Javad; MESKIN, Nader. An improved robust model predictive control for linear parameter-varying input-output models. **International Journal of Robust and Nonlinear Control**, Wiley Online Library, v. 28, n. 3, p. 859–880, 2018.
- ABBAS, Hossam S; TÓTH, Roland; PETRECZKY, Mihály; MESKIN, Nader; MOHAMMADPOUR, Javad. Embedding of nonlinear systems in a linear parameter-varying representation. **IFAC Proceedings Volumes**, Elsevier, v. 47, n. 3, p. 6907–6913, 2014.
- ALAMIR, Mazen. A framework for real-time implementation of low-dimensional parameterized NMPC. **Automatica**, Elsevier, v. 48, n. 1, p. 198–204, 2012.
- ALARCÓN, Martín A; ALARCÓN, Rodrigo G; GONZÁLEZ, Alejandro H; FERRAMOSCA, Antonio. Economic model predictive control for energy management of a microgrid connected to the main electrical grid. **Journal of Process Control**, Elsevier, v. 117, p. 40–51, 2022.
- ALCALÁ, Eugenio; PUIG, Vicenç; QUEVEDO, Joseba. LPV-MPC control for autonomous vehicles. **IFAC-Papers OnLine**, v. 52, n. 28, p. 106–113, 2019.
- ALCALÁ, Eugenio; PUIG, Vicenç; QUEVEDO, Joseba; ROSOLIA, Ugo. Autonomous racing using linear parameter varying-model predictive control (LPV-MPC). **Control Engineering Practice**, Elsevier, v. 95, p. 104270, 2020.
- ALLGOWER, Frank; FINDEISEN, Rolf; NAGY, Zoltan. Nonlinear model predictive control: From theory to application. **Journal-Chinese Institute Of Chemical Engineers**, CHINESE INST CHEM ENGINEERS, v. 35, n. 3, p. 299–316, 2004.
- ANDERSSON, Joel AE; GILLIS, Joris; HORN, Greg; RAWLINGS, James B; DIEHL, Moritz. CasADi: a software framework for nonlinear optimization and optimal control. **Mathematical Programming Computation**, v. 11, p. 1–36, 2019.
- ANDRADE, Richard; FERRAMOSCA, Antonio; NORMEY-RICO, Julio Elias; RAFFO, Guilherme V. Explicit model predictive control for a tiltrotor UAV in cargo transportation tasks. In: 1. CONGRESSO Brasileiro de Automática-CBA. [S. l.: s. n.], 2020. v. 2.
- ANDRADE, Richard; FERRAMOSCA, Antonio; NORMEY-RICO, Julio Elias; RAFFO, Guilherme V. Tube-Based Explicit Model Predictive Control for a Tiltrotor UAV in Cargo Transportation Tasks. **Journal of Control, Automation and Electrical Systems**, Springer, v. 35, n. 6, p. 1039–1058, 2024.

BEMPORAD, Alberto; MORARI, Manfred; DUA, Vivek; PISTIKOPOULOS, Efstratios N. The explicit linear quadratic regulator for constrained systems. **Automatica**, Elsevier, v. 38, n. 1, p. 3–20, 2002.

BERNARDI, Emanuel; MORATO, Marcelo Menezes; MENDES, Paulo Renato da Costa; NORMEY-RICO, Julio Elias; ADAM, Eduardo J. Fault-tolerant energy management for an industrial microgrid: A compact optimization method. **International journal of electrical power & energy systems**, Elsevier, v. 124, p. 106342, 2021.

BOMZE, Immanuel M; DEMYANOV, Vladimir F; FLETCHER, Roger; TERLAKY, Tamás; PÓLIK, Imre. Interior point methods for nonlinear optimization. *In*: **NONLINEAR Optimization: Lectures given at the CIME Summer School held in Cetraro, Italy, July 1-7, 2007**. [S. l.: s. n.], 2010. p. 215–276.

BORDONS, Carlos; GARCIA-TORRES, Félix; RIDAO, Miguel A. **Model predictive control of microgrids**. [S. l.]: Springer, 2020. v. 358.

BOYD, Stephen. **Convex optimization**. [S. l.]: Cambridge UP, 2004.

BOYD, Stephen; EL GHAOU, Laurent; FERON, Eric; BALAKRISHNAN, Venkataramanan. **Linear matrix inequalities in system and control theory**. [S. l.]: SIAM, 1994.

CAMACHO, Eduardo F; BORDONS, Carlos. Constrained model predictive control. *In*: **MODEL predictive control**. [S. l.]: Springer, 2007. p. 177–216.

CAMACHO, Eduardo F; BORDONS, Carlos. **Model predictive control**. [S. l.]: Springer Science & Business Media, 2013.

CARLOS, Bárbara Barros; SARTOR, Tommaso; ZANELLI, Andrea; FRISON, Gianluca; BURGARD, Wolfram; DIEHL, Moritz; ORIOLO, Giuseppe. An efficient real-time NMPC for quadrotor position control under communication time-delay. *In*: **IEEE. 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)**. [S. l.: s. n.], 2020. p. 982–989.

CHEN, Steven; SAULNIER, Kelsey; ATANASOV, Nikolay; LEE, Daniel D; KUMAR, Vijay; PAPPAS, George J; MORARI, Manfred. Approximating explicit model predictive control using constrained neural networks. *In*: **IEEE. 2018 Annual American control conference (ACC)**. [S. l.: s. n.], 2018. p. 1520–1527.

CIMINI, G.; BEMPORAD, Alberto; BERNARDINI, D. **ODYS QP Solver**. [S. l.: s. n.], Sept. 2017. ODYS S.r.l. Available from: <https://odys.it/qp>.

CISNEROS, Pablo Sebastian Gonzalez; VOSS, Sophia; WERNER, Herbert. Efficient nonlinear model predictive control via quasi-LPV representation. *In*: **IEEE. 55th CDC**. [S. l.: s. n.], 2016. p. 3216–3221.

CURTIS, Frank E; SCHENK, Olaf; WÄCHTER, Andreas. An interior-point algorithm for large-scale nonlinear optimization with inexact step computations. **SIAM Journal on Scientific Computing**, SIAM, v. 32, n. 6, p. 3447–3475, 2010.

ENGLERT, Tobias; VÖLZ, Andreas; MESMER, Felix; RHEIN, Sönke; GRAICHEN, Knut. A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC). **Optimization and Engineering**, Springer, v. 20, n. 3, p. 769–809, 2019.

FERREAU, Hans Joachim; KIRCHES, Christian; POTSCHKA, Andreas; BOCK, Hans Georg; DIEHL, Moritz. qpOASES: A parametric active-set algorithm for quadratic programming. **Mathematical Programming Computation**, Springer, v. 6, n. 4, p. 327–363, 2014.

FIACCHINI, Mirko; ALAMO, Teodoro; CAMACHO, Eduardo F. On the computation of convex robust control invariant sets for nonlinear systems. **Automatica**, Elsevier, v. 46, n. 8, p. 1334–1338, 2010.

FRISON, Gianluca; DIEHL, Moritz. HPIPM: a high-performance quadratic programming framework for model predictive control. **IFAC-PapersOnLine**, Elsevier, v. 53, n. 2, p. 6563–6569, 2020.

HAN, Renke; TUCCI, Michele; MARTINELLI, Andrea; GUERRERO, Josep M; FERRARI-TRECCATE, Giancarlo. Stability analysis of primary plug-and-play and secondary leader-based controllers for DC microgrid clusters. **IEEE Transactions on Power Systems**, IEEE, v. 34, n. 3, p. 1780–1800, 2018.

HOFFMANN, Christian; WERNER, Herbert. A survey of linear parameter-varying control applications validated by experiments or high-fidelity simulations. **IEEE Transactions on Control Systems Technology**, v. 23, n. 2, p. 416–433, 2014.

HOSE, Henrik; KÖHLER, Johannes; ZEILINGER, Melanie N; TRIMPE, Sebastian. Approximate nonlinear model predictive control with safety-augmented neural networks. **IEEE Transactions on Control Systems Technology**, IEEE, 2025.

HOUSKA, Boris; FERREAU, Hans Joachim; DIEHL, Moritz. ACADO toolkit—An open-source framework for automatic control and dynamic optimization. **Optimal control applications and methods**, Wiley Online Library, v. 32, n. 3, p. 298–312, 2011.

HU, Jiefeng; ZHU, Jianguo; GUERRERO, Josep M. Model predictive control of smart microgrids. In: IEEE. 2014 17th International Conference on Electrical Machines and Systems (ICEMS). [S. l.: s. n.], 2014. p. 2815–2820.

HU, Jiefeng *et al.* Economic model predictive control for microgrid optimization: A review. **IEEE Transactions on Smart Grid**, IEEE, v. 15, n. 1, p. 472–484, 2023.

LAHR, Amon; NÄF, Joshua; WABERSICH, Kim P; FREY, Jonathan; SIEHL, Pascal; CARRON, Andrea; DIEHL, Moritz; ZEILINGER, Melanie N. L4acados: Learning-based models for acados, applied to Gaussian process-based predictive control. **arXiv preprint arXiv:2411.19258**, 2024.

LASSETER, Robert H. Microgrids. *In: IEEE. 2002 IEEE power engineering society winter meeting. Conference proceedings (Cat. No. 02CH37309). [S. l.: s. n.], 2002. v. 1, p. 305–308.*

LASSETER, Robert H; PAIGI, Paolo. Microgrid: a conceptual solution. *In: IEEE. 2004 IEEE 35th annual power electronics specialists conference (IEEE Cat. No. 04CH37551). [S. l.: s. n.], 2004. v. 6, p. 4285–4290.*

LOFBERG, J. YALMIP : a toolbox for modeling and optimization in MATLAB. *In: 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508). [S. l.: s. n.], 2004. p. 284–289.*

LUP, Andrew Ng Kay *et al.* Sustainable energy technologies for the Global South: challenges and solutions toward achieving SDG 7. **Environmental Science: Advances**, v. 2, n. 4, p. 570–585, 2023.

MATTINGLEY, Jacob; BOYD, Stephen. CVXGEN: A code generator for embedded convex optimization. **Optimization and Engineering**, Springer, v. 13, n. 1, p. 1–27, 2012.

MENDES, Paulo Renato da Costa; ISORNA, Luis Valverde; BORDONS, Carlos; NORMEY-RICO, Julio Elias. Energy management of an experimental microgrid coupled to a V2G system. **Journal of Power Sources**, Elsevier, v. 327, p. 702–713, 2016.

MENDES, Paulo Renato da Costa; NORMEY-RICO, Julio Elias; BORDONS, Carlos. Economic energy management of a microgrid including electric vehicles. *In: IEEE. 2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM). [S. l.: s. n.], 2015. p. 869–874.*

MOHAMMADPOUR, Javad; SCHERER, Carsten W. **Control of linear parameter varying systems with applications**. [S. l.]: Springer Science & Business Media, 2012.

MORATO, Marcelo Menezes *et al.* LPV-MPC fault-tolerant energy management strategy for renewable microgrids. **International Journal of Electrical Power & Energy Systems**, v. 117, p. 105644, 2020.

MORATO, Marcelo Menezes; CUNHA, Victor M; SANTOS, Tito L M; NORMEY-RICO, Julio Elias; SENAME, Olivier. A robust nonlinear tracking MPC using qLPV embedding and zonotopic uncertainty propagation. **Journal of the Franklin Institute**, Elsevier, p. 106713, 2024.

MORATO, Marcelo Menezes; HOLICKI, Tobias; SCHERER, Carsten W. Stabilizing model predictive control synthesis using integral quadratic constraints and full-block multipliers. **International Journal of Robust and Nonlinear Control**, v. 33, n. 18, p. 11434–11457, 2023.

MORATO, Marcelo Menezes; JUNGERS, Marc; NORMEY-RICO, Julio Elias; SENAME, Olivier. A predictive fault tolerant control method for qLPV systems subject to input faults and constraints. **Journal of the Franklin Institute**, Elsevier, v. 359, n. 16, p. 9129–9167, 2022.

MORATO, Marcelo Menezes; MARQUEZ, JJ; ZAFRA-CABEZA, Ascensión; BORDONS, Carlos; NORMEY-RICO, Julio Elias. Fault-tolerant energy management in renewable microgrids using LPV MPC. **IFAC-PapersOnLine**, Elsevier, v. 55, n. 35, p. 1–6, 2022.

MORATO, Marcelo Menezes; MENDES, Paulo Renato da Costa; CANI, Alex Amadeu; NORMEY-RICO, Julio Elias; BORDONS, Carlos. Future hybrid local energy generation paradigm for the brazilian sugarcane industry scenario. **International Journal of Electrical Power & Energy Systems**, Elsevier, v. 101, p. 139–150, 2018.

MORATO, Marcelo Menezes; MENDES, Paulo Renato da Costa; NORMEY-RICO, Julio Elias; BORDONS, Carlos. LPV-MPC fault-tolerant energy management strategy for renewable microgrids. **International Journal of Electrical Power & Energy Systems**, Elsevier, v. 117, p. 105644, 2020.

MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias; SENAME, Olivier. Model predictive control design for linear parameter varying systems: A survey. **Annual Reviews in Control**, v. 49, p. 64–80, 2020.

MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias; SENAME, Olivier. Model predictive control design for linear parameter varying systems: A survey. **Annual Reviews in Control**, Elsevier, v. 49, p. 64–80, 2020.

MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias; SENAME, Olivier. Sub-optimal recursively feasible Linear Parameter-Varying predictive algorithm for semi-active suspension control. **IET Control Theory & Applications**, Wiley Online Library, v. 14, n. 18, p. 2764–2775, 2020.

MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias; SENAME, Olivier. Sufficient Conditions for Convergent Recursive Extrapolation of qLPV Scheduling Parameters Along a Prediction Horizon. **IEEE Trans. on Automatic Control**, v. 68, n. 6, p. 3182–3193, 2022.

MORATO, Marcelo Menezes; REGNER, Daniel J; MENDES, Paulo Renato da Costa; NORMEY-RICO, Julio Elias; BORDONS, Carlos. Fault analysis, detection and estimation for a microgrid via H_2/H_∞ LPV observers. **International Journal of Electrical Power & Energy Systems**, Elsevier, v. 105, p. 823–845, 2019.

MORATO, Marcelo Menezes; VERGARA-DIETRICH, José; ESPARCIA JR, Eugene A; OCON, Joey D; NORMEY-RICO, Julio Elias. Assessing demand compliance and reliability in the Philippine off-grid islands with Model Predictive Control microgrid coordination. **Renewable Energy**, Elsevier, v. 179, p. 1271–1290, 2021.

MUELLER, Gabriele L.; MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias. Small-Scale PV Emulator for Experimental Control Validation. Proceedings of the 2025 Solar World Congress. [S. l.], 2025.

NASSOUROU, Mohamadou; PUIG, Vicenç; BLESÁ, Joaquim; OCAMPO-MARTÍNEZ, Carlos. Economic model predictive control for energy dispatch of a smart micro-grid system. In: IEEE. 2017 4th International Conference on Control, Decision and Information Technologies (CoDIT). [S. l.: s. n.], 2017. p. 0944–0949.

NGUYEN, Khai; SCHOEDEL, Sam; ALAVILLI, Anoushka; PLANCHER, Brian; MANCHESTER, Zachary. Tinympc: Model-predictive control on resource-constrained microcontrollers. In: IEEE. 2024 IEEE International Conference on Robotics and Automation (ICRA). [S. l.: s. n.], 2024. p. 1–7.

OLIVARES, Daniel E; MEHRIZI-SANI, Ali; ETEMADI, Amir H; CAÑIZARES, Claudio A; IRAVANI, Reza; KAZERANI, Mehrdad; HAJIMIRAGHA, Amir H; GOMIS-BELLMUNT, Oriol; SAEEDIFARD, Maryam, *et al.* Trends in microgrid control. **IEEE Transactions on smart grid**, IEEE, v. 5, n. 4, p. 1905–1919, 2014.

OLIVEIRA, Lucas AL; GUELTON, Kevin; MOTCHON, Koffi MD; LEITE, Valter JS. Refining parameter mismatch handling in LPV sampled-data controller design. **Automatica**, v. 180, p. 112470, 2025.

OLIVEIRA LUNA, José Diogo Forte de; NASPOLINI, Amir; REIS, Guilherme Nascimento Gouvêa dos; MENDES, Paulo Renato da Costa; NORMEY-RICO, Julio Elias. A novel joint energy and demand management system for smart houses based on model predictive control, hybrid storage system and quality of experience concepts. **Applied Energy**, Elsevier, v. 369, p. 123466, 2024.

PARISIO, Alessandra; RIKOS, Evangelos; GLIELMO, Luigi. A model predictive control approach to microgrid operation optimization. **IEEE Transactions on Control Systems Technology**, IEEE, v. 22, n. 5, p. 1813–1827, 2014.

PECCIN, Vinícius Berndsen; LIMA, Daniel Martins; FLESCH, Rodolfo César Costa; NORMEY-RICO, Julio Elias. Fast algorithms for constrained generalised predictive control with on-line optimisation. **IET Control Theory & Applications**, v. 15, n. 4, p. 545–558, 2021.

PEIXOTO, Márcia Luciana da Costa; OLIVEIRA, Pedro Moreira de; BESSA, Iury; COUTINHO, Pedro Henrique; PESSIM, Paulo Sergio Pereira; PUIG, Vicenç; PALHARES, Reinaldo Martinez. Fault-tolerant dynamic output feedback control of LPV systems via fault hiding. **Automatica**, Elsevier, v. 174, p. 112191, 2025.

- PEREIRA, Mario; LIMÓN, Daniel; ALAMO, Teodoro; VALVERDE, Luis; BORDONS, Carlos. Economic model predictive control of a smartgrid with hydrogen storage and PEM fuel cell. *In: IEEE. IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society*. [S. l.: s. n.], 2013. p. 7920–7925.
- PLUCENIO, Agostinho; PAGANO, Daniel J; BRUCIAPAGLIA, AH; NORMEY-RICO, Julio Elias. A practical approach to predictive control for nonlinear processes. **IFAC Proceedings Volumes**, Elsevier, v. 40, n. 12, p. 210–215, 2007.
- RAFFO, Guilherme V; GOMES, Guilherme K; NORMEY-RICO, Julio Elias; KELBER, Christian R; BECKER, Leandro B. A predictive controller for autonomous vehicle path tracking. **IEEE transactions on intelligent transportation systems**, IEEE, v. 10, n. 1, p. 92–102, 2009.
- RAKOVIC, Saša V; KOUVARITAKIS, Basil; CANNON, Mark; PANOS, Christos; FINDEISEN, Rolf. Parameterized tube model predictive control. **IEEE Transactions on Automatic Control**, IEEE, v. 57, n. 11, p. 2746–2761, 2012.
- REGO, Brenner S; RAFFO, Guilherme V; SCOTT, Joseph K; RAIMONDO, Davide M. Guaranteed methods based on constrained zonotopes for set-valued state estimation of nonlinear discrete-time systems. **Automatica**, Elsevier, v. 111, p. 108614, 2020.
- ROOS, Cornelis; TERLAKY, Tamás; VIAL, J-Ph. **Interior point methods for linear optimization**. [S. l.: s. n.], 2005.
- ROTONDO, Damiano; NEJJARI, Fatiha; PUIG, Vicenç. Quasi-LPV modeling, identification and control of a twin rotor MIMO system. **Control Engineering Practice**, v. 21, n. 6, p. 829–846, 2013.
- SALA, Antonio. Generalising quasi-LPV and CDI models to quasi-convex difference inclusions. **IFAC-PapersOnLine**, Elsevier, v. 50, n. 1, p. 7560–7565, 2017.
- SANCHES, Vinícius Moreno *et al.* Lateral Control Techniques for Autonomous Vehicles. **Repositório Institucional da UFSC**, Toulouse, France, 2024.
- SANCHES, Vinícius Moreno; MACHADO, Juliana; SANCHES, Sergio Roberto; MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias. Path-Following in USVs using LPV NMPC: Experimental Validation in a Propeller-Driven Boat. Paper in preparation. [S. l.], Sept. 2025.
- SANCHES, Vinícius Moreno; MORATO, Marcelo Menezes; BORRELL, Ariel Medero; NORMEY-RICO, Julio Elias; SENAME, Olivier. An LPV-PILOT in Control: Real-time Nonlinear Predictive control using interior-point log-barrier Optimisation techniques and the LPV framework. [Pre-Print, available in <https://hal.science/hal-05281686>], 2025.

SANCHES, Vinícius Moreno; MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias; SENAME, Olivier. An interior-point LPV MPC solver for real-time systems: Experimental Processor-in-the-loop validation. *In: PROCEEDINGS of the 6th IFAC Workshop on Linear Parameter Varying Systems*. [S. l.]: IFAC, 2025. v. 1, p. 1–6.

SANCHES, Vinícius Moreno; SILVA, Samira Liana; CUNHA, Cecilia Gonçalves da; FISCHER, Johannes Daniel; MUELLER, Gabriele L.; MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias. A Low-Cost Scaled Renewable Microgrid: Modelling, Instrumentation and Validation. [S. l.], Dec. 2025.

SANTOS, Tito L M; BONZANINI, Angelo D; HEIRUNG, Tor Aksel N; MESBAH, Ali. A constraint-tightening approach to nonlinear model predictive control with chance constraints for stochastic systems. *In: AMERICAN Control Conference*. [S. l.]: IEEE, 2019. p. 1641–1647.

SCHERER, Carsten W. LPV control and full block multipliers. **Automatica**, Elsevier, v. 37, n. 3, p. 361–375, 2001.

SEN, Sachidananda; KUMAR, Vishal. Microgrid control: A comprehensive survey. **Annual Reviews in control**, Elsevier, v. 45, p. 118–151, 2018.

SEN, Sachidananda; KUMAR, Vishal. Microgrid modelling: A comprehensive survey. **Annual reviews in control**, Elsevier, v. 46, p. 216–250, 2018.

SENAME, Olivier; GASPAR, Peter; BOKOR, József. **Robust control and linear parameter varying approaches: application to vehicle dynamics**. [S. l.]: Springer, 2013. v. 437.

SHAMMA, Jeff S. An overview of LPV systems. **Control of linear parameter varying systems with applications**, Springer, p. 3–26, 2012.

SILVA, Samira Liana; FISCHER, Johannes Daniel; MORATO, Marcelo Menezes; NORMEY-RICO, Julio Elias. Development of a Didatic Wind Turbine: Identification, PI LPV- $\mathcal{H}_\infty$ Control and Experimental Validation [Text in Portuguese]. *In: SBA. ANAIS do XVII Simpósio Brasileiro de Automação Inteligente*. [S. l.: s. n.], 2025.

SOUZA, Carla de; LEITE, Valter JS; TARBOURIECH, Sophie; CASTELAN, Eugenio B; SILVA, Luis FP. A direct parameter-error codesign approach of discrete-time saturated LPV systems. **IEEE Transactions on Automatic Control**, IEEE, v. 67, n. 9, p. 4870–4876, 2022.

STANIMIROVIĆ, Ivan P. Full-rank block LDL decomposition and the inverses of $n \times n$ block matrices. **Journal of Applied Mathematics and Computing**, v. 40, p. 569–586, 2012.

STANIMIROVIĆ, Ivan P; TASIĆ, Milan B. Computation of generalized inverses by using the LDL decomposition. **Applied Mathematics Letters**, v. 25, n. 3, p. 526–531, 2012.

STELLATO, Bartolomeo; BANJAC, Goran; GOULART, Paul; BEMPORAD, Alberto; BOYD, Stephen. OSQP: An operator splitting solver for quadratic programs. **Mathematical Programming Computation**, Springer, v. 12, n. 4, p. 637–672, 2020.

VELARDE, Pablo; VALVERDE, Luis; MAESTRE, Jose Maria; OCAMPO-MARTÍNEZ, Carlos; BORDONS, Carlos. On the comparison of stochastic model predictive control strategies applied to a hydrogen-based microgrid. **Journal of Power Sources**, Elsevier, v. 343, p. 161–173, 2017.

VERGARA-DIETRICH, José D; MORATO, Marcelo Menezes; MENDES, Paulo Renato da Costa; CANI, Alex A; NORMEY-RICO, Julio Elias; BORDONS, Carlos. Advanced chance-constrained predictive control for the efficient energy management of renewable power systems. **Journal of Process Control**, Elsevier, v. 74, p. 120–132, 2019.

VERSCHUEREN, Robin *et al.* acados—a modular open-source framework for fast embedded optimal control. **Mathematical Programming Computation**, Springer, v. 14, n. 1, p. 147–183, 2022.

WÄCHTER, Andreas; BIEGLER, Lorenz T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. **Mathematical programming**, Springer, v. 106, n. 1, p. 25–57, 2006.

WANG, Yang; BOYD, Stephen. Fast model predictive control using online optimization. **IEEE Transactions on control systems technology**, v. 18, n. 2, p. 267–278, 2009.

YAMAZAKI, Ichitaro; NOOSHABADI, Saeid; TOMOV, Stanimire; DONGARRA, Jack. Structure-aware linear solver for realtime convex optimization for embedded systems. **IEEE Embedded Systems Letters**, IEEE, v. 9, n. 3, p. 61–64, 2017.

ZANETTE, Daniel V; PALMEIRA, Alessandra H K; MOREIRA, Luciano G; DA SILVA, Joao M Gomes. Periodic event-trigger control for continuous-time LPV systems. *In: 2021 European Control Conference (ECC)*. [S. l.]: IEEE, 2021. p. 156–161.

APPENDIX A – EXAMPLES OF QUASI-LPV EMBEDDING

A.1 LINEAR DIFFERENTIAL INCLUSION (LDI) PROPERTY

Any nonlinear system can be represented by an Linear Parameter Varying (LPV) system using the linear differential inclusion (LDI), since it respects the LDI property, which is presented in the following (Abbas *et al.*, 2014; Boyd *et al.*, 1994).

Consider a generic discrete-time nonlinear system with states $h \in \mathbb{R}^{n_h}$, measured outputs $y \in \mathbb{R}^{n_y}$, and control signals $u \in \mathbb{R}^{n_u}$:

$$h(k+1) = f_x(h(k), u(k)), \quad (31)a$$

$$y(k) = f_y(h(k), u(k)). \quad (31)b$$

According to Boyd *et al.* (1994), the Linear Differential Inclusion (LDI) property enables the transformation of such nonlinear systems into Linear Parameter-Varying (LPV) formulations. This property states that for every state h , output y , and control input u at any given time instant k , there exists a matrix $G(h, u, k) \in \mathcal{G}$ such that:

$$\begin{bmatrix} f_x(h(k), u(k)) \\ f_y(h(k), u(k)) \end{bmatrix} = G(h(k), u(k)) \begin{bmatrix} h(k) \\ u(k) \end{bmatrix}, \quad (32)$$

where $\mathcal{G} \subseteq \mathbb{R}^{(n_h+n_y) \times (n_h+n_u)}$.

Following the definition of the LDI property, Morato, Normey-Rico, and Sename (2020a) provides an example of how nonlinear systems can be cast into LPV representations. This case and other additional are detailed in this Appendix to provide further theoretical background on nonlinear system modelling, supporting the methods employed throughout this dissertation.

A.2 GENERIC EXAMPLE

Consider a cascade of two cylindrical water tank system given by the following equations (Morato; Normey-Rico; Sename, 2020b):

$$A_b \frac{dh_1(t)}{dt} = u(t) - \sqrt{2g} \sqrt{h_1(t)}, \quad (33)$$

$$A_b \frac{dh_2(t)}{dt} = \sqrt{2g} \sqrt{h_1(t)} - \sqrt{2g} \sqrt{h_2(t)}, \quad (34)$$

where A_b is the base area (equal for both tanks), u is the inlet flow of water in the first tank, h_1 and h_2 is the level of water of each tank and g is the gravity.

Using Euler discretization with a T_s sample, we can represent it in the state-space formulation:

$$\begin{aligned} \overbrace{\begin{bmatrix} h_1(k+1) \\ h_2(k+1) \end{bmatrix}}^{h(k+1)} &= f_x(h(k)) \\ &= \begin{bmatrix} h_1(k) - \frac{T_s}{A_b} \sqrt{2g} \sqrt{h_1(k)} + \frac{T_s}{A_b} u(k) \\ h_2(k) - \frac{T_s}{A_b} \sqrt{2g} \sqrt{h_2(k)} + \frac{T_s}{A_b} \sqrt{2g} \sqrt{h_1(k)} \end{bmatrix}, \end{aligned} \quad (35)$$

$$y(k) = f_y(h(k)) = h_2(k). \quad (36)$$

The LDI property is verified:

$$\begin{aligned} &\begin{bmatrix} f_x(h(k)) \\ f_y(h(k)) \end{bmatrix} \\ &= \overbrace{\begin{bmatrix} \left(1 - \frac{T_s \sqrt{2g} \sqrt{h_1(k)}}{A_b} \right) & 0 \\ \left(\frac{T_s \sqrt{2g} \sqrt{h_1(k)}}{A_b}\right) & \left(1 - \frac{T_s \sqrt{2g} \sqrt{h_2(k)}}{A_b}\right) \\ 0 & 1 \end{bmatrix}}^{G(h(k))} \begin{bmatrix} \frac{T_s}{A_b} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} h(k) \\ u(k) \end{bmatrix}, \end{aligned} \quad (37)$$

since $G(h(k))$ always exists under the assumption that the discontinuity $h_1 = h_2 = 0$ is avoided.

For a State-Space (SS) LPV representation, we define the scheduling vector $P_k = \begin{bmatrix} \rho_1(k) & \rho_2(k) \end{bmatrix}^T$, where:

$$\rho_1(k) = \frac{T_s \sqrt{2g} \sqrt{h_1(k)}}{A_b}, \quad \rho_2(k) = \frac{T_s \sqrt{2g} \sqrt{h_2(k)}}{A_b}. \quad (38)$$

Assuming $x(k) = h(k)$, the dynamics are governed by:

$$x(k+1) = A(\rho(k))x(k) + Bu(k), \quad (39)$$

where the parameter-dependent matrix $A(\rho(k))$ and the input matrix B are given by:

$$A(\rho(k)) = I_2 + \begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix} \text{diag}\{\rho(k)\}, \quad B = \begin{bmatrix} \frac{T_s}{A_b} \\ 0 \end{bmatrix}. \quad (40)$$

A.3 GENERALIZATION OF THE LDI PROPERTY FOR COMMON NONLINEARITIES

The systematic procedure applied to the coupled-tank system can be generalized to various types of nonlinearities. The core objective is to factor the nonlinear function $f(x)$ such that it can be expressed as $f(x) = \rho(x)x$, where $\rho(x)$ is defined as the scheduling parameter.

For instance, consider the following common cases usually encountered in nonlinear systems:

- **Trigonometric Functions:** For systems involving $\sin(\theta)$, one can define:

$$\rho(\theta) = \frac{\sin(\theta)}{\theta}, \quad \text{for } \theta \neq 0, \quad (41)$$

yielding the linear-like form $\sin(\theta) = \rho(\theta)\theta$.

- **Power and Root Laws:** Generalizing the square-root nonlinearity, a term x^n can be handled as:

$$x^n = \underbrace{(x^{n-1})}_{\rho(x)} x. \quad (42)$$

- **Product of Variables:** For terms involving the product of two states $x_1 x_2$, one can choose either x_1 or x_2 to be the scheduling parameter, depending on the desired structure of the $A(\rho)$ matrix.

By applying this "factorization" technique, the nonlinear dynamics are embedded into a quasi-LPV framework, allowing the use of the `LPV-PILOT` solver (Section 3) or other LPV-based control synthesis methods discussed in this work.

APPENDIX B – ALGORITHMS

Algorithm 4 LDLFactorization procedure

```

1: procedure LDLFACTORIZATION( $A$ )
2:    $n \leftarrow$  number of rows of  $A$ 
3:    $L \leftarrow I_n$   $\triangleright$  Identity matrix
4:    $d \leftarrow$  zero vector of length  $n$   $\triangleright$  Just the diagonal
5:   for  $i = 1$  to  $n$  do
6:      $d_i \leftarrow A_{i,i} - \sum_{k=1}^{i-1} L_{i,k}^2 \cdot d_k$ 
7:     for  $j = i + 1$  to  $n$  do
8:        $L_{j,i} \leftarrow \frac{A_{j,i} - \sum_{k=1}^{i-1} L_{j,k} \cdot d_k \cdot L_{i,k}}{d_i}$ 
9:    $D \leftarrow d$ 
10:  return  $L, D$ 

```

Algorithm 5 ForwardSubstitution procedure

```

1: procedure FORWARDSUBSTITUTION( $L, b$ )
2:    $y \leftarrow$  zero vector
3:   for  $i = 1$  to  $n$  do
4:      $y_i \leftarrow b_i - \sum_{j=1}^{i-1} L_{i,j} \cdot y_j$   $\triangleright$  Does not need to divide by  $L(i,i)$  because it is 1
5:   return  $y$ 

```

Algorithm 6 DiagonalSolve procedure

```

1: procedure DIAGONALSOLVE( $D, y$ )
2:   for  $i = 1$  to  $n$  do
3:      $z_i \leftarrow \frac{y_i}{D_i}$ 
4:   return  $z$ 

```

Algorithm 7 BackSubstitution procedure

```

1: procedure BACKSUBSTITUTION( $U, y$ )
2:    $x \leftarrow$  zero vector
3:   for  $i = n$  down to 1 do
4:      $x_i \leftarrow y_i - \sum_{j=i+1}^n U_{i,j} \cdot x_j$   $\triangleright$  Does not need to divide by  $U(i,i)$  because it is 1
5:   return  $x$ 

```

APPENDIX C – SCHEDULING PARAMETERS FOR THE TRMS MODEL

The scheduling parameters $\rho(t) = [\rho_1(t), \dots, \rho_{14}(t)]^\top$ for the TRMS system in Section 3.3.1 are derived by embedding the system's non-linearities into a quasi-LPV representation. Following the development in Rotondo, Nejjari, and Puig (2013), these parameters are defined as:

1. Tail rotor dynamics (ρ_1, ρ_{12})

The dynamics of the tail fan speed ω_h depend on the torque and friction coefficients:

- $\rho_1(t) = -\frac{1}{J_{tr}} \left(\frac{k_a^2}{R_a} + B_{tr} + \frac{f_1(\omega_h(t))}{\omega_h(t)} \right),$
- $\rho_{12}(t) = \frac{k_a k_1}{J_{tr} R_a}.$

2. Horizontal aero-mechanics (ρ_2 to ρ_7, ρ_{13})

These parameters describe the yaw motion Ω_h and the cross-coupling from the vertical axis:

- $\rho_2(t) = \frac{l_t \cos \theta_v(t) f_2(\omega_h(t)) / \omega_h(t)}{K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F},$
- $\rho_3(t) = -\frac{k_{oh}}{K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F},$
- $\rho_4(t) = -\frac{f_3(\theta_h(t)) / \theta_h(t)}{K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F},$
- $\rho_5(t) = -\frac{k_m(k_a^2/R_a + B_{mr} + f_4(\omega_v(t)) / \omega_v(t)) \cos \theta_v(t)}{J_{mr}(K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F)},$
- $\rho_6(t) = \frac{k_m \omega_v(t) \sin \theta_v(t) (K_D \cos^2 \theta_v(t) - K_E \sin^2 \theta_v(t) - K_F - 2K_E \cos^2 \theta_v(t))}{(K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F)^2},$
- $\rho_7(t) = \frac{f_6(\theta_v(t)) / (\theta_v(t) - \theta_v^0)}{K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F},$
- $\rho_{13}(t) = \frac{k_m \cos \theta_v(t) k_a k_2}{(K_D \cos^2 \theta_v(t) + K_E \sin^2 \theta_v(t) + K_F) R_a J_{mr}}.$

3. Main rotor dynamics (ρ_8, ρ_{14})

The dynamics of the main fan speed ω_v :

- $\rho_8(t) = -\frac{1}{J_{mr}} \left(\frac{k_a^2}{R_a} + B_{mr} + \frac{f_4(\omega_v(t))}{\omega_v(t)} \right),$
- $\rho_{14}(t) = \frac{k_a k_2}{J_{mr} R_a}.$

4. Vertical aero-mechanics (ρ_9 to ρ_{12})

These parameters describe the pitch motion Ω_v :

- $\rho_9(t) = \frac{k_g \cos \theta_v(t) f_5(\omega_v(t)) - \Omega_h(t) K_H \sin \theta_v(t) \cos \theta_v(t)}{J_v},$
- $\rho_{10}(t) = \frac{l_m f_5(\omega_v(t)) / \omega_v(t)}{J_v},$
- $\rho_{11}(t) = -\frac{k_{gv}}{J_v},$
- $\rho_{12}(t) = \frac{g((K_A - K_B) \cos \theta_v(t) - K_C \sin \theta_v(t))}{J_v(\theta_v(t) - \theta_v^0)}.$

The non-linear functions f_1 through f_6 are defined to capture friction and aerodynamic coupling effects as follows (Rotondo; Nejjari; Puig, 2013):

- $f_1(\omega_h)$ and $f_4(\omega_v)$ represent the drag friction of the tail and main propellers, respectively.
- $f_2(\omega_h)$ and $f_5(\omega_v)$ are the aerodynamic forces generated by the tail and main rotors.
- $f_3(\theta_h)$ accounts for the torque produced by the flat cable centrifugal forces.
- $f_6(\theta_v)$ represents the non-linear coupling between the pitch and yaw dynamics observed during experimental identification.

Furthermore, the physical constants and moments of inertia used in the scheduling parameters are defined by the following relations (Rotondo; Nejjari; Puig, 2013):

- J_{tr} and J_{mr} are the moments of inertia of the tail and main propellers.
- B_{tr} and B_{mr} denote the viscous friction coefficients of the tail and main rotors.
- k_{oh} and k_{ov} are the friction coefficients of the beam in the horizontal and vertical planes, respectively.
- k_a is the motor torque constant, while R_a represents the armature resistance of the DC motors.
- k_1 and k_2 are the input constants for the tail and main motors.
- l_m and l_t represent the distances from the pivot to the main and tail rotors.
- k_g is the gyroscopic constant.
- g is the gravitational acceleration.

The inertial parameters K_A through K_F and K_H are defined based on the mass distribution of the system components (beam, counterweight, motors, and shields) according to the following equations (Rotondo; Nejjari; Puig, 2013):

- $K_A = (m_t/2 + m_{tr} + m_{ts})l_t$.
- $K_B = (m_m/2 + m_{mr} + m_{ms})l_m$.
- $K_C = (m_b l_b/2 + m_{cb} l_{cb})$.
- $K_D = (m_m/3 + m_{mr} + m_{ms})l_m^2 + (m_t/3 + m_{tr} + m_{ts})l_t^2$.
- $K_E = m_b l_b^2/3 + m_{cb} l_{cb}^2$.
- $K_F = m_{ms} r_{ms}^2 + m_{ts} r_{ts}^2/2$.
- $K_H = K_A l_t + K_B l_m + m_b l_b^2/2 + m_{cb} l_{cb}$.

Finally, J_v represents the total vertical moment of inertia, and θ_v^0 is the equilibrium pitch angle corresponding to a zero input voltage $u_v = 0$.